EPSON
EXCEED YOUR VISION

CMOS 16-BIT SINGLE CHIP MICROCONTROLLER

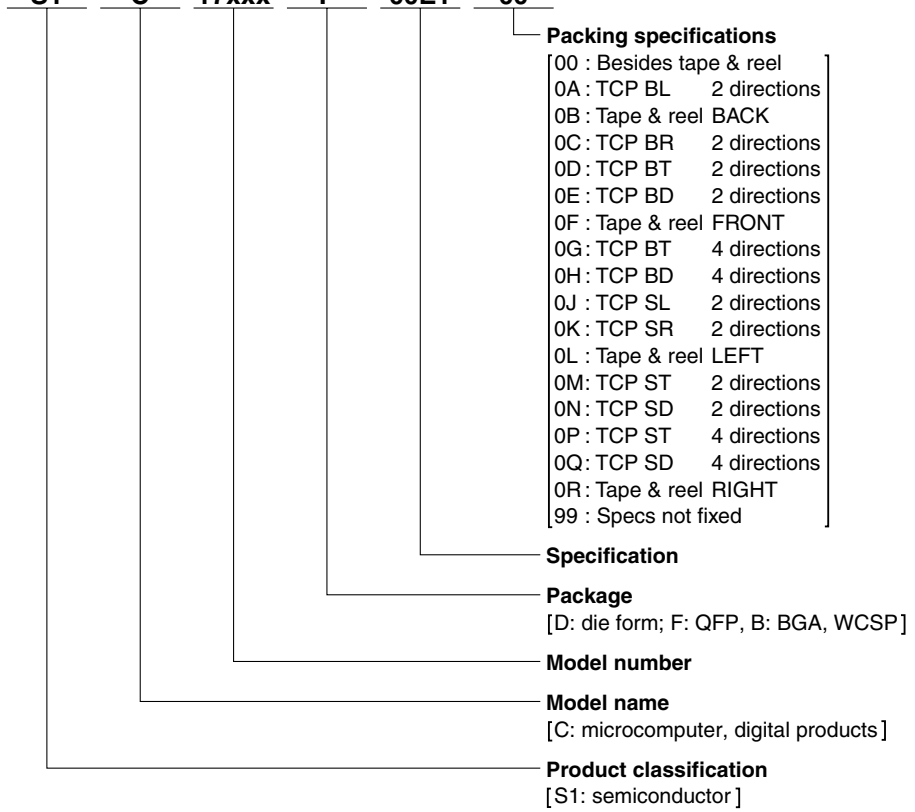
S1C17F13
Technical Manual

SEIKO EPSON CORPORATION

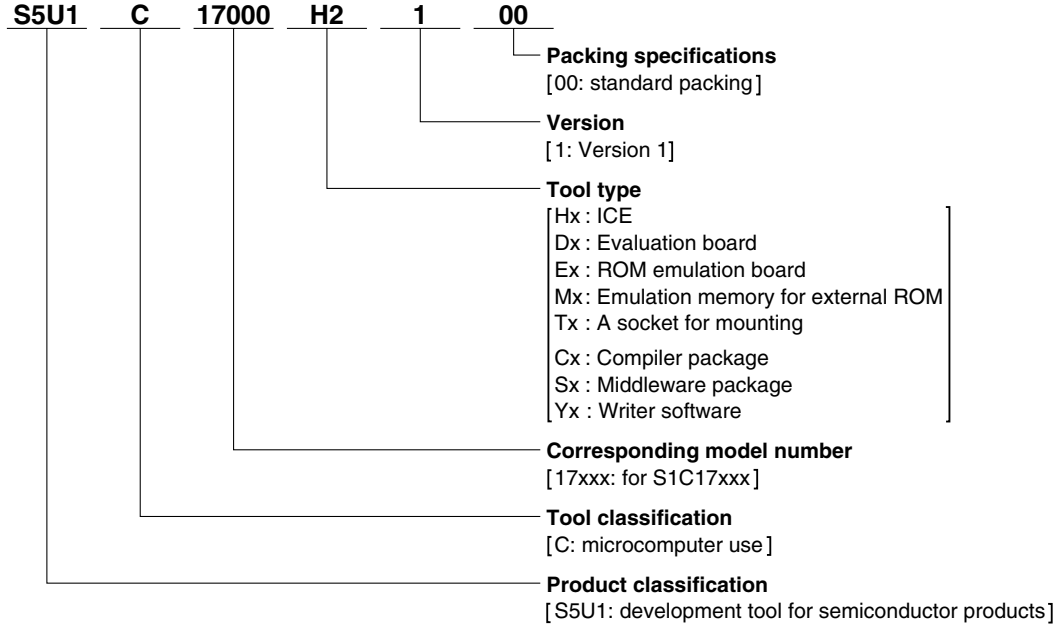
Rev. 1.1

Configuration of product number

Devices



Development tools



Preface

This is a technical manual for designers and programmers who develop a product using the S1C17F13. This document describes the functions of the IC, embedded peripheral circuit operations, and their control methods.

For the CPU functions and instructions, refer to the "S1C17 Family S1C17 Core Manual." For the functions and operations of the debugging tools, refer to the respective tool manuals. (Our "Products: Document Downloads" website provides the downloadable manuals.)

Notational conventions and symbols in this manual

Register address

Peripheral circuit chapters do not provide control register addresses. Refer to "Peripheral Circuit Area" in the "Memory and Bus" chapter or "List of Peripheral Circuit Control Registers" in the Appendix.

Register and control bit names

In this manual, the register and control bit names are described as shown below to distinguish from signal and pin names.

XXX register: Represents a register including its all bits.
XXX.YYY bit: Represents the one control bit YYY in the XXX register.
XXX.ZZZ[1:0] bits: Represents the two control bits ZZZ1 and ZZZ0 in the XXX register.

Register table contents and symbols

Initial: Value set at initialization

Reset: Initialization condition. The initialization condition depends on the reset group (H0, H1, or S0). For more information on the reset groups, refer to "Initialization Conditions (Reset Groups)" in the "Power Supply, Reset, and Clocks" chapter.

R/W: R = Read only bit
W = Write only bit
WP = Write only bit with a write protection using the MSCPROT.PROT[15:0] bits
R/W = Read/write bit
R/WP = Read/write bit with a write protection using the MSCPROT.PROT[15:0] bits

Control bit read/write values

This manual describes control bit values in a hexadecimal notation except for one-bit values (and except when decimal or binary notation is required in terms of explanation). The values are described as shown below according to the control bit width.

1 bit: 0 or 1
2 to 4 bits: 0x0 to 0xf
5 to 8 bits: 0x00 to 0xff
9 to 12 bits: 0x000 to 0xfff
13 to 16 bits: 0x0000 to 0xffff

Decimal: 0 to 9999...
Binary: 0b0000... to 0b1111...

Channel number

Multiple channels may be implemented in some peripheral circuits (e.g., 16-bit timer, etc.). The peripheral circuit chapters use '*n*' as the value that represents the channel number in the register and pin names regardless of the number of channel actually implemented. Normally, the descriptions are applied to all channels. If there is a channel that has different functions from others, the channel number is specified clearly.

Example) T16_*n*CTL register of the 16-bit timer
If one channel is implemented (Ch.0 only): T16_*n*CTL = T16_0CTL only
If two channels are implemented (Ch.0 and Ch.1): T16_*n*CTL = T16_0CTL and T16_1CTL

For the number of channels implemented in the peripheral circuits of this IC, refer to "Features" in the "Overview" chapter.

– Contents –

Configuration of product number i
Preface ii
Notational conventions and symbols in this manual ii

1 Overview 1-1
1.1 Features 1-1
1.2 Block Diagram 1-3
1.3 Pins 1-4
1.3.1 Pin Configuration Diagram (TQFP13-64pin) 1-4
1.3.2 Pad Configuration Diagram (Chip) 1-5
1.3.3 Pin Descriptions 1-6

2 Power Supply, Reset, and Clocks 2-1
2.1 Power Generator (PWG) 2-1
2.1.1 Overview 2-1
2.1.2 Pins 2-2
2.1.3 V_{D1} Regulator 2-2
2.1.4 V_{OSC} Regulator 2-2
2.1.5 Flash Programming Power Supply (V_{PP}) 2-2
2.2 System Reset Controller (SRC) 2-2
2.2.1 Overview 2-2
2.2.2 Input Pin 2-3
2.2.3 Reset Sources 2-3
2.2.4 Initialization Conditions (Reset Groups) 2-4
2.3 Clock Generator (CLG) 2-5
2.3.1 Overview 2-5
2.3.2 Input/Output Pins 2-6
2.3.3 Clock Sources 2-6
2.3.4 Operations 2-8
2.4 Operating Mode 2-11
2.4.1 Initial Boot Sequence 2-11
2.4.2 Transition between Operating Modes 2-11
2.5 Interrupts 2-12
2.6 Control Registers 2-13
PWG V_{D1} Regulator Control Register 2-13
CLG System Clock Control Register 2-13
CLG Oscillation Control Register 2-14
CLG OSC3B Control Register 2-15
CLG OSC1 Control Register 2-15
CLG OSC3A Control Register 2-16
CLG Interrupt Flag Register 2-16
CLG Interrupt Enable Register 2-17
CLG FOUT Control Register 2-17

3 CPU and Debugger 3-1
3.1 Overview 3-1
3.2 CPU Core 3-2
3.2.1 CPU Registers 3-2
3.2.2 Instruction Set 3-2
3.2.3 Reading PSR 3-2
3.2.4 I/O Area Reserved for the S1C17 Core 3-2

3.3 Debugger 3-2
3.3.1 Debugging Functions 3-2
3.3.2 Resource Requirements and Debugging Tools 3-2
3.3.3 List of debugger input/output pins 3-3
3.3.4 External Connection 3-3
3.4 Control Register 3-3
MISC PSR Register 3-3
Debug RAM Base Register 3-4

4 Memory and Bus 4-1
4.1 Overview 4-1
4.2 Bus Access Cycle 4-1
4.3 Flash Memory 4-2
4.3.1 Flash Bus Access Cycle Setting 4-2
4.3.2 Flash Programming 4-2
4.3.3 Flash Security Function 4-3
4.4 RAM1 4-3
4.5 RAM2 4-3
4.6 Peripheral Circuit Control Registers 4-3
4.6.1 System-Protect Function 4-7
4.7 Control Registers 4-7
MISC System Protect Register 4-7
MISC IRAM Size Register 4-7
FLASHC Flash Read Cycle Register 4-8

5 Interrupt Controller (ITC) 5-1
5.1 Overview 5-1
5.2 Vector Table 5-1
5.2.1 Vector Table Base Address (TTBR) 5-3
5.3 Initialization 5-3
5.4 Maskable Interrupt Control and Operations 5-3
5.4.1 Peripheral Circuit Interrupt Control 5-3
5.4.2 ITC Interrupt Request Processing 5-4
5.4.3 Conditions to Accept Interrupt Requests by the CPU 5-4
5.5 NMI 5-4
5.6 Software Interrupts 5-4
5.7 Interrupt Processing by the CPU 5-5
5.8 Control Registers 5-5
MISC Vector Table Address Low Register 5-5
MISC Vector Table Address High Register 5-5
ITC Interrupt Level Setup Register *x* 5-5

6 I/O Ports (PPORT) 6-1
6.1 Overview 6-1
6.2 I/O Cell Structure and Functions 6-2
6.2.1 Schmitt Input 6-2
6.2.2 Over Voltage Tolerant Fail-Safe Type I/O Cell 6-2
6.2.3 Pull-Up/Pull-Down 6-2
6.2.4 CMOS Output and High Impedance State 6-3
6.3 Clock Settings 6-3
6.3.1 PPORT Operating Clock 6-3
6.3.2 Clock Supply in SLEEP Mode 6-3
6.3.3 Clock Supply in DEBUG Mode 6-3

6.4 Operations 6-3
6.4.1 Initialization 6-3
6.4.2 Port Input/Output Control 6-5
6.5 Interrupts 6-6
6.6 Control Registers 6-6
Px Port Data Register 6-6
Px Port Enable Register 6-7
Px Port Pull-up/down Control Register 6-7
Px Port Interrupt Flag Register 6-8
Px Port Interrupt Control Register 6-8
Px Port Chattering Filter Enable Register 6-8
Px Port Mode Select Register 6-8
Px Port Function Select Register 6-9
P Port Clock Control Register 6-9
P Port Interrupt Flag Group Register 6-10
6.7 Control Register and Port Function Configuration of this IC 6-11
6.7.1 P0 Port Group 6-11
6.7.2 P1 Port Group 6-12
6.7.3 P2 Port Group 6-12
6.7.4 P3 Port Group 6-13
6.7.5 P4 Port Group 6-14
6.7.6 Pd Port Group 6-15
6.7.7 Common Registers between Port Groups 6-15

7 Watchdog Timer (WDT) 7-1

7.1 Overview 7-1
7.2 Clock Settings 7-1
7.2.1 WDT Operating Clock 7-1
7.2.2 Clock Supply in DEBUG Mode 7-2
7.3 Operations 7-2
7.3.1 WDT Control 7-2
7.3.2 Operations in HALT and SLEEP Modes 7-2
7.4 Control Registers 7-3
WDT Clock Control Register 7-3
WDT Control Register 7-3

8 Supply Voltage Detector (SVD) 8-1

8.1 Overview 8-1
8.2 Input Pin and External Connection 8-2
8.2.1 Input Pin 8-2
8.2.2 External Connection 8-2
8.3 Clock Settings 8-2
8.3.1 SVD Operating Clock 8-2
8.3.2 Clock Supply in SLEEP Mode 8-2
8.3.3 Clock Supply in DEBUG Mode 8-3
8.4 Operations 8-3
8.4.1 SVD Control 8-3
8.4.2 SVD Operations 8-4
8.5 SVD Interrupt and Reset 8-4
8.5.1 SVD Interrupt 8-4
8.5.2 SVD Reset 8-5
8.6 Control Registers 8-5
SVD Clock Control Register 8-5
SVD Control Register 8-6

SVD Status and Interrupt Flag Register 8-7
SVD Interrupt Enable Register 8-8

9 16-bit Timers (T16) 9-1

9.1 Overview 9-1
9.2 Input Pin 9-1
9.3 Clock Settings 9-2
9.3.1 T16 Operating Clock 9-2
9.3.2 Clock Supply in SLEEP Mode 9-2
9.3.3 Clock Supply in DEBUG Mode 9-2
9.3.4 Event Counter Clock 9-2
9.4 Operations 9-2
9.4.1 Initialization 9-2
9.4.2 Counter Underflow 9-3
9.4.3 Operations in Repeat Mode 9-3
9.4.4 Operations in One-shot Mode 9-3
9.4.5 Counter Value Read 9-4
9.5 Interrupt 9-4
9.6 Control Registers 9-4
T16 Ch.*n* Clock Control Register 9-4
T16 Ch.*n* Mode Register 9-5
T16 Ch.*n* Control Register 9-5
T16 Ch.*n* Reload Data Register 9-6
T16 Ch.*n* Counter Data Register 9-6
T16 Ch.*n* Interrupt Flag Register 9-6
T16 Ch.*n* Interrupt Enable Register 9-7

10 UART (UART) 10-1

10.1 Overview 10-1
10.2 Input/Output Pins and External Connections 10-2
10.2.1 List of Input/Output Pins 10-2
10.2.2 External Connections 10-2
10.2.3 Input Pin Pull-Up Function 10-2
10.2.4 Output Pin Open-Drain Output Function 10-2
10.3 Clock Settings 10-2
10.3.1 UART Operating Clock 10-2
10.3.2 Clock Supply in SLEEP Mode 10-2
10.3.3 Clock Supply in DEBUG Mode 10-3
10.3.4 Baud Rate Generator 10-3
10.4 Data Format 10-3
10.5 Operations 10-4
10.5.1 Initialization 10-4
10.5.2 Data Transmission 10-4
10.5.3 Data Reception 10-5
10.5.4 IrDA Interface 10-6
10.6 Receive Errors 10-7
10.6.1 Framing Error 10-7
10.6.2 Parity Error 10-8
10.6.3 Overrun Error 10-8
10.7 Interrupts 10-8
10.8 Control Registers 10-8
UART Ch.*n* Clock Control Register 10-8
UART Ch.*n* Mode Register 10-9
UART Ch.*n* Baud–Rate Register 10-10

UART Ch.*n* Control Register 10-10
UART Ch.*n* Transmit Data Register 10-11
UART Ch.*n* Receive Data Register 10-11
UART Ch.*n* Status and Interrupt Flag Register 10-11
UART Ch.*n* Interrupt Enable Register 10-12

11 Synchronous Serial Interface (SPI) 11-1

11.1 Overview 11-1
11.2 Input/Output Pins and External Connections 11-2
11.2.1 List of Input/Output Pins 11-2
11.2.2 External Connections 11-2
11.2.3 Pin Functions in Master Mode and Slave Mode 11-3
11.2.4 Input Pin Pull-Up/Pull-Down Function 11-3
11.3 Clock Settings 11-3
11.3.1 SPI Operating Clock 11-3
11.3.2 Clock Supply in DEBUG Mode 11-4
11.3.3 SPI Clock (SPICLK*n*) Phase and Polarity 11-4
11.4 Data Format 11-5
11.5 Operations 11-5
11.5.1 Initialization 11-5
11.5.2 Data Transmission in Master Mode 11-5
11.5.3 Data Reception in Master Mode 11-7
11.5.4 Terminating Data Transfer in Master Mode 11-8
11.5.5 Data Transfer in Slave Mode 11-8
11.5.6 Terminating Data Transfer in Slave Mode 11-10
11.6 Interrupts 11-10
11.7 Control Registers 11-11
SPI Ch.*n* Mode Register 11-11
SPI Ch.*n* Control Register 11-11
SPI Ch.*n* Transmit Data Register 11-12
SPI Ch.*n* Receive Data Register 11-12
SPI Ch.*n* Interrupt Flag Register 11-12
SPI Ch.*n* Interrupt Enable Register 11-13

12 I^2C (I2C) 12-1

12.1 Overview 12-1
12.2 Input/Output Pins and External Connections 12-2
12.2.1 List of Input/Output Pins 12-2
12.2.2 External Connections 12-2
12.3 Clock Settings 12-3
12.3.1 I2C Operating Clock 12-3
12.3.2 Clock Supply in DEBUG Mode 12-3
12.3.3 Baud Rate Generator 12-3
12.4 Operations 12-4
12.4.1 Initialization 12-4
12.4.2 Data Transmission in Master Mode 12-5
12.4.3 Data Reception in Master Mode 12-7
12.4.4 10-bit Addressing in Master Mode 12-9
12.4.5 Data Transmission in Slave Mode 12-10
12.4.6 Data Reception in Slave Mode 12-12
12.4.7 Slave Operations in 10-bit Address Mode 12-14
12.4.8 Automatic Bus Clearing Operation 12-14
12.4.9 Error Detection 12-15
12.5 Interrupts 12-16

12.6 Control Registers 12-17
I2C Ch.*n* Clock Control Register 12-17
I2C Ch.*n* Mode Register 12-18
I2C Ch.*n* Baud-Rate Register 12-18
I2C Ch.*n* Own Address Register 12-18
I2C Ch.*n* Control Register 12-19
I2C Ch.*n* Transmit Data Register 12-20
I2C Ch.*n* Receive Data Register 12-20
I2C Ch.*n* Status and Interrupt Flag Register 12-20
I2C Ch.*n* Interrupt Enable Register 12-21

13 Clock Timer (CT) 13-1
13.1 Overview 13-1
13.2 Clock Settings 13-1
13.3 Operations 13-1
13.4 Interrupts 13-2
13.5 Control Registers 13-3
CT Control Register 13-3
CT Counter Data Register 13-3
CT Interrupt Flag Register 13-4
CT Interrupt Enable Register 13-4

14 Real-Time Clock (RTC) 14-1
14.1 Overview 14-1
14.2 Clock Settings 14-1
14.3 RTC Counters 14-1
14.4 Operations 14-3
14.4.1 Time Setting 14-3
14.4.2 Time Read 14-4
14.5 Interrupts 14-4
14.6 Control Registers 14-5
RTC Control Register 14-5
RTC Interrupt Enable Register 14-6
RTC Interrupt Flag Register 14-6
RTC Minute/Second Register 14-7
RTC Hour Register 14-7

15 Theoretical Regulation (TR) 15-1
15.1 Overview 15-1
15.2 Output Pin 15-1
15.3 Operations 15-1
15.3.1 Executing Theoretical Regulation 15-1
15.3.2 Regulated Clock External Monitor 15-3
15.4 Control Register 15-3
Theoretical Regulation Control Register 15-3

16 16-bit PWM Timers (T16A3) 16-1
16.1 Overview 16-1
16.2 Input/Output Pins 16-2
16.3 Clock Settings 16-2
16.3.1 T16A3 Operating Clock 16-2
16.3.2 Clock Supply in SLEEP Mode 16-3
16.3.3 Clock Supply in DEBUG Mode 16-3
16.3.4 Event Counter Clock 16-3

16.4 Operations 16-3
16.4.1 Initialization 16-3
16.4.2 Counter Block Operations 16-4
16.4.3 Comparator/Capture Block Operations 16-5
16.4.4 TOUT Output Control 16-9
16.5 Interrupt 16-11
16.6 Control Registers 16-11
T16A3 Ch.*n* Clock Control Register 16-11
T16A3 Counter Ch.*n* Control Register 16-12
T16A3 Counter Ch.*n* Data Register 16-13
T16A3 Comparator/Capture Ch.*n* Control Register 16-14
T16A3 Comparator/Capture Ch.*n* A Data Register 16-15
T16A3 Comparator/Capture Ch.*n* B Data Register 16-15
T16A3 Ch.*n* Interrupt Flag Register 16-16
T16A3 Ch.*n* Interrupt Enable Register 16-16

17 Parallel Interface (PIO) 17-1

17.1 Overview 17-1
17.2 Input/Output Pins and External Connections 17-1
17.2.1 List of Input/Output Pins 17-1
17.2.2 External Connections 17-2
17.2.3 Pin Pull-Up Function 17-2
17.3 Clock Settings 17-2
17.3.1 PIO Operating Clock 17-2
17.3.2 Clock Supply in SLEEP Mode 17-2
17.3.3 Clock Supply in DEBUG Mode 17-2
17.4 Operations 17-2
17.4.1 Initialization 17-2
17.4.2 Operations in SRAM Mode 17-3
17.4.3 Operations in GPIO Mode 17-4
17.5 Control Registers 17-5
PIO Clock Control Register 17-5
PIO Mode Register 17-5
PIO Control Register 17-6
PIO Address/Write Data Register 17-6
PIO Read Data Register 17-6
PIO Status Register 17-7

18 EPD Timing Controller (EPD Tcon) 18-1

18.1 Overview 18-1
18.2 Interrupt 18-1
18.3 Control Registers 18-2
EPD Tcon Control Register 18-2
EPD Tcon Interrupt Flag and Status Register 18-2
EPD Tcon Interrupt Enable Register 18-2

19 R/F Converter (RFC) 19-1

19.1 Overview 19-1
19.2 Input/Output Pins and External Connections 19-2
19.2.1 List of Input/Output Pins 19-2
19.2.2 External Connections 19-2
19.3 Clock Settings 19-3
19.3.1 RFC Operating Clock 19-3
19.3.2 Clock Supply in SLEEP Mode 19-3
19.3.3 Clock Supply in DEBUG Mode 19-3

19.4 Operations 19-3
19.4.1 Initialization 19-3
19.4.2 Operating Modes 19-4
19.4.3 RFC Counters 19-4
19.4.4 Converting Operations and Control Procedure 19-5
19.4.5 CR Oscillation Frequency Monitoring Function 19-7
19.5 Interrupts 19-7
19.6 Control Registers 19-8
RFC Ch.*n* Clock Control Register 19-8
RFC Ch.*n* Control Register 19-8
RFC Ch.*n* Oscillation Trigger Register 19-9
RFC Ch.*n* Measurement Counter Low and High Registers 19-10
RFC Ch.*n* Time Base Counter Low and High Registers 19-10
RFC Ch.*n* Interrupt Flag Register 19-11
RFC Ch.*n* Interrupt Enable Register 19-11

20 Temperature Detection Circuit (TEM) 20-1
20.1 Overview 20-1
20.2 Clock Settings 20-2
20.2.1 TEM Operating Clock 20-2
20.2.2 Clock Supply in SLEEP Mode 20-2
20.2.3 Clock Supply in DEBUG Mode 20-2
20.3 Operations 20-2
20.3.1 Initialization 20-2
20.3.2 Comparison Time Setting 20-2
20.3.3 Temperature Detection 20-3
20.4 Interrupt 20-5
20.5 Control Registers 20-5
TEM Clock Control Register 20-5
TEM Timing Register 20-5
TEM Control Register 20-6
TEM Conversion Result Register 20-6
TEM Interrupt Flag and Status Register 20-6
TEM Interrupt Enable Register 20-7

21 Multiplier/Divider (COPRO) 21-1
21.1 Overview 21-1
21.2 Operation Mode and Output Mode 21-1
21.3 Multiplication 21-2
21.4 Division 21-3
21.5 MAC 21-4
21.6 Reading Operation Results 21-6

22 Electrical Characteristics 22-1
22.1 Absolute Maximum Ratings 22-1
22.2 Recommended Operating Conditions 22-1
22.3 Current Consumption 22-2
22.4 System Reset Controller (SRC) Characteristics 22-4
22.5 Clock Generator (CLG) Characteristics 22-5
22.6 Flash Memory Characteristics 22-6
22.7 Input/Output Port (PPORT) Characteristics 22-7
22.8 Supply Voltage Detector (SVD) Characteristics 22-8
22.9 UART (UART) Characteristics 22-9
22.10 Synchronous Serial Interface (SPIA) Characteristics 22-9

22.11 I²C (I2C) Characteristics 22-10
22.12 Parallel Interface (PIO) Characteristics 22-10
22.13 EPD Timing Controller (EPD Tcon) Characteristics 22-11
22.14 R/F Converter (RFC) Characteristics 22-11
22.15 Temperature Detection Circuit Characteristics 22-12

23 Basic External Connection Diagram 23-1

24 Package 24-1

Appendix A List of Peripheral Circuit Control Registers AP-A-1
0x4000–0x4008 Misc Registers (MISC) AP-A-1
0x4020 Power Generator (PWG) AP-A-1
0x4040–0x404e Clock Generator (CLG) AP-A-1
0x4052 Theoretical Regulation (TR) AP-A-2
0x4080–0x4092 Interrupt Controller (ITC) AP-A-2
0x40a0–0x40a2 Watchdog Timer (WDT) AP-A-3
0x40c0–0x40c8 Real-time Clock (RTC) AP-A-4
0x4100–0x4106 Supply Voltage Detector (SVD) AP-A-4
0x4160–0x416c 16-bit Timer (T16) Ch.0 ..... AP-A-5
0x41b0 Flash Controller (FLASHC) AP-A-5
0x4200–0x42e2 I/O Ports (PPORT) AP-A-5
0x4380–0x438e UART (UART) AP-A-8
0x43a0–0x43ac 16-bit Timer (T16) Ch.1 ..... AP-A-9
0x43b0–0x43ba SPI (SPI) Ch.0 ..... AP-A-10
0x43c0–0x43d2 I²C (I2C) AP-A-10
0x5000–0x500e 16-bit PWM Timer (T16A3) Ch.0 ..... AP-A-11
0x5020–0x502e 16-bit PWM Timer (T16A3) Ch.1 ..... AP-A-12
0x5180–0x5186 Clock Timer (CT) AP-A-13
0x5260–0x526c 16-bit Timer (T16) Ch.2 ..... AP-A-14
0x5270–0x527a SPI (SPI) Ch.1 ..... AP-A-14
0x5280–0x528c 16-bit Timer (T16) Ch.3 ..... AP-A-15
0x5290–0x529a SPI (SPI) Ch.2 ..... AP-A-15
0x52e0–0x52ea Parallel Interface (PIO) AP-A-16
0x5380–0x5384 EPD Timing Controller (EPD Tcon) AP-A-16
0x5440–0x5450 R/F Converter (RFC) Ch.0 ..... AP-A-17
0x5460–0x5470 R/F Converter (RFC) Ch.1 ..... AP-A-18
0x54c0–0x54ca Temperature Detection Circuit (TEM) AP-A-19
0xffff90 Debugger (DBG) AP-A-19

Appendix B Power Saving AP-B-1
B.1 Operating Status Configuration Examples for Power Saving AP-B-1
B.2 Other Power Saving Methods AP-B-2

Appendix C Mounting Precautions AP-C-1

Appendix D Measures Against Noise AP-D-1

Appendix E Initialization Routine AP-E-1

Revision History

1 Overview

The S1C17F13 is an ultra low-power MCU equipped with a display memory and an EPD timing controller to send display data for using the active EPD panels. This IC includes the synchronous serial interface, parallel interface, UART, and I²C to communicate with an EPD panel and other devices. This IC allows measurement of various environmental conditions such as a temperature and humidity measurement using the R/F converter, and a supply voltage measurement using the supply voltage detector and brownout reset circuits.

1.1 Features

Table 1.1.1 Features

Model	S1C17F13
CPU	
CPU core	Seiko Epson original 16-bit RISC CPU core S1C17
Multiplier/Divider (COPRO)	16-bit × 16-bit multiplier
	16-bit × 16-bit + 32-bit multiply and accumulation unit
	16-bit ÷ 16-bit divider
Other	On-chip debugger
Embedded Flash memory	
Capacity	128K bytes (for both instructions and data) *1
Erase/program count	50 times (min.) * Programming by the debugging tool ICDmini
Other	Security function to protect from reading/programming by ICDmini
	On-board programming function using ICDmini
	Embedded Flash voltage booster to generate the Flash erasing/programming voltage
Embedded RAM	
Capacity	6K bytes (area accessed by CPU only)
	14K bytes (area accessed by CPU and EPD Tcon)
Clock generator (CLG)	
System clock source	5 sources (OSC3B, OSC3A, OSC1B, OSC1A, and EXOSC)
System clock frequency (operating frequency)	20 MHz (max.)
OSC3B internal high-speed oscillator circuit (boot clock source)	20/16/12/8 MHz (typ.) selectable via software
OSC1B internal low-speed oscillator circuit	32 kHz (typ.)
OSC3A high-speed oscillator circuit	20 MHz (max.) crystal or ceramic oscillator circuit
OSC1A low-speed oscillator circuit	32.768 kHz (typ.) crystal oscillator circuit
EXOSC clock input	20 MHz (max.) square or sine wave input
Other	Configurable system clock division ratio
	Configurable system clock (except for OSC1A and OSC1B) used at wake up from SLEEP state
	Operating clock frequency for the CPU and all peripheral circuits is selectable.
I/O port (PPORT)	
Number of general-purpose I/O ports	37 bits (max.) (Pins are shared with the peripheral I/O.)
Number of input interrupt ports	8 bits
Other	All pins contain a pull-up/down resistor that can be enabled/disabled via software.
	16 bits contain an interrupt function and a chattering filter function.
Display control	
EPD timing controller (EPD Tcon)	Controls display on the active-matrix EPD via the embedded SPI or PIO.
	Includes a display data read function from the embedded RAM (area for both CPU and EPD Tcon).
	Can be controlled with the dedicated API library.
Communication interfaces	
UART (UART)	1 channel
	IrDA1.0 supported
	Embedded baud-rate generator
Synchronous serial interface (SPI)	3 channels
	Configurable as the communication interface for EPD Tcon (SPI Ch.1)
I²C (I2C)	1 channel
	Master and slave operations supported
	Embedded baud-rate generator
Parallel interface (PIO)	Address length: 8 bits (max.)
	Data width: 8 bits (max.)
	Control signals: #CE, #RD, #WR
	Configurable as the communication interface for EPD Tcon

Timers	
Watchdog timer (WDT)	1 channel
	Generates watchdog timer reset.
16-bit timer (T16)	4 channels
	Generates the SPI master clocks. (Ch.1 to Ch.3)
Clock timer (CT)	1 channel
	128–1 Hz counter
Real-time clock (RTC)	Hour, minute, and second counters
Theoretical regulation function (TR)	Time adjustment function in -31/32,768 to +32/32,768 second units (applied to T16A3, CT, and RTC clocks)
	Supports correction value alteration according to temperature variations.
16-bit PWM timer (T16A3)	2 channels
	PWM output, event counter, and count capture functions
Supply voltage detector (SVD)	
Detection level	19 values (1.8 to 3.6 V)
Other	Intermittent operation mode
	Generates an interrupt or reset according to the detection level evaluation.
R/F converter (RFC)	
Conversion method	CR oscillation type with 24-bit counters
Number of conversion channels	2 channels (Up to four sensors can be connected.)
Supported sensors	DC-bias resistive sensors and AC-bias resistive sensors
Temperature detection circuit (TEM)	
Resolution/accuracy	1 °C steps, ±5 °C accuracy
Reset	
#RESET pin	Reset when the reset pin is set to low.
Power-on reset	Reset at power-on.
Brownout reset	Reset when brownout (V_{DD} = 1.45 V typ.) is detected.
Key entry reset	Reset when the P00 to P01/P02/P03 keys are pressed simultaneously (can be enabled/disabled using a register).
Watchdog timer reset	Reset when the watchdog timer overflows (can be enabled/disabled using a register).
Supply voltage detector reset	Reset when the supply voltage detector detects the set voltage level (can be enabled/disabled using a register).
Interrupt	
Non-maskable interrupt	4 systems (Reset, address misaligned interrupt, debug, NMI)
Programmable interrupt	External interrupt: 1 system (8 levels)
	Internal interrupt: 19 systems (8 levels)
Power supply voltage	
V_{DD} operating voltage	2.0 to 3.6 V
Operating temperature	
Operating temperature range	-20 to 70 °C
Current consumption	
SLEEP mode	0.35 µA OSC1 = OFF, RTC = OFF, OSC3B = OFF, OSC3A = OFF
HALT mode	0.78 µA OSC1 = 32 kHz (OSC1A), RTC = OFF, OSC3B = OFF, OSC3A = OFF
	0.80 µA OSC1 = 32 kHz (OSC1A), RTC = ON, OSC3B = OFF, OSC3A = OFF
RUN mode	11.9 µA OSC1 = 32 kHz (OSC1A), RTC = OFF, OSC3B = OFF, OSC3A = OFF
	5.43 mA OSC1 = OFF, RTC = OFF, OSC3B = OFF, OSC3A = 20 MHz ceramic
	5.50 mA OSC1 = OFF, RTC = OFF, OSC3B = 20 MHz, OSC3A = OFF
Shipping form	
1	TQFP13-64pin (Lead pitch: 0.5 mm)
2	Chip (Pad pitch: 90 µm)

∗1 When using the EPD timing controller (EPD Tcon), an area for storing the timing parameters must be allocated in the Flash memory. When using the internal Flash voltage booster as the Flash programing power supply, an area for storing the control program must be allocated in the Flash memory.

1.2 Block Diagram

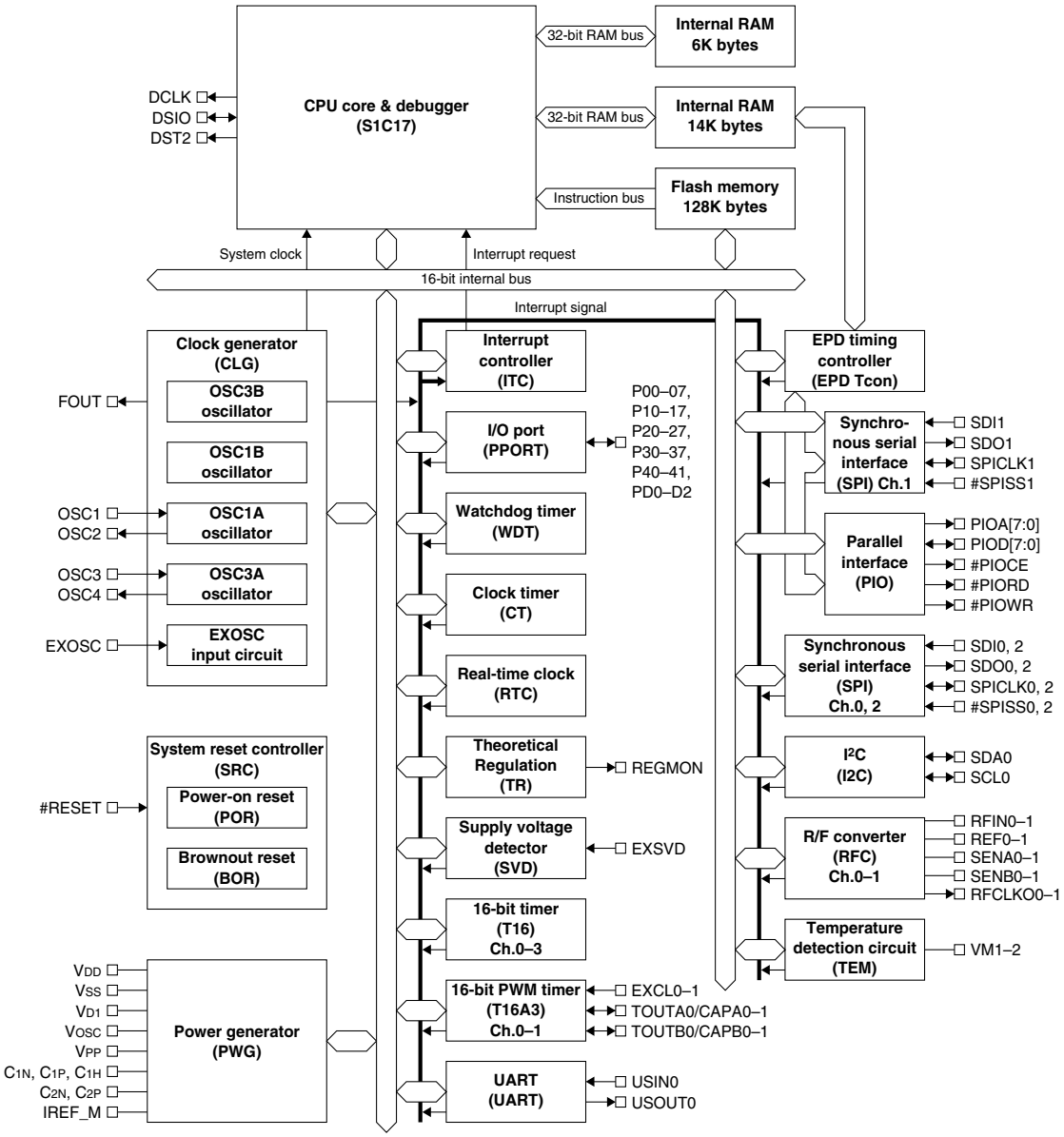


Figure 1.2.1 S1C17F13 Block Diagram

1.3 Pins

1.3.1 Pin Configuration Diagram (TQFP13-64pin)

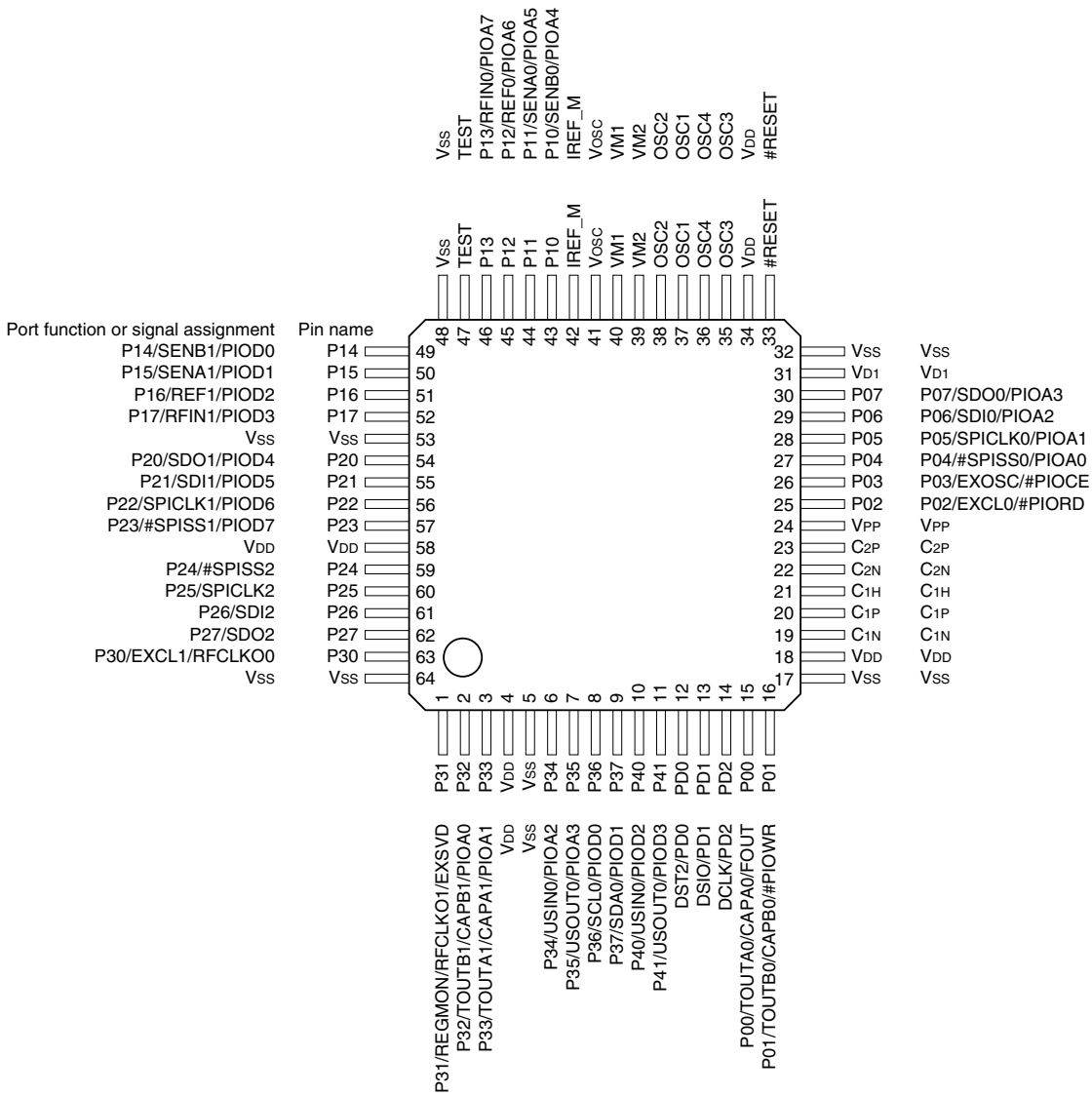


Figure 1.3.1.1 S1C17F13 Pin Configuration Diagram (TQFP13-64pin)

1.3.2 Pad Configuration Diagram (Chip)

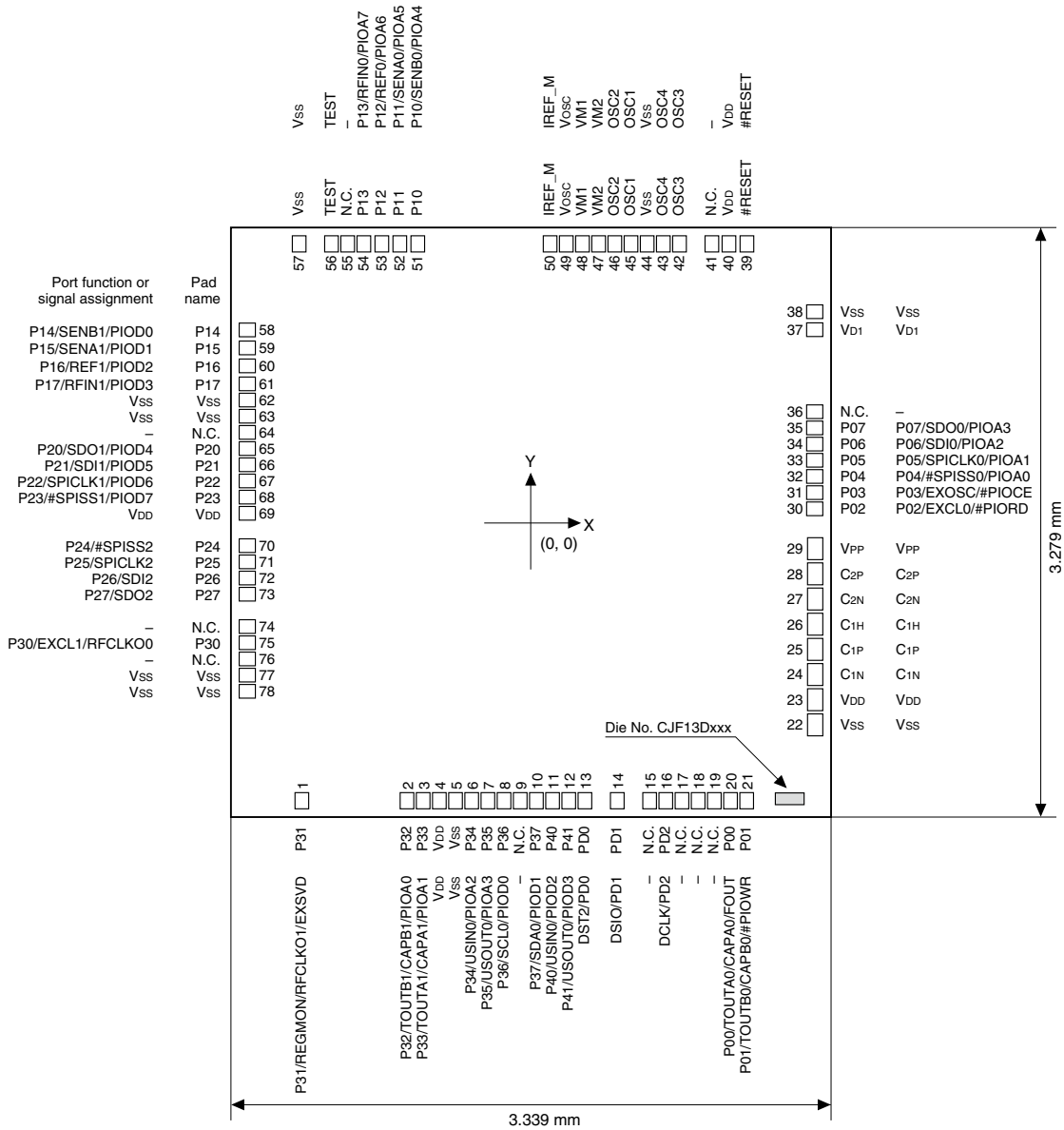


Figure 1.3.2.1 S1C17F13 Pad Configuration Diagram (Chip)

Pad opening No. 1–21, 39–57: X = 76 µm, Y = 90 µm
No. 22–29: X = 85 µm, Y = 122 µm
No. 30–38, 58–78: X = 90 µm, Y = 76 µm
Chip thickness 400 µm

Table 1.3.2.1 Pad Coordinates

No.	X [µm]	Y [µm]	No.	X [µm]	Y [µm]	No.	X [µm]	Y [µm]	No.	X [µm]	Y [µm]
1	-1275.0	-1548.6	22	1581.1	-1126.7	39	1201.3	1548.6	58	-1578.6	1070.0
2	-690.0	-1548.6	23	1581.1	-986.7	40	1101.3	1548.6	59	-1578.6	970.0
3	-600.0	-1548.6	24	1581.1	-846.7	41	1006.3	1548.6	60	-1578.6	870.0
4	-510.0	-1548.6	25	1581.1	-706.7	42	826.3	1548.6	61	-1578.6	770.0
5	-420.0	-1548.6	26	1581.1	-566.7	43	736.3	1548.6	62	-1578.6	680.0
6	-330.0	-1548.6	27	1581.1	-426.7	44	646.3	1548.6	63	-1578.6	590.0
7	-240.0	-1548.6	28	1581.1	-286.7	45	556.3	1548.6	64	-1578.6	500.0
8	-150.0	-1548.6	29	1581.1	-146.7	46	466.3	1548.6	65	-1578.6	410.0
9	-60.0	-1548.6	30	1578.6	75.0	47	376.3	1548.6	66	-1578.6	320.0
10	30.0	-1548.6	31	1578.6	165.0	48	286.3	1548.6	67	-1578.6	230.0
11	120.0	-1548.6	32	1578.6	255.0	49	196.3	1548.6	68	-1578.6	140.0
12	210.0	-1548.6	33	1578.6	345.0	50	106.3	1548.6	69	-1578.6	50.0
13	300.0	-1548.6	34	1578.6	435.0	51	-630.0	1548.6	70	-1578.6	-130.0
14	480.0	-1548.6	35	1578.6	525.0	52	-730.0	1548.6	71	-1578.6	-220.0
15	660.0	-1548.6	36	1578.6	615.0	53	-830.0	1548.6	72	-1578.6	-310.0
16	750.0	-1548.6	37	1578.6	1071.3	54	-930.0	1548.6	73	-1578.6	-400.0
17	840.0	-1548.6	38	1578.6	1171.3	55	-1020.0	1548.6	74	-1578.6	-580.0
18	930.0	-1548.6	–	–	–	56	-1110.0	1548.6	75	-1578.6	-670.0
19	1020.0	-1548.6	–	–	–	57	-1290.0	1548.6	76	-1578.6	-760.0
20	1110.0	-1548.6	–	–	–	–	–	–	77	-1578.6	-850.0
21	1200.0	-1548.6	–	–	–	–	–	–	78	-1578.6	-940.0

1.3.3 Pin Descriptions

Symbol meanings

Assigned signal: The signal listed at the top of each pin is assigned in the initial state. The pin function must be switched via software to assign another signal (see the “I/O Ports” chapter).

I/O:
- I = Input
- O = Output
- I/O = Input/output
- P = Power supply
- A = Analog signal
- Hi-Z = High impedance state

Initial state:
- I (Pull-up) = Input with pulled up
- I (Pull-down) = Input with pulled down
- Hi-Z = High impedance state
- O (H) = High level output
- O (L) = Low level output

Table 1.3.3.1 Pin description

Pin/pad name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function
V_{DD}	V_{DD}	P	–	–	Power supply (+)
V_{SS}	V_{SS}	P	–	–	GND
V_{D1}	V_{D1}	A	–	–	Embedded regulator output (internal circuit operating voltage)
V_{OSC}	V_{OSC}	A	–	–	Embedded regulator output (oscillator circuit operating voltage)
V_{PP}	V_{PP}	P	–	–	Flash programming power supply (Leave the pin open during normal operation.)
C_{1N}	C_{1N}	A	–	–	Capacitor connect pin for Flash voltage booster
C_{1P}	C_{1P}	A	–	–	Capacitor connect pin for Flash voltage booster
C_{1H}	C_{1H}	A	–	–	Capacitor connect pin for Flash voltage booster
C_{2N}	C_{2N}	A	–	–	Capacitor connect pin for Flash voltage booster
C_{2P}	C_{2P}	A	–	–	Capacitor connect pin for Flash voltage booster
IREF_M	IREF_M	A	–	–	IREF constant current monitor pin (Leave the pin open during normal operation.)
VM1	VM1	A	–	–	Temperature sensor voltage monitor pin (Leave the pin open during normal operation.)
VM2	VM2	A	–	–	Temperature sensor voltage monitor pin (Leave the pin open during normal operation.)
OSC1	OSC1	A	–	–	OSC1A oscillator circuit input

Pin/pad name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function
OSC2	OSC2	A	–	–	OSC1A oscillator circuit output
OSC3	OSC3	A	–	–	OSC3A oscillator circuit input
OSC4	OSC4	A	–	–	OSC3A oscillator circuit output
#RESET	#RESET	I	I (Pull-up)	–	Reset input
TEST	TEST	I	I (Pull-down)	–	Test input (Connect to Vss during normal operation.)
P00	P00	I/O	Hi-Z (TBD)	✓	General-purpose I/O port
	TOUTA0/CAPA0	I/O			16-bit PWM timer Ch.0 TOUTA signal output/capture A trigger signal input
	FOUT	O			Clock generator clock output
P01	P01	I/O	Hi-Z (TBD)	✓	General-purpose I/O port
	TOUTB0/CAPB0	I/O			16-bit PWM timer Ch.0 TOUTB signal output/capture B trigger signal input
	#PIOWR	O			Parallel interface write signal output
P02	P02	I/O	Hi-Z (TBD)	✓	General-purpose I/O port
	EXCL0	I			16-bit PWM timer Ch.0 external clock input
	#PIORD	O			Parallel interface read signal output
P03	P03	I/O	Hi-Z (TBD)	✓	General-purpose I/O port
	EXOSC	I			Clock generator external clock input
	#PIOCE	O			Parallel interface chip enable signal output
P04	P04	I/O	Hi-Z (TBD)	✓	General-purpose I/O port
	#SPISS0	I			Synchronous serial interface Ch.0 slave select input
	PIOA0	O			Parallel interface address output
P05	P05	I/O	Hi-Z	✓	General-purpose I/O port
	SPICLK0	I/O			Synchronous serial interface Ch.0 clock input/output
	PIOA1	O			Parallel interface address output
P06	P06	I/O	Hi-Z	✓	General-purpose I/O port
	SDI0	I			Synchronous serial interface Ch.0 data input
	PIOA2	O			Parallel interface address output
P07	P07	I/O	Hi-Z	✓	General-purpose I/O port
	SDO0	O			Synchronous serial interface Ch.0 data output
	PIOA3	O			Parallel interface address output
P10	P10	I/O	Hi-Z	–	General-purpose I/O port
	SENB0	A			R/F converter Ch.0 sensor B oscillation control
	PIOA4	O			Parallel interface address output
P11	P11	I/O	Hi-Z	–	General-purpose I/O port
	SENA0	A			R/F converter Ch.0 sensor A oscillation control
	PIOA5	O			Parallel interface address output
P12	P12	I/O	Hi-Z	–	General-purpose I/O port
	REF0	A			R/F converter Ch.0 reference oscillation control
	PIOA6	O			Parallel interface address output
P13	P13	I/O	Hi-Z	–	General-purpose I/O port
	RFIN0	A			R/F converter Ch.0 oscillation input
	PIOA7	O			Parallel interface address output
P14	P14	I/O	Hi-Z	–	General-purpose I/O port
	SENB1	A			R/F converter Ch.1 sensor B oscillation control
	PIOD0	I/O			Parallel interface data input/output
P15	P15	I/O	Hi-Z	–	General-purpose I/O port
	SENA1	A			R/F converter Ch.1 sensor A oscillation control
	PIOD1	I/O			Parallel interface data input/output
P16	P16	I/O	Hi-Z	–	General-purpose I/O port
	REF1	A			R/F converter Ch.1 reference oscillation control
	PIOD2	I/O			Parallel interface data input/output
P17	P17	I/O	Hi-Z	–	General-purpose I/O port
	RFIN1	A			R/F converter Ch.1 oscillation input
	PIOD3	I/O			Parallel interface data input/output
P20	P20	I/O	Hi-Z	✓	General-purpose I/O port
	SDO1	O			Synchronous serial interface Ch.1 data output
	PIOD4	I/O			Parallel interface data input/output

Pin/pad name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function
P21	P21	I/O	Hi-Z	✓	General-purpose I/O port
	SDI1	I			Synchronous serial interface Ch.1 data input
	PIOD5	I/O			Parallel interface data input/output
P22	P22	I/O	Hi-Z	✓	General-purpose I/O port
	SPICLK1	I/O			Synchronous serial interface Ch.1 clock input/output
	PIOD6	I/O			Parallel interface data input/output
P23	P23	I/O	Hi-Z	✓	General-purpose I/O port
	#SPISS1	I			Synchronous serial interface Ch.1 slave select input
	PIOD7	I/O			Parallel interface data input/output
P24	P24	I/O	Hi-Z	✓	General-purpose I/O port
	#SPISS2	I			Synchronous serial interface Ch.2 slave select input
P25	P25	I/O	Hi-Z	✓	General-purpose I/O port
	SPICLK2	I/O			Synchronous serial interface Ch.2 clock input/output
P26	P26	I/O	Hi-Z	✓	General-purpose I/O port
	SDI2	I			Synchronous serial interface Ch.2 data input
P27	P27	I/O	Hi-Z	✓	General-purpose I/O port
	SDO2	O			Synchronous serial interface Ch.2 data output
P30	P30	I/O	Hi-Z	✓	General-purpose I/O port
	EXCL1	I			16-bit PWM timer Ch.1 external clock input
	RFCLKO0	O			R/F converter Ch.0 clock monitor output
P31	P31	I/O	Hi-Z	–	General-purpose I/O port
	REGMON	O			Theoretical regulation clock monitor output
	RFCLKO1	O			R/F converter Ch.1 clock monitor output
	EXSVD	A			External power supply voltage detection input
P32	P32	I/O	Hi-Z	✓	General-purpose I/O port
	TOUTB1/CAPB1	I/O			16-bit PWM timer Ch.1 TOUTB signal output/capture B trigger signal input
	PIOA0	O			Parallel interface address output
P33	P33	I/O	Hi-Z	✓	General-purpose I/O port
	TOUTA1/CAPA1	I/O			16-bit PWM timer Ch.1 TOUTA signal output/capture A trigger signal input
	PIOA1	O			Parallel interface address output
P34	P34	I/O	Hi-Z	✓	General-purpose I/O port
	USIN0	I			UART Ch.0 data input
	PIOA2	O			Parallel interface address output
P35	P35	I/O	Hi-Z	✓	General-purpose I/O port
	USOUT0	O			UART Ch.0 data output
	PIOA3	O			Parallel interface address output
P36	P36	I/O	Hi-Z	✓	General-purpose I/O port
	SCL0	I/O			I^2C Ch.0 clock input/output
	PIOD0	I/O			Parallel interface data input/output
P37	P37	I/O	Hi-Z	✓	General-purpose I/O port
	SDA0	I/O			I^2C Ch.0 data input/output
	PIOD1	I/O			Parallel interface data input/output
P40	P40	I/O	Hi-Z	✓	General-purpose I/O port
	USIN0	I			UART Ch.0 data input
	PIOD2	I/O			Parallel interface data input/output
P41	P41	I/O	Hi-Z	✓	General-purpose I/O port
	USOUT0	O			UART Ch.0 data output
	PIOD3	I/O			Parallel interface data input/output
PD0	DST2	O	O (L)	✓	On-chip debugger status output
	PD0	I/O			General-purpose I/O port
PD1	DSIO	I/O	I (pull-up)	✓	On-chip debugger data input/output
	PD1	I/O			General-purpose I/O port
PD2	DCLK	O	O (H)	✓	On-chip debugger clock output
	PD2	O			General-purpose I/O port

Note: In the peripheral circuit descriptions, the assigned signal name is used as the pin name.

2 Power Supply, Reset, and Clocks

The power supply, reset, and clocks in this IC are managed by the embedded power generator, system reset controller, and clock generator, respectively.

2.1 Power Generator (PWG)

2.1.1 Overview

PWG is the power generator that controls the internal power supply system to drive this IC with stability and low power. The main features of PWG are outlined below.

- Embedded V_{D1} regulator
  - The V_{D1} regulator generates the V_{D1} voltage to drive internal circuits, this makes it possible to keep current consumption constant independent of the V_{DD} voltage level.
  - The V_{D1} regulator supports two operation modes, normal mode and economy mode, and setting the V_{D1} regulator into economy mode at light loads helps achieve low-power operations.
- Embedded V_{OSC} regulator
  - The V_{OSC} regulator drives the low-speed oscillator circuit in low power consumption and achieves stabilized low-speed clock that is used for timers such as RTC.
- Embedded voltage booster for generating the Flash erasing/programming voltage

Figure 2.1.1.1 shows the PWG configuration.

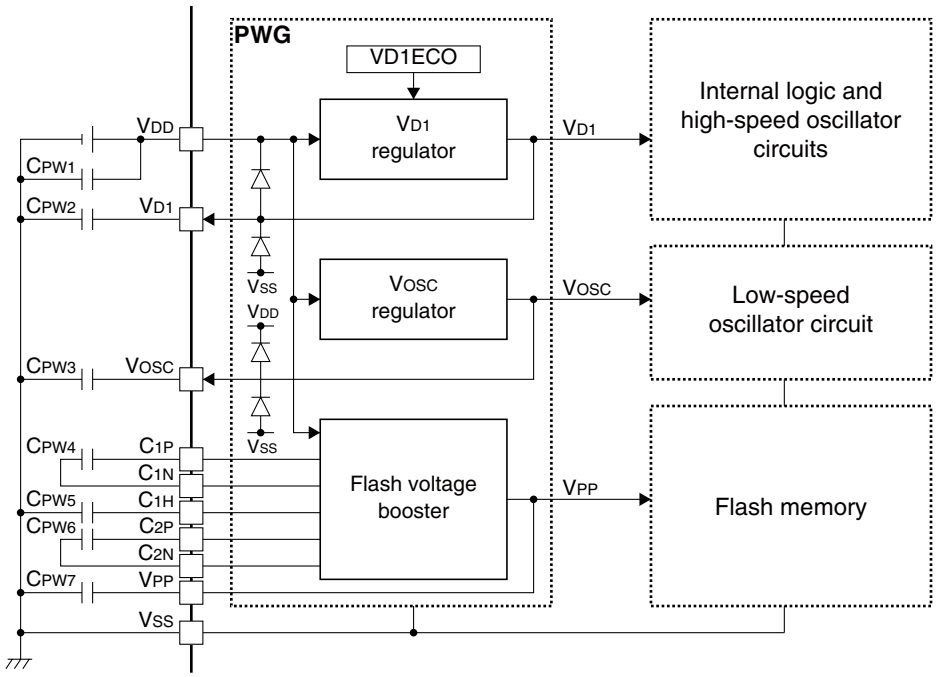


Figure 2.1.1.1 PWG Configuration

2.1.2 Pins

Table 2.1.2.1 lists the PWG pins.

Table 2.1.2.1 List of PWG Pins

Pin name	I/O	Initial status	Function
V_{DD}	P	–	Power supply (+)
V_{SS}	P	–	GND
V_{D1}	A	–	Embedded regulator output (internal circuit operating voltage)
V_{OSC}	A	–	Embedded regulator output (oscillator circuit operating voltage)
V_{PP}	P	–	Flash programming power supply (Leave the pin open during normal operation.)
C_{1P}, C_{1N}, C_{1H}, C_{2P}, C_{2N}	A	–	Capacitor connect pins for Flash voltage booster

For the V_{DD} operating voltage range and recommended external parts, refer to "Recommended Operating Conditions, Power supply voltage V_{DD}" in the "Electrical Characteristics" chapter and the "Basic External Connection Diagram" chapter, respectively.

2.1.3 V_{D1} Regulator

The V_{D1} regulator generates the operating voltage for the internal logic and high-speed oscillator circuits. This regulator always operates. The V_{D1} regulator supports two operation modes, normal mode and economy mode. Setting the V_{D1} regulator into economy mode at light loads helps achieve low-power operations. Table 2.1.3.1 lists examples of light load conditions in which economy mode can be set.

Table 2.1.3.1 Examples of Light Load Conditions in which Economy Mode Can be Set

Light load condition	Exceptions
SLEEP mode (when all oscillators are stopped, or OSC1 only is active)	When a clock source except for OSC1 is active
HALT mode (when OSC1 only is active)	
RUN mode (when OSC1 only is active)	

The V_{D1} regulator also supports automatic mode in which the hardware detects a light load condition and automatically switches between normal mode and economy mode. Use the V_{D1} regulator in automatic mode when no special control is required.

2.1.4 V_{OSC} Regulator

The V_{OSC} regulator generates the operating voltage for the low-speed oscillator circuit. This regulator always operates.

2.1.5 Flash Programming Power Supply (V_{PP})

V_{PP} is the power supply for erasing/programming the embedded Flash memory. The V_{PP} voltage may be applied from an external power supply to the V_{PP} pin or internally generated by the embedded Flash voltage booster (controlled from the Flash programming function of the debugger or self-programming module).
For the V_{PP} voltage value to be applied from an external power supply or the V_{DD} requirements for generating V_{PP} internally, refer to "Recommended Operating Conditions, Flash programming voltage V_{PP} and Power supply voltage V_{DD}" in the "Electrical Characteristics" chapter.

Note: Leave the V_{PP} pin open during normal operation.

2.2 System Reset Controller (SRC)

2.2.1 Overview

SRC is the system reset controller that resets the internal circuits according to the requests from the reset sources to archive steady IC operations. The main features of SRC are outlined below.

- Embedded reset hold circuit maintains reset state to boot the system safely while the internal power supply is unstable after power on or the oscillation frequency is unstable after the clock source is initiated.

- Supports reset requests from multiple reset sources.
  - #RESET pin
  - POR and BOR
  - Key-entry reset
  - Watchdog timer reset
  - Supply voltage detector reset
  - Peripheral circuit software reset (supports some peripheral circuits only)
- The CPU registers and peripheral circuit control bits will be reset with an appropriate initialization condition according to changes in status.

Figure 2.2.1.1 shows the SRC configuration.

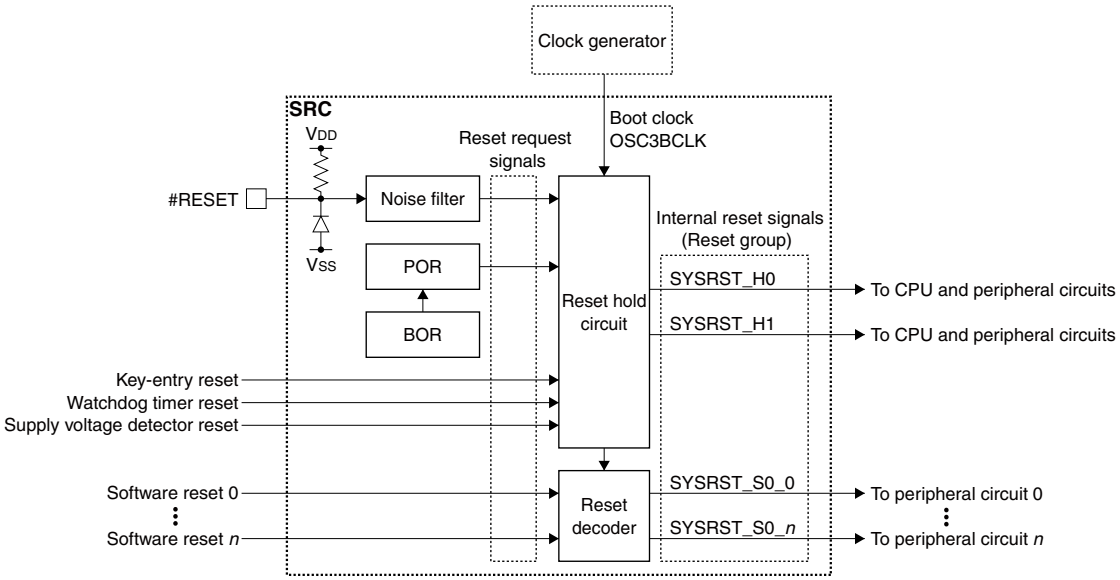


Figure 2.2.1.1 SRC Configuration

2.2.2 Input Pin

Table 2.2.2.1 shows the SRC pin.

Table 2.2.2.1 SRC Pin

Pin name	I/O	Initial status	Function
#RESET	I	I (Pull-up)	Reset input

The #RESET pin is connected to the noise filter that removes pulses not conforming to the requirements. An internal pull-up resistor is connected to the #RESET pin, so the pin can be left open. For the #RESET pin characteristics, refer to "#RESET pin characteristics" in the "Electrical Characteristics" chapter.

2.2.3 Reset Sources

The reset source refers to causes that request system initialization. The following shows the reset sources.

#RESET pin

Inputting a reset signal with a certain low level period to the #RESET pin issues a reset request.

POR and BOR

POR (Power On Reset) issues a reset request when the rise of VDD is detected. BOR (Brownout Reset) issues a reset request when a certain VDD voltage level is detected. Reset requests from these circuits ensure that the system will be reset properly when the power is turned on and the supply voltage is out of the operating voltage range. Figure 2.2.3.1 shows an example of POR and BOR internal reset operation according to variations in VDD.

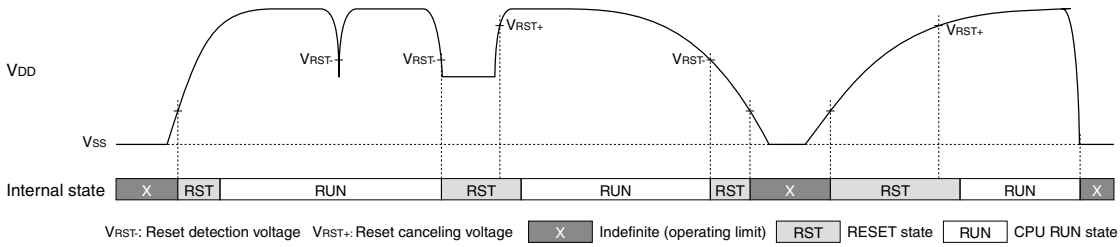


Figure 2.2.3.1 Example of Internal Reset by POR and BOR

For the POR and BOR electrical specifications, refer to "POR/BOR characteristics" in the "Electrical Characteristics" chapter.

Key-entry reset

Inputting a low level signal of a certain period to the I/O port pins configured to a reset input issues a reset request. This function must be enabled using an I/O port register. For more information, refer to the "I/O Ports" chapter.

Watchdog timer reset

The watchdog timer issues a reset request when the counter overflows. This helps return the runaway CPU to a normal operating state. For more information, refer to the "Watchdog timer" chapter.

Supply voltage detector reset

By enabling the low power supply voltage detection reset function, the supply voltage detector will issue a reset request when a drop in the power supply voltage is detected. This makes it possible to put the system into reset state if the IC must be stopped under a low voltage condition. For more information, refer to the "Supply Voltage Detector" chapter.

Peripheral circuit software reset

Some peripheral circuits provide a control bit for software reset (MODEN or SFTRST). Setting this bit initializes the peripheral circuit control bits. Note, however, that the software reset operations depend on the peripheral circuit. For more information, refer to "Control Registers" in each peripheral circuit chapter.

Note: The MODEN bit of some peripheral circuits does not issue software reset.

2.2.4 Initialization Conditions (Reset Groups)

A different initialization condition is set for the CPU registers and peripheral circuit control bits, individually. The reset group refers to an initialization condition. Initialization is performed when a reset source included in a reset group issues a reset request. Table 2.2.4.1 lists the reset groups. For the reset group to initialize the registers and control bits, refer to the "CPU and Debugger" chapter or "Control Registers" in each peripheral circuit chapter.

Table 2.2.4.1 List of Reset Groups

Reset group	Reset source	Reset cancelation timing
H0	#RESET pin POR and BOR Supply voltage detector reset Key-entry reset Watchdog timer reset	Reset state is maintained for the reset hold time t_{RSTR} after the reset request is canceled.
H1	#RESET pin POR and BOR	
S0	Peripheral circuit software reset (MODEN and SFTRST bits. The software reset operations depend on the peripheral circuit.	Reset state is canceled immediately after the reset request is canceled.

2.3 Clock Generator (CLG)

2.3.1 Overview

CLG is the clock generator that controls the clock sources and manages clock supply to the CPU and the peripheral circuits. The main features of CLG are outlined below.

- Supports multiple clock sources.
  - OSC3B oscillator circuit that oscillates with a fast startup and no external parts required
  - Low-power OSC1B oscillator circuit that oscillates with no external parts required
  - High-precision and low-power OSC1A oscillator circuit that uses a 32.768 kHz crystal resonator
  - OSC3A oscillator circuit that supports high-speed crystal and ceramic resonators up to 20 MHz
  - EXOSC clock input circuit that allows input of square wave and sine wave clock signals
- The system clock (SYSCLK), which is used as the operating clock for the CPU and bus, and the peripheral circuit operating clocks can be configured individually by selecting the suitable clock source and division ratio.
- OSC3BCLK output from the OSC3B oscillator circuit is used as the boot clock for fast booting.
- Controls the oscillator and clock input circuits to enable/disable according to the operating mode, RUN or SLEEP mode.
- Provides a flexible system clock switching function at SLEEP mode cancelation.
  - The clock sources to be stopped in SLEEP mode can be selected.
  - SYSCLK to be used at SLEEP mode cancelation can be selected from all clock sources (OSC3B, OSC3A, and EXOSC).
  - The oscillator and clock input circuit on/off state can be maintained or changed at SLEEP mode cancelation.
- Provides the FOUT function to output an internal clock for driving external ICs or for monitoring the internal state.

Figure 2.3.1.1 shows the CLG configuration.

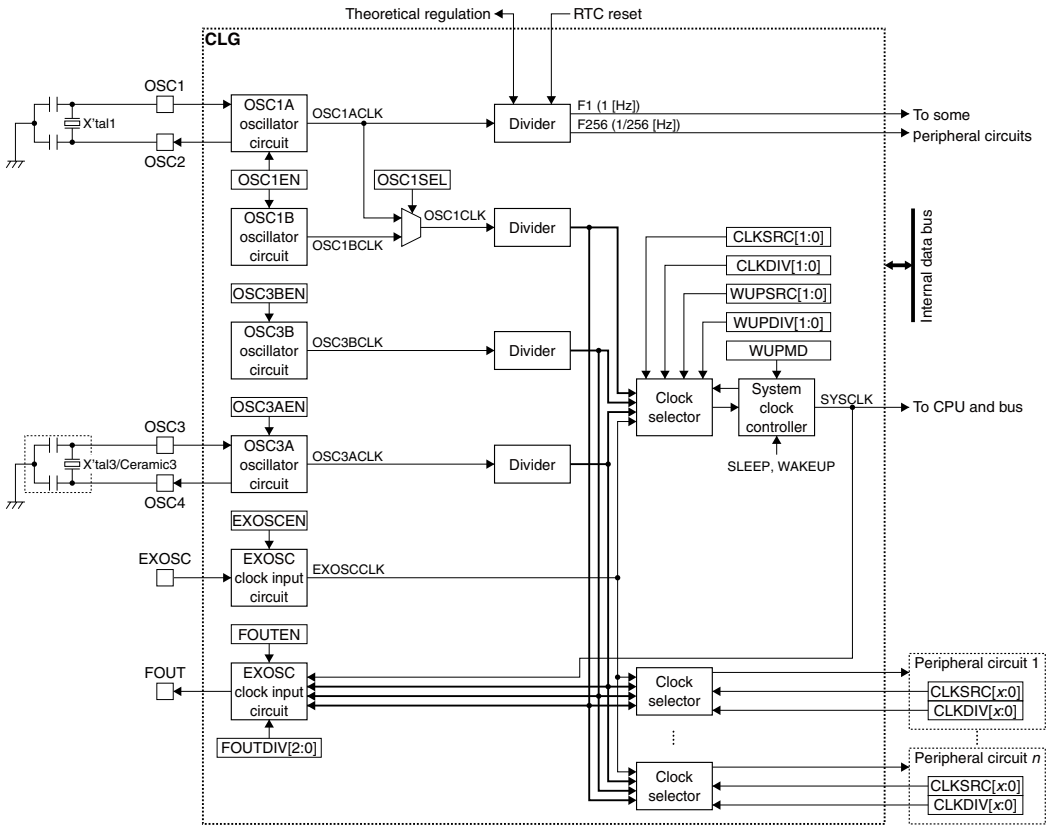


Figure 2.3.1.1 CLG Configuration

2.3.2 Input/Output Pins

Table 2.3.2.1 lists the CLG pins.

Table 2.3.2.1 List of CLG Pins

Pin name	I/O*	Initial status*	Function
OSC1	A	–	OSC1A oscillator circuit input
OSC2	A	–	OSC1A oscillator circuit output
OSC3	A	–	OSC3A oscillator circuit input
OSC4	A	–	OSC3A oscillator circuit output
EXOSC	I	I	EXOSC clock input
FOUT	O	O (L)	FOUT clock output

* Indicates the status when the pin is configured for CLG.

If the port is shared with the CLG input/output function and other functions, the CLG function must be assigned to the port. For more information, refer to the "I/O Ports" chapter.

2.3.3 Clock Sources

OSC3B oscillator circuit

The OSC3B oscillator circuit features a fast startup and no external parts are required for oscillating. Figure 2.3.3.1 shows the configuration of the OSC3B oscillator circuit.

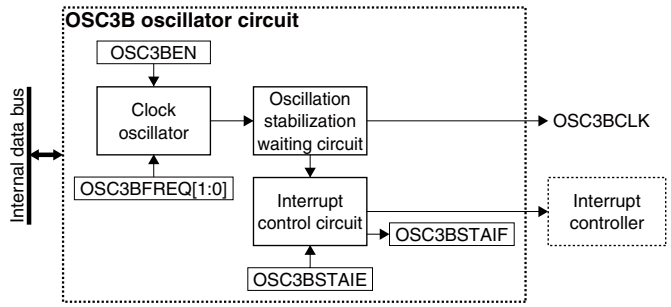


Figure 2.3.3.1 OSC3B Oscillator Circuit Configuration

The OSC3B oscillator circuit output clock OSC3BCLK is used as SYSCLK at booting.
The OSC3BCLK frequency can be selected using the CLGOSC3B.OSC3BFREQ[1:0] bits.
For more information on the oscillation characteristics, refer to "OSC3B oscillator circuit characteristics" in the "Electrical Characteristics" chapter.

OSC3A oscillator circuit

The OSC3A oscillator circuit is a high-speed oscillator circuit that uses a crystal or ceramic resonator. Figure 2.3.3.2 shows the configuration of the OSC3A oscillator circuit.

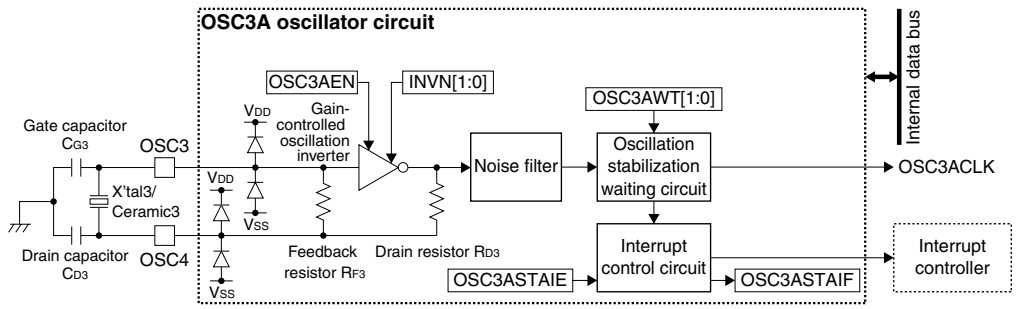


Figure 2.3.3.2 OSC3A Oscillator Circuit Configuration

A crystal resonator (X'tal3) or ceramic resonator (Ceramic3) should be connected between the OSC3 and OSC4 pins. Additionally, two capacitors (C_{G3} and C_{D3}) should be connected between the OSC3/OSC4 pins and V_{SS}. The embedded gain-controlled oscillation inverter allows selection of the resonator from a wide frequency range. For the recommended parts and the oscillation characteristics, refer to the "Basic External Connection Diagram" chapter and "OSC3A oscillator circuit characteristics" in the "Electrical Characteristics" chapter, respectively.

OSC1 oscillator circuit

OSC1 is a low-power oscillator circuit that generates the timer operating clock and the system clock for low-speed operations. The OSC1 oscillator circuit consists of two oscillators, OSC1A and OSC1B, and the oscillator to be used should be selected using the CLGOSC1.OSC1SEL bit. Figure 2.3.3.3 shows the configuration of the OSC1 oscillator circuit.

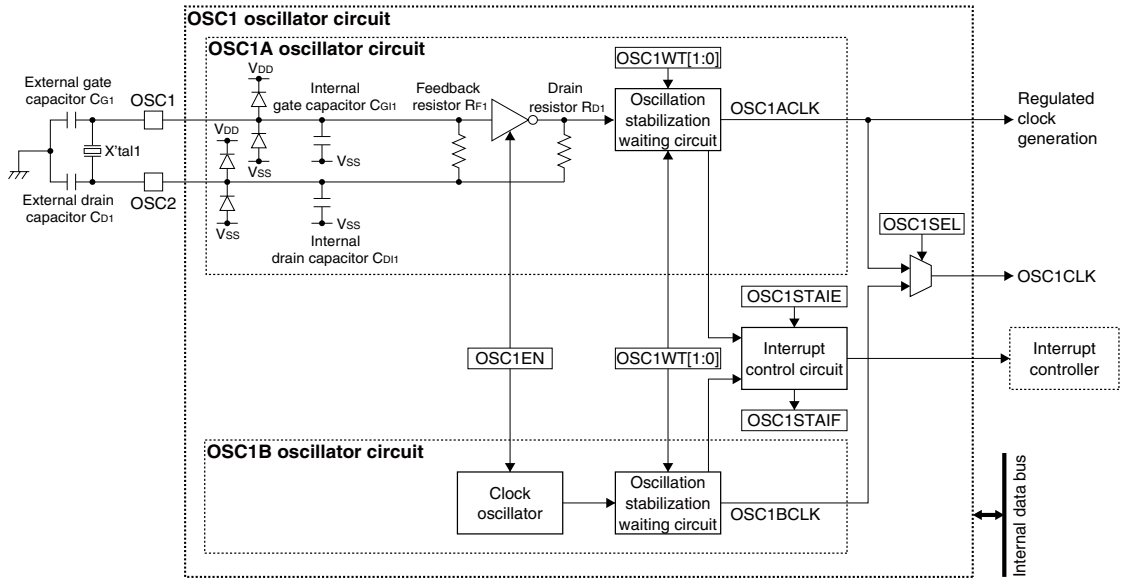


Figure 2.3.3.3 OSC1 Oscillator Circuit Configuration

OSC1A oscillator circuit

The OSC1A oscillator circuit is a high-precision and low-power oscillator circuit that uses a 32.768 kHz crystal resonator. A crystal resonator (X'tal1) should be connected between the OSC1 and OSC2 pins. Additionally, two capacitors (C_{G1} and C_{D1}) should be connected between the OSC1/OSC2 pins and V_{SS}. For the recommended parts and the oscillation characteristics, refer to the "Basic External Connection Diagram" chapter and "OSC1A oscillator circuit characteristics" in the "Electrical Characteristics" chapter, respectively.

OSC1B oscillator circuit

The OSC1B oscillator circuit generates about 32 kHz clock without external components.

EXOSC clock input

EXOSC is an external clock input circuit that supports square wave and sine wave clocks. Figure 2.3.3.4 shows the configuration of the EXOSC clock input circuit.

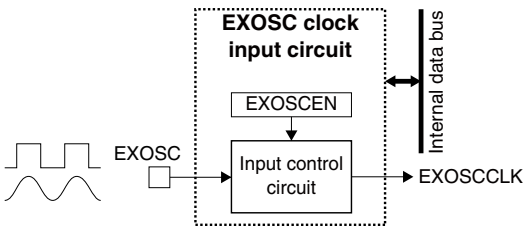


Figure 2.3.3.4 EXOSC Clock Input Circuit

EXOSC has no oscillation stabilization waiting circuit included, therefore, it must be enabled when a stabilized clock is being supplied. For the input clock characteristics, refer to "EXOSC external clock input characteristics" in the "Electrical Characteristics" chapter.

2.3.4 Operations

Oscillation start time and oscillation stabilization waiting time

The oscillation start time refers to the time after the oscillator circuit is enabled until the oscillation signal is actually sent to the internal circuits. The oscillation stabilization waiting time refers to the time it takes the clock to stabilize after the oscillation starts. To avoid malfunctions of the internal circuits due to an unstable clock during this period, the oscillator circuit includes an oscillation stabilization waiting circuit that can disable supplying the clock to the system until the designated time has elapsed. Figure 2.3.4.1 shows the relationship between the oscillation start time and the oscillation stabilization waiting time.

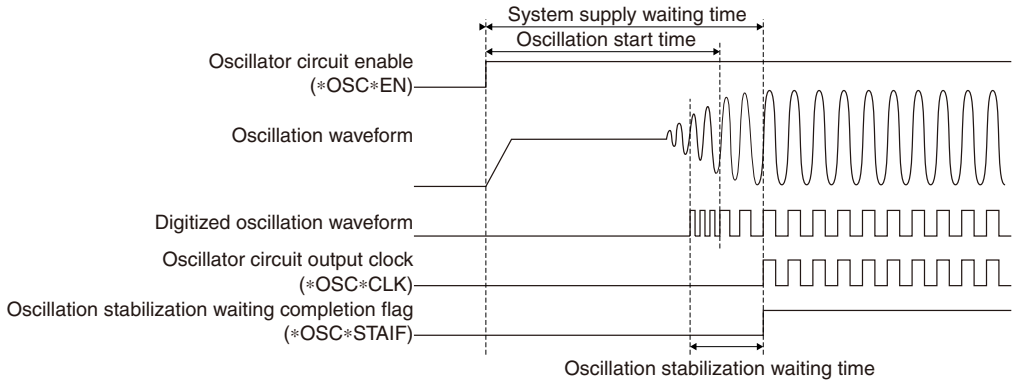


Figure 2.3.4.1 Oscillation Start Time and Oscillation Stabilization Waiting Time

The oscillation stabilization waiting times for the OSC1 (OSC1A and OSC1B) and OSC3A oscillator circuits can be set using the CLGOSC1.OSC1WT[1:0] and CLGOSC3A.OSC3AWT[1:0] bits, respectively. Allow an ample margin for the setting value according to the resonator type used. To check whether the oscillation stabilization waiting time is set properly and the clock is stabilized immediately after the oscillation starts or not, monitor the oscillation clock using the FOUT output function. The oscillation stabilization waiting time for the OSC3B oscillator circuit is fixed at 128 OSC3BCLK clocks.

When the oscillation stabilization waiting operation has completed, the oscillator circuit sets the oscillation stabilization waiting completion flag and starts clock supply to the internal circuits.

Note: The oscillation stabilization waiting time is always expended at start of oscillation even if the oscillation stabilization waiting completion flag has not be cleared to 0.

Oscillation start procedure for the OSC3B oscillator circuit

Follow the procedure shown below to start oscillation of the OSC3B oscillator circuit.

1. Write 1 to the CLGINTF.OSC3BSTAIF bit. (Clear interrupt flag)
2. Write 1 to the CLGINTE.OSC3BSTAIE bit. (Enable interrupt)
3. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the CLGOSC3B.OSC3BFREQ bit. (Select frequency)
5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
6. Write 1 to the CLGOSC.OSC3BEN bit. (Start oscillation)
7. OSC3BCLK can be used if the CLGINTF.OSC3BSTAIF bit = 1 after an interrupt occurs.

Oscillation start procedure for the OSC3A oscillator circuit

Follow the procedure shown below to start oscillation of the OSC3A oscillator circuit.

1. Write 1 to the CLGINTF.OSC3ASTAIF bit. (Clear interrupt flag)
2. Write 1 to the CLGINTE.OSC3ASTAIE bit. (Enable interrupt)
3. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGOSC3A register bits according to the resonator used.
   - CLGOSC3A.INVN[1:0] bits (Set oscillation inverter gain)
   - CLGOSC3A.OSC3AWT[1:0] bits (Set oscillation stabilization waiting time)

5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
6. Write 1 to the CLGOSC.OSC3AEN bit. (Start oscillation)
7. OSC3ACLK can be used if the CLGINTF.OSC3ASTAIF bit = 1 after an interrupt occurs.

The setting values of the CLGOSC3A.INVN[1:0] and CLGOSC3A.OSC3AWT[1:0] bits should be determined after performing evaluation using the populated circuit board.

Oscillation start procedure for the OSC1B oscillator circuit

Follow the procedure shown below to start oscillation of the OSC1B oscillator circuit.

1. Write 1 to the CLGINTF.OSC1STAIF bit. (Clear interrupt flag)
2. Write 1 to the CLGINTE.OSC1STAIE bit. (Enable interrupt)
3. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGOSC1 register bits:
   - CLGOSC1.OSC1WT[1:0] bits (Set oscillation stabilization waiting time)
   - Set the CLGOSC1.OSC1SEL bit to 1. (Select OSC1B)
5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
6. Write 1 to the CLGOSC.OSC1EN bit. (Start oscillation)
7. OSC1CLK can be used if the CLGINTF.OSC1STAIF bit = 1 after an interrupt occurs.

Oscillation start procedure for the OSC1A oscillator circuit

Follow the procedure shown below to start oscillation of the OSC1A oscillator circuit.

1. Write 1 to the CLGINTF.OSC1STAIF bit. (Clear interrupt flag)
2. Write 1 to the CLGINTE.OSC1STAIE bit. (Enable interrupt)
3. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGOSC1 register bits according to the resonator used:
   - CLGOSC1.OSC1WT[1:0] bits (Set oscillation stabilization waiting time)
   - Set the CLGOSC1.OSC1SEL bit to 0. (Select OSC1A)
5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
6. Write 1 to the CLGOSC.OSC1EN bit. (Start oscillation)
7. OSC1CLK can be used if the CLGINTF.OSC1STAIF bit = 1 after an interrupt occurs.

The setting value of the CLGOSC1.OSC1WT[1:0] bits should be determined after performing evaluation using the populated circuit board.

System clock switching

The CPU boots using OSC3BCLK as SYSCLK. After booting, the clock source of SYSCLK can be switched according to the processing speed required. The SYSCLK frequency can also be set by selecting the clock source division ratio, this makes it possible to run the CPU at the most suitable performance for the process to be executed. The CLGSCLK.CLKSRC[1:0] and CLGSCLK.CLKDIV[1:0] bits are used for this control. The CLGSCLK register bits are protected against writings by the system protect function, therefore, the system protection must be removed by writing 0x0096 to the MSCPROT.PROT[15:0] bits before the register setting can be altered. For the transition between the operating modes including the system clock switching, refer to "Operating Mode."

Clock control in SLEEP mode

The CPU enters SLEEP mode when it executes the slp instruction. Whether the clock sources being operated are stopped or not at this point can be selected in each source individually. This allows the CPU to fast switch between SLEEP mode and RUN mode, and the peripheral circuits to continue operating without disabling the clock in SLEEP mode. The CLGOSC.OSC3BSLPC, CLGOSC.OSC1SLPC, CLGOSC.OSC3ASLPC, and CLGOSC.EXOSCSLPC bits are used for this control. Figure 2.3.4.2 shows a control example.

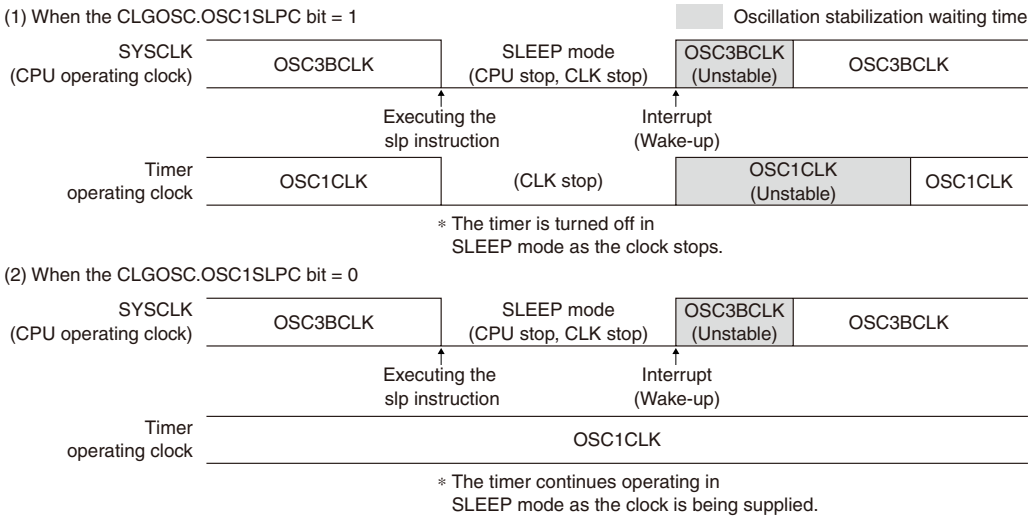


Figure 2.3.4.2 Clock Control Example in SLEEP Mode

The SYSCLK condition (clock source and division ratio) at wake-up from SLEEP mode to RUN mode can also be configured. This allows flexible clock control according to the wake-up process. Configure the clock using the CLGSCLK.WUPSRC[1:0] and CLGSCLK.WUPDIV[1:0] bits, and write 1 to the CLGSCLK.WUPMD bit to enable this function.

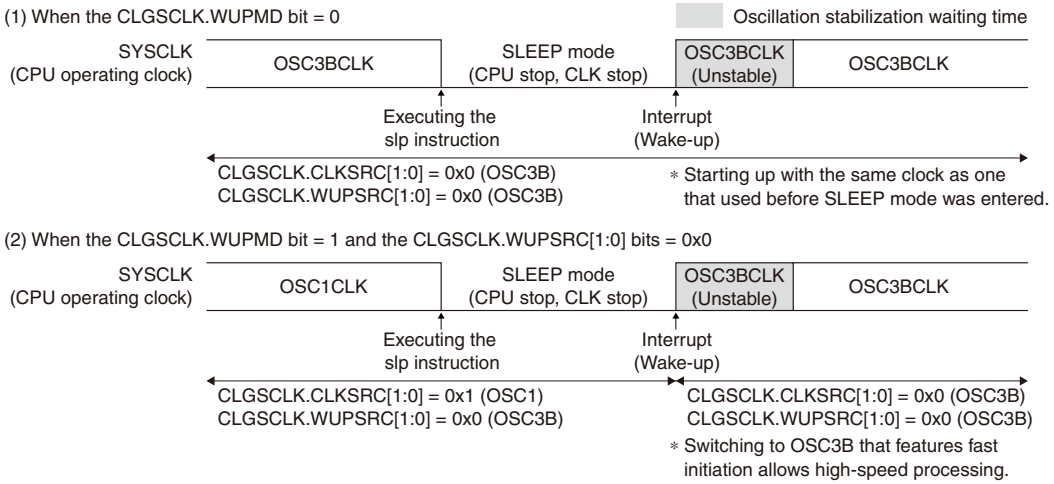


Figure 2.3.4.3 Clock Control Example at SLEEP Cancelation

Note: When OSC1 (OSC1A or OSC1B) is configured to stop in SLEEP mode (CLGOSC.OSC1SLPC bit = 1), executing the slp instruction while a timer is running with OSC1 as the clock source will destabilize the timer operation at restarting from SLEEP mode. When switching to SLEEP mode with CLGOSC.OSC1SLPC bit set to 1, stop the timer before executing the slp instruction. It is not necessary to stop the timer when OSC1 is configured to operate in SLEEP mode.

Clock external output (FOUT)

The FOUT pin can output the clock generated by a clock source or its divided clock to outside the IC. This allows monitoring the oscillation frequency of the oscillator circuit or supplying an operating clock to external ICs. Follow the procedure shown below to start clock external output.

1. Assign the FOUT function to the port. (Refer to the "I/O Ports" chapter.)
2. Configure the following CLGFOUT register bits:
   - CLGFOUT.FOUTSRC[1:0] bits (Select clock source)
   - CLGFOUT.FOUTDIV[2:0] bits (Set clock division ratio)
   - Set the CLGFOUT.FOUTEN bit to 1. (Enable clock external output)

2.4 Operating Mode

2.4.1 Initial Boot Sequence

Figure 2.4.1.1 shows the initial boot sequence after power is turned on.

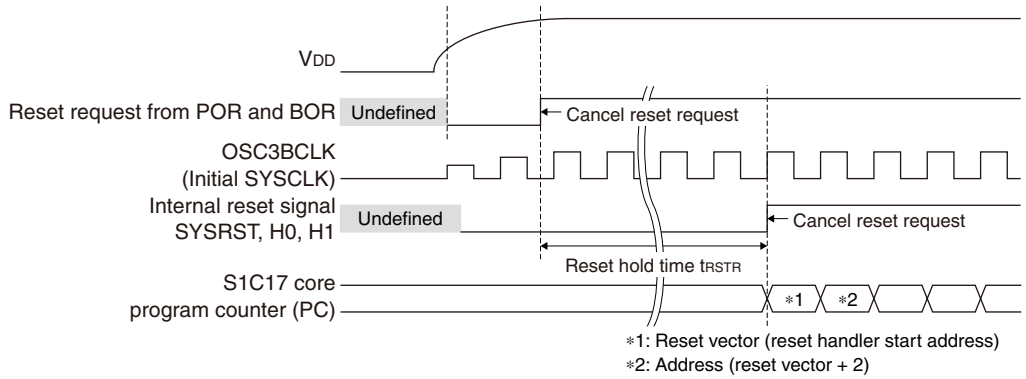


Figure 2.4.1.1 Initial Boot Sequence

Note: The reset cancelation time at power-on varies according to the power rise time and reset request cancelation time.

For the reset hold time t_{RSTR}, refer to "Reset hold circuit characteristics" in the "Electrical Characteristics" chapter.

2.4.2 Transition between Operating Modes

State transitions between operating modes shown in Figure 2.4.2.1 take place in this IC.

RUN mode

RUN mode refers to the state in which the CPU is executing the program. A transition to this mode takes place when the system reset request from the system reset controller is canceled. RUN mode is classified into "OSC3B RUN," "OSC1 RUN," "OSC3A RUN," and "EXOSC RUN" by the SYSCLK clock source.

HALT mode

When the CPU executes the halt instruction, it suspends program execution and stops operating. This state is HALT mode. In this mode, the clock sources and peripheral circuits keep operating. This mode can be set while no software processing is required and it reduces power consumption as compared with RUN mode. HALT mode is classified into "OSC3B HALT," "OSC1 HALT," "OSC3A HALT," and "EXOSC HALT" by the SYSCLK clock source.

SLEEP mode

When the CPU executes the slp instruction, it suspends program execution and stops operating. This state is SLEEP mode. In this mode, the clock sources stop operating as well. However, the clock source in which the CLGOSC.OSC3BSLPC/OSC1SLPC/OSC3ASLPC/EXOSCSLPC bit is set to 0 keeps operating, so the peripheral circuits with the clock being supplied can also operate. By setting this mode when no software processing and peripheral circuit operations are required, power consumption can be less than HALT mode.

Note: The current consumption when a clock source is active in SLEEP mode by setting the CLGOSC.OSC3BSLPC/OSC1SLPC/OSC3ASLPC/EXOSCSLPC bit to 0 is equivalent to the value in HALT mode with the same clock source condition (refer to "Current Consumption, Current consumption in HALT mode I_{HALT1}– I_{HALT4}" in the "Electrical Characteristics" chapter).

DEBUG mode

When a debug interrupt occurs, the CPU enters DEBUG mode. DEBUG mode is canceled when the retd instruction is executed. For more information on DEBUG mode, refer to "Debugger" in the "CPU and Debugger" chapter.

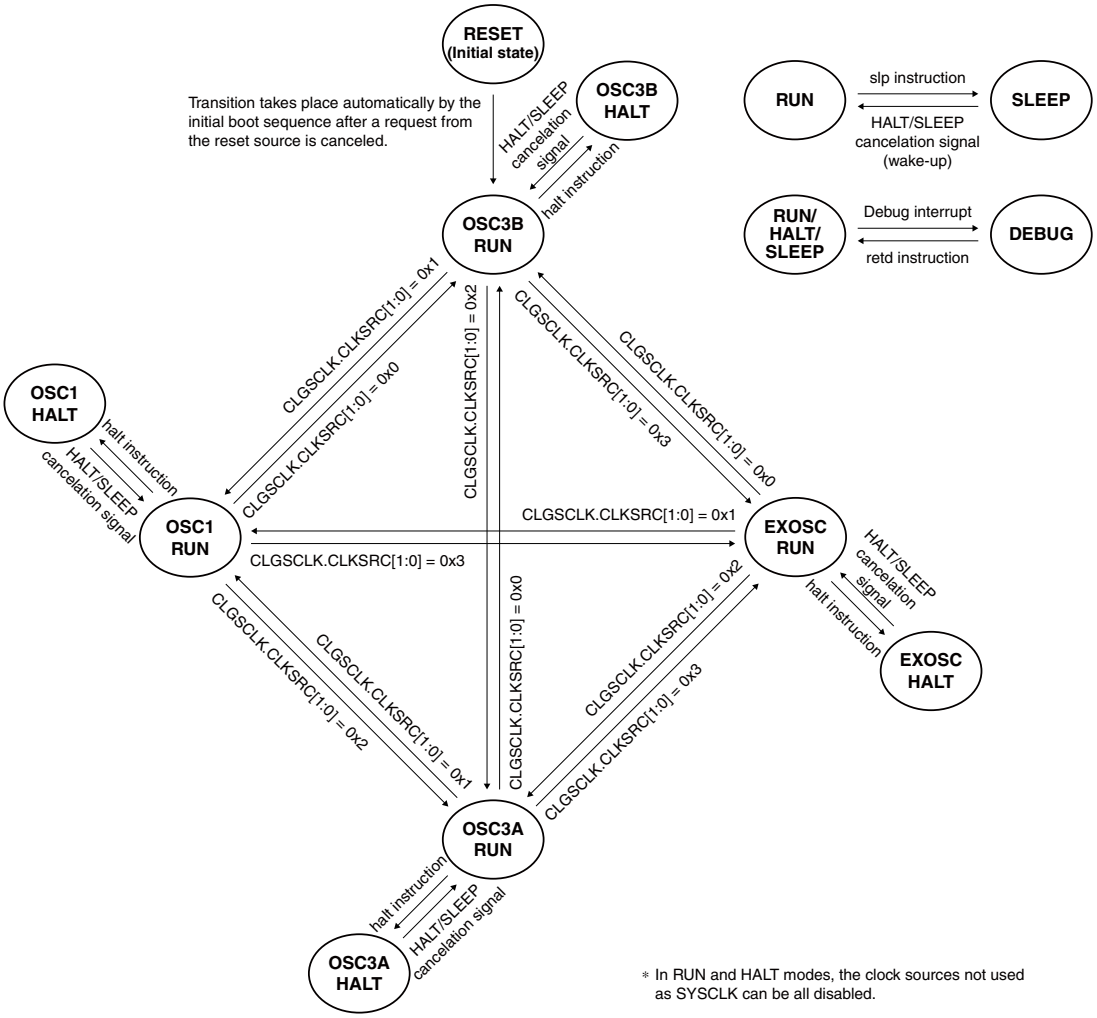


Figure 2.4.2.1 Operating Mode-to-Mode State Transition Diagram

Canceling HALT or SLEEP mode

The conditions listed below generate the HALT/SLEEP cancelation signal to cancel HALT or SLEEP mode and put the CPU into RUN mode. This transition is executed even if the CPU does not accept the interrupt request.

- Interrupt request from a peripheral circuit
- NMI
- Debug interrupt
- Reset request

2.5 Interrupts

CLG has a function to generate the interrupts shown in Table 2.5.1.

Table 2.5.1 CLG Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
OSC3B oscillation stabilization waiting completion	CLGINTF.OSC3BSTAIF	When the OSC3B oscillation stabilization waiting operation has completed after the oscillation starts	Writing 1
OSC1 oscillation stabilization waiting completion	CLGINTF.OSC1STAIF	When the OSC1 oscillation stabilization waiting operation has completed after the oscillation starts	Writing 1
OSC3A oscillation stabilization waiting completion	CLGINTF.OSC3ASTAIF	When the OSC3A oscillation stabilization waiting operation has completed after the oscillation starts	Writing 1

CLG provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

2.6 Control Registers

PWG V_{D1} Regulator Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PWGVD1CTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1–0	REGMODE[1:0]	0x0	H0	R/WP	

Bits 15–2 Reserved

Bits 1–0 REGMODE[1:0]

These bits control the internal regulator operating mode.

Table 2.6.1 Internal Regulator Operating Mode

PWGVD1CTL.REGMODE[1:0] bits	Operating mode
0x3	Economy mode
0x2	Normal mode
0x1	Reserved
0x0	Automatic mode

CLG System Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGSCLK	15	WUPMD	0	H0	R/WP	–
	14	–	0	–	R	
	13–12	WUPDIV[1:0]	0x0	H0	R/WP	
	11–10	–	0x0	–	R	
	9–8	WUPSRC[1:0]	0x0	H0	R/WP	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/WP	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/WP	

Bit 15 WUPMD

This bit enables the SYSCLK switching function at wake-up.
1 (R/WP): Enable
0 (R/WP): Disable

When the CLGSCLK.WUPMD bit = 1, setting values of the CLGSCLK.WUPSRC[1:0] bits and the CLGSCLK.WUPDIV[1:0] bits are loaded to the CLGSCLK.CLKSRC[1:0] bits and the CLGSCLK.CLKDIV[1:0] bits, respectively, at wake-up from SLEEP mode to switch SYSCLK. When the CLGSCLK.WUPMD bit = 0, the CLGSCLK.CLKSRC[1:0] and CLGSCLK.CLKDIV[1:0] bits are not altered at wake-up.

Notes:
- When the CLGSCLK.WUPMD bit = 1, the clock source enable bits (CLGOSC.EXOSCEN, CLGOSC.OSC3AEN, CLGOSC.OSC1EN, CLGOSC.OSC3BEN) except for the SYSCLK source selected by the CLGSCLK.CLKSRC[1:0] bits will be cleared to 0 to stop the clocks after a system wake-up. However, the enable bit of the clock source being operated during SLEEP mode by setting the CLGOSC.****SLPC bit retains 1 after a wake-up.
- When the CLKSCLK.WUPMD bit = 1, be sure to avoid setting both the CLGSCLK.WUPSRC[1:0] bits and the CLGSCLK.WUPDIV[1:0] bits to the same values as the CLGSCLK.CLKSRC[1:0] bits and the CLGSCLK.CLKDIV[1:0] bits, respectively. If the same clock source and division ratio as those that are configured before placing the IC into SLEEP mode are used at wake-up, set the CLKSCLK.WUPMD bit to 0.

Bit 14 Reserved

Bits 13–12 WUPDIV[1:0]

These bits select the SYSCLK division ratio for resetting the CLGSCLK.CLKDIV[1:0] bits at wake-up. This setting is ineffective when the CLGSCLK.WUPMD bit = 0.

Bits 11–10 Reserved

Bits 9–8 WUPSRC[1:0]

These bits select the SYSCLK clock source for resetting the CLGSCLK.CLKSRC[1:0] bits at wake-up. However, this setting is ineffective when the CLGSCLK.WUPMD bit = 0.

Note: Do not select a clock source that has stopped. When selecting it, set the clock source enable bit to 1 before executing the slp instruction.

Table 2.6.2 SYSCLK Clock Source and Division Ratio Settings at Wake-up

CLGSCLK. WUPDIV[1:0] bits	CLGSCLK.WUPSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3BCLK	–	OSC3ACLK	EXOSCCLK
0x3	Reserved	Reserved	1/8	Reserved
0x2	Reserved	Reserved	1/4	Reserved
0x1	1/2	Reserved	1/2	Reserved
0x0	1/1	Reserved	1/1	1/1

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits set the division ratio of the clock source to determine the SYSCLK frequency.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the SYSCLK clock source.
When a currently stopped clock source is selected, it will automatically start oscillating or clock input.

Table 2.6.3 SYSCLK Clock Source and Division Ratio Settings

CLGSCLK. CLKDIV[1:0] bits	CLGSCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3BCLK	OSC1CLK	OSC3ACLK	EXOSCCLK
0x3	Reserved	Reserved	1/8	Reserved
0x2	Reserved	Reserved	1/4	Reserved
0x1	1/2	1/2	1/2	Reserved
0x0	1/1	1/1	1/1	1/1

CLG Oscillation Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGOSC	15–12	–	0x0	–	R	–
	11	EXOSCSLPC	1	H0	R/W	
	10	OSC3ASLPC	1	H0	R/W	
	9	OSC1SLPC	1	H0	R/W	
	8	OSC3BSLPC	1	H0	R/W	
	7–4	–	0x0	–	R	
	3	EXOSCEN	0	H0	R/W	
	2	OSC3AEN	0	H0	R/W	
	1	OSC1EN	0	H0	R/W	
	0	OSC3BEN	1	H0	R/W	

Bits 15–12 Reserved

Bit 11 EXOSCSLPC
Bit 10 OSC3ASLPC
Bit 9 OSC1SLPC
Bit 8 OSC3BSLPC

These bits control the clock source operations in SLEEP mode.
1 (R/W): Stop clock source in SLEEP mode
0 (R/W): Continue operation state before SLEEP

Each bit corresponds to the clock source as follows:
CLGOSC.EXOSCSLPC bit: EXOSC clock input
CLGOSC.OSC3ASLPC bit: OSC3A oscillator circuit
CLGOSC.OSC1SLPC bit: OSC1 oscillator circuit
CLGOSC.OSC3BSLPC bit: OSC3B oscillator circuit

Bits 7–4 Reserved

Bit 3 EXOSCEN
Bit 2 OSC3AEN
Bit 1 OSC1EN
Bit 0 OSC3BEN

These bits control the clock source operation.
1(R/W): Start oscillating or clock input
0(R/W): Stop oscillating or clock input

Each bit corresponds to the clock source as follows:
CLGOSC.EXOSCEN bit: EXOSC clock input
CLGOSC.OSC3AEN bit: OSC3A oscillator circuit
CLGOSC.OSC1EN bit: OSC1 oscillator circuit
CLGOSC.OSC3BEN bit: OSC3B oscillator circuit

CLG OSC3B Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGOSC3B	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1–0	OSC3BFREQ[1:0]	0x0	H0	R/WP	

Bits 15–2 Reserved

Bits 1–0 OSC3BFREQ[1:0]

These bits select the OSC3BCLK frequency.

Table 2.6.4 OSC3BCLK Frequency Selection

CLGOSC3B.OSC3BFREQ[1:0] bits	OSC3BCLK frequency MHz
0x3	20
0x2	16
0x1	12
0x0	8

CLG OSC1 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGOSC1	15–8	–	0x1a	–	R	–
	7–4	–	0xc	–	R	
	3	–	0	–	R	
	2	OSC1SEL	1	H0	R/WP	
	1–0	OSC1WT[1:0]	0x2	H0	R/WP	

Bits 15–3 Reserved

Bit 2 OSC1SEL

This bit selects the OSC1 clock source.
1 (R/WP): OSC1B oscillator circuit
0 (R/WP): OSC1A oscillator circuit

Bits 1–0 OSC1WT[1:0]

These bits set the oscillation stabilization waiting time for the OSC1 oscillator circuit.

Table 2.6.5 OSC1 Oscillation Stabilization Waiting Time Setting

CLGOSC1A.OSC1WT[1:0] bits	Oscillation stabilization waiting time	
	OSC1B	OSC1A
0x3	32 clocks	65,536 clocks
0x2	16 clocks	16,384 clocks
0x1	8 clocks	4,096 clocks
0x0	Reserved	Reserved

CLG OSC3A Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGOSC3A	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5–4	INVN[1:0]	0x1	H0	R/WP	
	3–2	–	0x0	–	R	
	1–0	OSC3AWT[1:0]	0x2	H0	R/WP	

Bits 15–6 Reserved

Bits 5–4 INVN[1:0]

These bits set the oscillation inverter gain of the OSC3A oscillator circuit.

Table 2.6.6 OSC3A Oscillation Inverter Gain Setting

CLGOSC3A.INVN[1:0] bits	Inverter gain
0x3	Max.
0x2	↑
0x1	↓
0x0	Min.

Bits 3–2 Reserved

Bits 1–0 OSC3AWT[1:0]

These bits set the oscillation stabilization waiting time for the OSC3A oscillator circuit.

Table 2.6.7 OSC3A Oscillation Stabilization Waiting Time Setting

CLGOSC3A.OSC3AWT[1:0] bits	Oscillation stabilization waiting time
0x3	4,096 clocks
0x2	1,024 clocks
0x1	256 clocks
0x0	Reserved

CLG Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGINTF	15–8	–	0x00	–	R	–
	7–3	–	0x0	–	R	
	2	OSC3ASTAIF	0	H0	R/W	Cleared by writing 1.
	1	OSC1STAIF	0	H0	R/W	
	0	OSC3BSTAIF	0	H0	R/W	

Bits 15–3 Reserved

Bit 2 OSC3ASTAIF
Bit 1 OSC1STAIF
Bit 0 OSC3BSTAIF

These bits indicate the oscillation stabilization waiting completion interrupt cause occurrence status in each clock source.

1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

Each bit corresponds to the clock source as follows:
CLGINTF.OSC3ASTAIF bit: OSC3A oscillator circuit
CLGINTF.OSC1STAIF bit: OSC1 oscillator circuit
CLGINTF.OSC3BSTAIF bit: OSC3B oscillator circuit

Note: The CLGINTF.OSC3BSTAIF bit is 0 after system reset is canceled, but OSC3BCLK has already been stabilized.

CLG Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGINTE	15–8	–	0x00	–	R	–
	7–3	–	0x0	–	R	
	2	OSC3ASTAIE	0	H0	R/W	
	1	OSC1STAIE	0	H0	R/W	
	0	OSC3BSTAIE	0	H0	R/W	

Bits 15–3 Reserved

Bit 2 OSC3ASTAIE
Bit 1 OSC1STAIE
Bit 0 OSC3BSTAIE

These bits enable the oscillation stabilization waiting completion interrupt of each clock source.

1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

Each bit corresponds to the clock source as follows:
CLGINTE.OSC3ASTAIE bit: OSC3A oscillator circuit
CLGINTE.OSC1STAIE bit: OSC1 oscillator circuit
CLGINTE.OSC3BSTAIE bit: OSC3B oscillator circuit

CLG FOUT Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGFOUT	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6–4	FOUTDIV[2:0]	0x0	H0	R/W	
	3–2	FOUTSRC[1:0]	0x0	H0	R/W	
	1	–	0	–	R	
	0	FOUTEN	0	H0	R/W	

Bits 15–7 Reserved

Bits 6–4 FOUTDIV[2:0]

These bits set the FOUT clock division ratio.

Bits 3–2 FOUTSRC[1:0]

These bits select the FOUT clock source.

Table 2.6.8 FOUT Clock Source and Division Ratio Settings

CLGFOUT. FOUTDIV[2:0] bits	CLGFOUT.FOUTSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3BCLK	OSC1CLK	OSC3ACLK	SYSCLK
0x7	1/128	1/32,768	1/128	Reserved
0x6	1/64	1/4,096	1/64	Reserved
0x5	1/32	1/1,024	1/32	Reserved
0x4	1/16	1/256	1/16	Reserved
0x3	1/8	1/8	1/8	Reserved
0x2	1/4	1/4	1/4	Reserved
0x1	1/2	1/2	1/2	Reserved
0x0	1/1	1/1	1/1	1/1

Note: When the CLGFOUT.FOUTSRC[1:0] bits are set to 0x3, the FOUT output will be stopped in SLEEP/HALT mode as SYSCLK is stopped.

Bit 1 Reserved

Bit 0 FOUTEN

This bit controls the FOUT clock external output.

1 (R/W): Enable external output

0 (R/W): Disable external output

Note: Since the FOUT signal generated is out of sync with writings to the CLGFOUT.FOUTEN bit, a glitch may occur when the FOUT output is enabled or disabled.

3 CPU and Debugger

3.1 Overview

This IC incorporates the Seiko Epson original 16-bit CPU core (S1C17) with a debugger. The main features of the CPU core are listed below.

- Seiko Epson original 16-bit RISC processor
  - 24-bit general-purpose registers: 8
  - 24-bit special registers: 2
  - 8-bit special register: 1
  - Up to 16M bytes of memory space (24-bit address)
  - Harvard architecture using separated instruction bus and data bus
- Compact and fast instruction set optimized for development in C language
  - Code length: 16-bit fixed length
  - Number of instructions: 111 basic instructions (184 including variations)
  - Execution cycle: Main instructions are executed in one cycle.
  - Extended immediate instructions: Immediate data can be extended up to 24 bits.
- Supports reset, NMI, address misaligned, debug, and external interrupts.
  - Reads a vector from the vector table and branches to the interrupt handler routine directly.
  - Can generate software interrupts with a vector number specified (all vector numbers specifiable).
- HALT mode (halt instruction) and SLEEP mode (slp instruction) are provided as the standby function.
- Incorporates a debugger with three-wire communication interface to assist in software development.

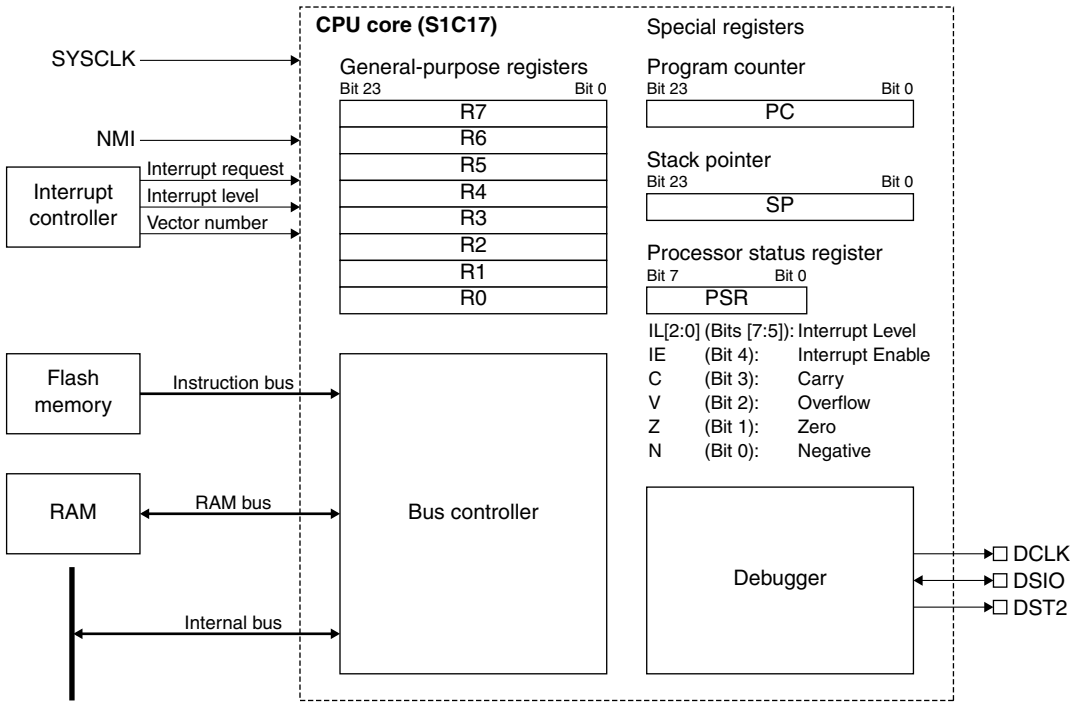


Figure 3.1.1 S1C17 Configuration

3.2 CPU Core

3.2.1 CPU Registers

The CPU includes eight general-purpose registers and three special registers (Table 3.2.1.1).

Table 3.2.1.1 Initialization of CPU Registers

CPU register name			Initial	Reset
General-purpose registers		R0 to R7	0x000000	H0
Special registers	Program counter	PC	The reset vector is automatically loaded.	H0
	Stack pointer	SP	0x000000	H0
	Processor status register	PSR	0x00	H0

For details on the CPU registers, refer to the "S1C17 Family S1C17 Core Manual." For more information on the reset vector, refer to the "Interrupt Controller" chapter.

3.2.2 Instruction Set

The CPU instruction codes are all fixed to 16 bits in length which, combined with pipelined processing, allows the most important instructions to be executed in one cycle. For details on the instructions, refer to the "S1C17 Family S1C17 Core Manual."

3.2.3 Reading PSR

The PSR contents can be read through the MSCPSR register. Note, however, that data cannot be written to PSR through the MSCPSR register.

3.2.4 I/O Area Reserved for the S1C17 Core

The address range from 0xfffc00 to 0xffffff is the I/O area reserved for the S1C17 core. Do not access this area except when it is required.

3.3 Debugger

3.3.1 Debugging Functions

The debugger provides the following functions:

- Instruction break: A debug interrupt is generated immediately before the set instruction address is executed. An instruction break can be set at up to four addresses.
- Single step: A debug interrupt is generated after each instruction has been executed.
- Forcible break: A debug interrupt is generated using an external input signal.
- Software break: A debug interrupt is generated when the brk instruction is executed.

When a debug interrupt occurs, the CPU enters DEBUG mode. The peripheral circuit operations in DEBUG mode depend on the setting of the DBRUN bit provided in the clock control register of each peripheral circuit. For more information on the DBRUN bit, refer to "Clock Supply in DEBUG Mode" in each peripheral circuit chapter. DEBUG mode continues until a cancel command is sent from the personal computer or the CPU executes the retd instruction. Neither hardware interrupts nor NMI are accepted during DEBUG mode.

3.3.2 Resource Requirements and Debugging Tools

Debugging work area

Debugging requires a 64-byte debugging work area. For more information on the work area location, refer to the "Memory and Bus" chapter. The start address of this debugging work area can be read from the DBRAM register.

Debugging tools

To perform debugging, connect ICDmini (S5U1C17001H) to the input/output pin for the debugger embedded in this IC and control it from the personal computer. This requires the tools shown below.

- S1C17 Family In-Circuit Debugger ICDmini (S5U1C17001H)
- S1C17 Family C Compiler Package (e.g., S5U1C17001C)

3.3.3 List of debugger input/output pins

Table 3.3.3.1 lists the debug pins.

Table 3.3.3.1 List of Debug Pins

Pin name	I/O	Initial state	Function
DCLK	O	O	On-chip debugger clock output pin Outputs a clock to the ICDmini (S5U1C17001H).
DSIO	I/O	I	On-chip debugger data input/output pin Used to input/output debugging data and input the break signal.
DST2	O	O	On-chip debugger status output pin Outputs the processor status during debugging.

The debugger input/output pins are shared with general-purpose I/O ports and are initially set as the debug pins. If the debugging function is not used, these pins can be switched to general-purpose I/O port pins. For details, refer to the "I/O Ports" chapter.

3.3.4 External Connection

Figure 3.3.4.1 shows a connection example between this IC and ICDmini when performing debugging.

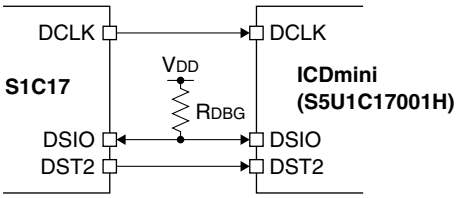


Figure 3.3.4.1 External Connection

For the recommended pull-up resistor value, refer to "Recommended Operating Conditions, DSIO pull-up resistor RDBG" in the "Electrical Characteristics" chapter. RDBG is not required when using the DSIO pin as a general-purpose I/O port pin.

3.4 Control Register

MISC PSR Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MSCPSR	15–8	–	0x00	–	R	–
	7–5	PSRIL[2:0]	0x0	H0	R	
	4	PSRIE	0	H0	R	
	3	PSRC	0	H0	R	
	2	PSRV	0	H0	R	
	1	PSRZ	0	H0	R	
	0	PSRN	0	H0	R	

Bits 15–8 Reserved

Bits 7–5 PSRIL[2:0]

The value (0 to 7) of the PSR IL[2:0] (interrupt level) bits can be read out with these bits.

Bit 4 PSRIE

The value (0 or 1) of the PSR IE (interrupt enable) bit can be read out with this bit.

Bit 3 PSRC

The value (0 or 1) of the PSR C (carry) flag can be read out with this bit.

Bit 2 PSRV

The value (0 or 1) of the PSR V (overflow) flag can be read out with this bit.

Bit 1 PSRZ

The value (0 or 1) of the PSR Z (zero) flag can be read out with this bit.

Bit 0 PSRN

The value (0 or 1) of the PSR N (negative) flag can be read out with this bit.

Debug RAM Base Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DBRAM	31–24	–	0x00	–	R	–
	23–0	DBRAM[23:0]	*1	H0	R	

*1 Debugging work area start address

Bits 31–24 Reserved

Bits 23–0 DBRAM[23:0]

The start address of the debugging work area (64 bytes) can be read out with these bits.

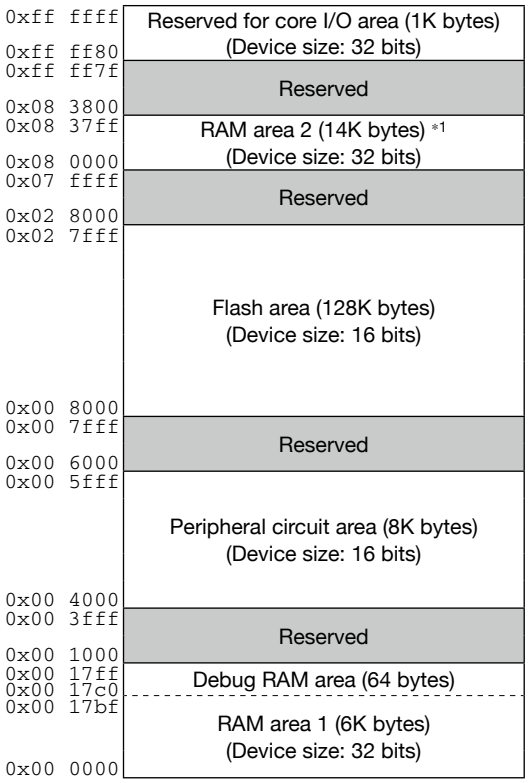
4 Memory and Bus

4.1 Overview

This IC supports up to 16M bytes of accessible memory space for both instructions and data.
The features are listed below.

- Embedded Flash memory that supports on-board programming
- Almost all memory and control registers are accessible in 16-bit width and one cycle. *1
- Write-protect function to protect system control registers

Figure 4.1.1 shows the memory map.



*1 RAM area 2 is a shared area for the CPU and EPD timing controller, and one wait cycle will be inserted to the CPU access cycle (two-cycle access) if this area is accessed from both simultaneously.

Figure 4.1.1 Memory Map

Note: Be sure to avoid data writing operations to the Reserved areas.

4.2 Bus Access Cycle

The CPU uses the system clock for bus access operations. First, "Bus access cycle," "Device size," and "Access size" are defined as follows:

- Bus access cycle: One system clock period = 1 cycle
- Device size: Bit width of the memory and peripheral circuits that can be accessed in one cycle
- Access size: Access size designated by the CPU instructions (e.g., ld %rd, [%rb] → 16-bit data transfer)

Table 4.2.1 lists numbers of bus access cycles by different device size and access size. The peripheral circuits can be accessed with an 8-bit, 16-bit, or 32-bit instruction.

Table 4.2.1 Number of Bus Access Cycles

Device size	Access size	Number of bus access cycles
8 bits	8 bits	1
	16 bits	2
	32 bits	4
16 bits	8 bits	1
	16 bits	1
	32 bits	2
32 bits	8 bits	1
	16 bits	1
	32 bits	1

Note: When data is transferred to a memory in 32-bit access, the eight high-order bits are written to the memory as 0x00 since the bit width of the S1C17 core general-purpose registers is 24 bits. Conversely when sending from a memory to a register, the eight high-order bits are ignored.
The CPU performs 32-bit access for stack operations in an interrupt handling. In this case, the CPU read/write 32-bit data that consists of the PSR value as the eight high-order bits and the return address as the 24 low-order bits. For more information, refer to the "S1C17 Family S1C17 Core Manual."

The CPU adopts Harvard architecture that allows simultaneous processing of an instruction fetch and a data access. However, they are not performed simultaneously under one of the conditions listed below. This prolongs the instruction fetch cycle for the number of data area bus cycles.

- When the CPU executes an instruction stored in the Flash area and accesses data in the Flash area
- When the CPU executes an instruction stored in the internal RAM area and accesses data in the internal RAM area

4.3 Flash Memory

The Flash memory is used to store application programs and data. Address 0x8000 in the Flash area is defined as the vector table base address by default, therefore a vector table must be located beginning from this address. For more information on the vector table, refer to "Vector Table" in the "Interrupt Controller" chapter.

4.3.1 Flash Bus Access Cycle Setting

There is a limit of frequency to access the Flash memory with no wait cycle, therefore, the number of bus access cycles for reading must be changed according to the system clock frequency. The number of bus access cycles for reading can be configured using the FLASHCWAIT.RDWAIT[1:0] bits. Select a setting for higher frequency than the system clock.

4.3.2 Flash Programming

The Flash memory supports on-board programming, so it can be programmed with the ROM data by using the debugger through an ICDmini. Figure 4.3.2.1 shows a connection diagram for on-board programming.

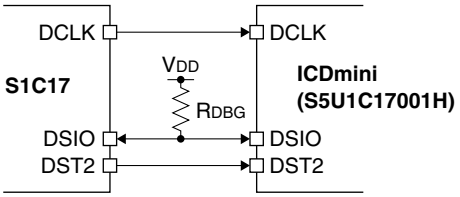


Figure 4.3.2.1 External Connection

For detailed information on ROM data programming method, refer to the "(S1C17 Family C Compiler Package) S5U1C17001C Manual." The IC can also be shipped after being programmed in the factory with the ROM data developed. Should you desire to ship the IC with ROM data programmed from the factory, please contact our customer support.

4.3.3 Flash Security Function

This IC provides a security function to protect the internal Flash memory from unauthorized reading and tampering by using the debugger through ICDmini. Figure 4.3.3.1 shows a Flash security function setting flow.

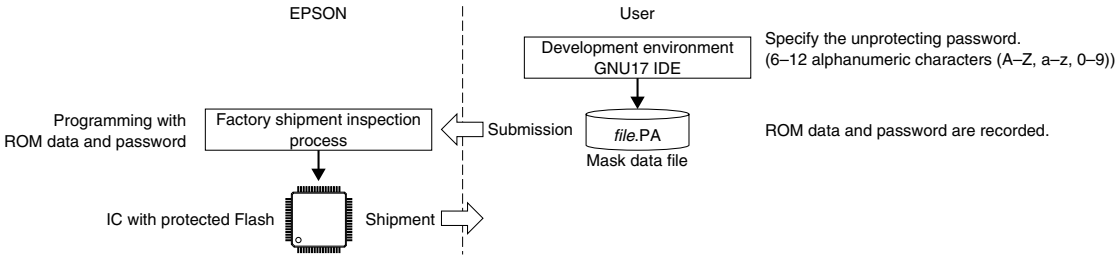


Figure 4.3.3.1 Shipment of IC with ROM Data Programmed and Flash Security Function Setting Flow

The following shows the status of the IC with protected Flash:

- The Flash memory data is undefined if it is read from the debugger.
- An error occurs if an attempt is made to program the Flash memory through ICDmini.

However, the Flash security function can be disabled by entering the unprotecting password predefined to GNU17 IDE (the security function will take effect again after a reset). For setting the password, refer to the "(S1C17 Family C Compiler Package) S5U1C17001C Manual."

Note: Disable the Flash security function before debugging an IC with protected Flash via ICDmini. The debugging functions may not run normally if the Flash security function is enabled.

4.4 RAM1

RAM1 can be used to execute the instruction codes copied from another memory as well as storing variables or other data. This allows higher speed processing and lower power consumption than Flash memory. RAM1 can only be accessed by the CPU.

Note: The 64 bytes at the end of RAM1 is reserved as the debug RAM area. When using the debug functions under application development, do not access this area from the application program. This area can be used for applications of mass-produced devices that do not need debugging.

The RAM1 size used by the application can be configured to equal or less than the implemented size using the MSCIRAMSZ.IRAMSZ[2:0] bits. For example, this function can be used to prevent creating programs that seek to access areas outside the RAM area of the target model when developing an application for a model in which the RAM size is smaller than this IC. After the limitation is applied, accessing an address outside the RAM1 area results in the same operation (undefined value is read out) as when a reserved area is accessed.

4.5 RAM2

The embedded RAM2 is used to store display data for the EPD. RAM2 allows the EPD timing controller to read data as well as accesses from the CPU.
The entire RAM2 area or the area unused for display data can be used as a general-purpose RAM.

4.6 Peripheral Circuit Control Registers

The control registers for the peripheral circuits are located in the 8K-byte area beginning with address 0x4000. Table 4.6.1 shows the control register map. For details of each control register, refer to "List of Peripheral Circuit Registers" in the Appendix or "Control Registers" in each peripheral circuit chapter.

Table 4.6.1 Peripheral Circuit Control Register Map

Peripheral circuit	Address	Register name	
MISC registers (MISC)	0x4000	MSCPROT	MISC System Protect Register
	0x4002	MSCIRAMSZ	MISC IRAM Size Register
	0x4004	MSCTTBRL	MISC Vector Table Address Low Register
	0x4006	MSCTTBRH	MISC Vector Table Address High Register
	0x4008	MSCPSR	MISC PSR Register
Power generator (PWG)	0x4020	PWGVD1CTL	PWG V_{D1} Regulator Control Register
Clock generator (CLG)	0x4040	CLGSCLK	CLG System Clock Control Register
	0x4042	CLGOSC	CLG Oscillation Control Register
	0x4044	CLGOSC3B	CLG OSC3B Control Register
	0x4046	CLGOSC1	CLG OSC1 Control Register
	0x4048	CLGOSC3A	CLG OSC3A Control Register
	0x404a	CLGINTF	CLG Interrupt Flag Register
	0x404c	CLGINTE	CLG Interrupt Enable Register
	0x404e	CLGFOUT	CLG FOUT Control Register
Theoretical regulation (TR)	0x4052	TRCTL	Theoretical Regulation Control Register
Interrupt controller (ITC)	0x4080	ITCLV0	ITC Interrupt Level Setup Register 0
	0x4082	ITCLV1	ITC Interrupt Level Setup Register 1
	0x4084	ITCLV2	ITC Interrupt Level Setup Register 2
	0x4086	ITCLV3	ITC Interrupt Level Setup Register 3
	0x4088	ITCLV4	ITC Interrupt Level Setup Register 4
	0x408a	ITCLV5	ITC Interrupt Level Setup Register 5
	0x408c	ITCLV6	ITC Interrupt Level Setup Register 6
	0x408e	ITCLV7	ITC Interrupt Level Setup Register 7
	0x4090	ITCLV8	ITC Interrupt Level Setup Register 8
	0x4092	ITCLV9	ITC Interrupt Level Setup Register 9
Watchdog timer (WDT)	0x40a0	WDTCLK	WDT Clock Control Register
	0x40a2	WDTCTL	WDT Control Register
Real-time clock (RTC)	0x40c0	RTCCTL	RTC Control Register
	0x40c2	RTCINTE	RTC Interrupt Enable Register
	0x40c4	RTCINTF	RTC Interrupt Flag Register
	0x40c6	RTCMIN	RTC Minute/Second Register
	0x40c8	RTCHUR	RTC Hour Register
Supply voltage detector (SVD)	0x4100	SVDCLK	SVD Clock Control Register
	0x4102	SVDCTL	SVD Control Register
	0x4104	SVDINTF	SVD Status and Interrupt Flag Register
	0x4106	SVDINTE	SVD Interrupt Enable Register
16-bit timer (T16) Ch.0	0x4160	T16_0CLK	T16 Ch.0 Clock Control Register
	0x4162	T16_0MOD	T16 Ch.0 Mode Register
	0x4164	T16_0CTL	T16 Ch.0 Control Register
	0x4166	T16_0TR	T16 Ch.0 Reload Data Register
	0x4168	T16_0TC	T16 Ch.0 Counter Data Register
	0x416a	T16_0INTF	T16 Ch.0 Interrupt Flag Register
	0x416c	T16_0INTE	T16 Ch.0 Interrupt Enable Register
Flash controller (FLASHC)	0x41b0	FLASHCWAIT	FLASHC Flash Read Cycle Register
I/O ports (PPORT)	0x4200	P0DAT	P0 Port Data Register
	0x4202	P0IOEN	P0 Port Enable Register
	0x4204	P0RCTL	P0 Port Pull-up/down Control Register
	0x4206	P0INTF	P0 Port Interrupt Flag Register
	0x4208	P0INTCTL	P0 Port Interrupt Control Register
	0x420a	P0CHATEN	P0 Port Chattering Filter Enable Register
	0x420c	P0MODSEL	P0 Port Mode Select Register
	0x420e	P0FNCSEL	P0 Port Function Select Register
	0x4210	P1DAT	P1 Port Data Register
	0x4212	P1IOEN	P1 Port Enable Register
	0x4214	P1RCTL	P1 Port Pull-up/down Control Register
	0x4216	P1INTF	P1 Port Interrupt Flag Register
	0x4218	P1INTCTL	P1 Port Interrupt Control Register
	0x421a	P1CHATEN	P1 Port Chattering Filter Enable Register
	0x421c	P1MODSEL	P1 Port Mode Select Register

Peripheral circuit	Address	Register name	
I/O ports (PPORT)	0x421e	P1FNCSEL	P1 Port Function Select Register
	0x4220	P2DAT	P2 Port Data Register
	0x4222	P2IOEN	P2 Port Enable Register
	0x4224	P2RCTL	P2 Port Pull-up/down Control Register
	0x422c	P2MODSEL	P2 Port Mode Select Register
	0x422e	P2FNCSEL	P2 Port Function Select Register
	0x4230	P3DAT	P3 Port Data Register
	0x4232	P3IOEN	P3 Port Enable Register
	0x4234	P3RCTL	P3 Port Pull-up/down Control Register
	0x423c	P3MODSEL	P3 Port Mode Select Register
	0x423e	P3FNCSEL	P3 Port Function Select Register
	0x4240	P4DAT	P4 Port Data Register
	0x4242	P4IOEN	P4 Port Enable Register
	0x4244	P4RCTL	P4 Port Pull-up/down Control Register
	0x424c	P4MODSEL	P4 Port Mode Select Register
	0x424e	P4FNCSEL	P4 Port Function Select Register
	0x42d0	PDDAT	Pd Port Data Register
	0x42d2	PDIOEN	Pd Port Enable Register
	0x42d4	PDRCTL	Pd Port Pull-up/down Control Register
	0x42dc	PDMODSEL	Pd Port Mode Select Register
	0x42de	PDFNCSEL	Pd Port Function Select Register
	0x42e0	PCLK	P Port Clock Control Register
	0x42e2	PINTFGRP	P Port Interrupt Flag Group Register
UART (UART)	0x4380	UA0CLK	UART Ch.0 Clock Control Register
	0x4382	UA0MOD	UART Ch.0 Mode Register
	0x4384	UA0BR	UART Ch.0 Baud-Rate Register
	0x4386	UA0CTL	UART Ch.0 Control Register
	0x4388	UA0TXD	UART Ch.0 Transmit Data Register
	0x438a	UA0RXD	UART Ch.0 Receive Data Register
	0x438c	UA0INTF	UART Ch.0 Status and Interrupt Flag Register
	0x438e	UA0INTE	UART Ch.0 Interrupt Enable Register
16-bit timer (T16) Ch.1	0x43a0	T16_1CLK	T16 Ch.1 Clock Control Register
	0x43a2	T16_1MOD	T16 Ch.1 Mode Register
	0x43a4	T16_1CTL	T16 Ch.1 Control Register
	0x43a6	T16_1TR	T16 Ch.1 Reload Data Register
	0x43a8	T16_1TC	T16 Ch.1 Counter Data Register
	0x43aa	T16_1INTF	T16 Ch.1 Interrupt Flag Register
	0x43ac	T16_1INTE	T16 Ch.1 Interrupt Enable Register
SPI (SPI) Ch.0	0x43b0	SPI0MOD	SPI Ch.0 Mode Register
	0x43b2	SPI0CTL	SPI Ch.0 Control Register
	0x43b4	SPI0TXD	SPI Ch.0 Transmit Data Register
	0x43b6	SPI0RXD	SPI Ch.0 Receive Data Register
	0x43b8	SPI0INTF	SPI Ch.0 Interrupt Flag Register
	0x43ba	SPI0INTE	SPI Ch.0 Interrupt Enable Register
I^2C (I2C)	0x43c0	I2C0CLK	I2C Ch.0 Clock Control Register
	0x43c2	I2C0MOD	I2C Ch.0 Mode Register
	0x43c4	I2C0BR	I2C Ch.0 Baud-Rate Register
	0x43c8	I2C0OADR	I2C Ch.0 Own Address Register
	0x43ca	I2C0CTL	I2C Ch.0 Control Register
	0x43cc	I2C0TXD	I2C Ch.0 Transmit Data Register
	0x43ce	I2C0RXD	I2C Ch.0 Receive Data Register
	0x43d0	I2C0INTF	I2C Ch.0 Status and Interrupt Flag Register
	0x43d2	I2C0INTE	I2C Ch.0 Interrupt Enable Register
16-bit PWM timer (T16A3) Ch.0	0x5000	T16A0CLK	T16A3 Ch.0 Clock Control Register
	0x5002	T16A0CTL	T16A3 Counter Ch.0 Control Register
	0x5004	T16A0TC	T16A3 Counter Ch.0 Data Register
	0x5006	T16A0CCCTL	T16A3 Comparator/Capture Ch.0 Control Register
	0x5008	T16A0CCA	T16A3 Comparator/Capture Ch.0 A Data Register
	0x500a	T16A0CCB	T16A3 Comparator/Capture Ch.0 B Data Register
	0x500c	T16A0INTF	T16A3 Ch.0 Interrupt Flag Register

Peripheral circuit	Address	Register name	
16-bit PWM timer (T16A3) Ch.0	0x500e	T16A0INTE	T16A3 Ch.0 Interrupt Enable Register
16-bit PWM timer (T16A3) Ch.1	0x5020	T16A1CLK	T16A3 Ch.1 Clock Control Register
	0x5022	T16A1CTL	T16A3 Counter Ch.1 Control Register
	0x5024	T16A1TC	T16A3 Counter Ch.1 Data Register
	0x5026	T16A1CCCTL	T16A3 Comparator/Capture Ch.1 Control Register
	0x5028	T16A1CCA	T16A3 Comparator/Capture Ch.1 A Data Register
	0x502a	T16A1CCB	T16A3 Comparator/Capture Ch.1 B Data Register
	0x502c	T16A1INTF	T16A3 Ch.1 Interrupt Flag Register
	0x502e	T16A1INTE	T16A3 Ch.1 Interrupt Enable Register
Clock timer (CT)	0x5180	CTCTL	CT Control Register
	0x5182	CTDAT	CT Counter Data Register
	0x5184	CTINTF	CT Interrupt Flag Register
	0x5186	CTINTE	CT Interrupt Enable Register
16-bit timer (T16) Ch.2	0x5260	T16_2CLK	T16 Ch.2 Clock Control Register
	0x5262	T16_2MOD	T16 Ch.2 Mode Register
	0x5264	T16_2CTL	T16 Ch.2 Control Register
	0x5266	T16_2TR	T16 Ch.2 Reload Data Register
	0x5268	T16_2TC	T16 Ch.2 Counter Data Register
	0x526a	T16_2INTF	T16 Ch.2 Interrupt Flag Register
	0x526c	T16_2INTE	T16 Ch.2 Interrupt Enable Register
SPI (SPI) Ch.1	0x5270	SPI1MOD	SPI Ch.1 Mode Register
	0x5272	SPI1CTL	SPI Ch.1 Control Register
	0x5274	SPI1TXD	SPI Ch.1 Transmit Data Register
	0x5276	SPI1RXD	SPI Ch.1 Receive Data Register
	0x5278	SPI1INTF	SPI Ch.1 Interrupt Flag Register
	0x527a	SPI1INTE	SPI Ch.1 Interrupt Enable Register
16-bit timer (T16) Ch.3	0x5280	T16_3CLK	T16 Ch.3 Clock Control Register
	0x5282	T16_3MOD	T16 Ch.3 Mode Register
	0x5284	T16_3CTL	T16 Ch.3 Control Register
	0x5286	T16_3TR	T16 Ch.3 Reload Data Register
	0x5288	T16_3TC	T16 Ch.3 Counter Data Register
	0x528a	T16_3INTF	T16 Ch.3 Interrupt Flag Register
	0x528c	T16_3INTE	T16 Ch.3 Interrupt Enable Register
SPI (SPI) Ch.2	0x5290	SPI2MOD	SPI Ch.2 Mode Register
	0x5292	SPI2CTL	SPI Ch.2 Control Register
	0x5294	SPI2TXD	SPI Ch.2 Transmit Data Register
	0x5296	SPI2RXD	SPI Ch.2 Receive Data Register
	0x5298	SPI2INTF	SPI Ch.2 Interrupt Flag Register
	0x529a	SPI2INTE	SPI Ch.2 Interrupt Enable Register
Parallel interface (PIO)	0x52e0	PIOCLK	PIO Clock Control Register
	0x52e2	PIOMOD	PIO Mode Register
	0x52e4	PIOCTL	PIO Control Register
	0x52e6	PIOWRDAT	PIO Address/Write Data Register
	0x52e8	PIORDDAT	PIO Read Data Register
	0x52ea	PIOSTAT	PIO Status Register
EPD timing controller (EPD Tcon)	0x5380	EPDCTL	EPD Tcon Control Register
	0x5382	EPDINTF	EPD Tcon Interrupt Flag and Status Register
	0x5384	EPDINTE	EPD Tcon Interrupt Enable Register
R/F converter (RFC) Ch.0	0x5440	RFC0CLK	RFC Ch.0 Clock Control Register
	0x5442	RFC0CTL	RFC Ch.0 Control Register
	0x5444	RFC0TRG	RFC Ch.0 Oscillation Trigger Register
	0x5446	RFC0MCL	RFC Ch.0 Measurement Counter Low Register
	0x5448	RFC0MCH	RFC Ch.0 Measurement Counter High Register
	0x544a	RFC0TCL	RFC Ch.0 Time Base Counter Low Register
	0x544c	RFC0TCH	RFC Ch.0 Time Base Counter High Register
	0x544e	RFC0INTF	RFC Ch.0 Interrupt Flag Register
	0x5450	RFC0INTE	RFC Ch.0 Interrupt Enable Register
R/F converter (RFC) Ch.1	0x5460	RFC1CLK	RFC Ch.1 Clock Control Register
	0x5462	RFC1CTL	RFC Ch.1 Control Register
	0x5464	RFC1TRG	RFC Ch.1 Oscillation Trigger Register

Peripheral circuit	Address	Register name	
R/F converter (RFC) Ch.1	0x5466	RFC1MCL	RFC Ch.1 Measurement Counter Low Register
	0x5468	RFC1MCH	RFC Ch.1 Measurement Counter High Register
	0x546a	RFC1TCL	RFC Ch.1 Time Base Counter Low Register
	0x546c	RFC1TCH	RFC Ch.1 Time Base Counter High Register
	0x546e	RFC1INTF	RFC Ch.1 Interrupt Flag Register
	0x5470	RFC1INTE	RFC Ch.1 Interrupt Enable Register
Temperature detection circuit (TEM)	0x54c0	TEMCLK	TEM Clock Control Register
	0x54c2	TEMTMG	TEM Timing Register
	0x54c4	TEMCTL	TEM Control Register
	0x54c6	TEMRSLT	TEM Conversion Result Register
	0x54c8	TEMINTF	TEM Interrupt Flag and Status Register
	0x54ca	TEMINTE	TEM Interrupt Enable Register

4.6.1 System-Protect Function

The system-protect function protects control registers and bits from writings. They cannot be rewritten unless write protection is removed by writing 0x0096 to the MSCPROT.PROT[15:0] bits. This function is provided to prevent deadlock that may occur when a system-related register is altered by a runaway CPU. See "Control Registers" in each peripheral circuit to identify the registers and bits with write protection.

Note: Once write protection is removed using the MSCPROT.PROT[15:0] bits, write enabled status is maintained until write protection is applied again. After the registers/bits required have been altered, apply write protection.

4.7 Control Registers

MISC System Protect Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MSCPROT	15–0	PROT[15:0]	0x0000	H0	R/W	–

Bits 15–0 PROT[15:0]

These bits protect the control registers related to the system against writings.
0x0096 (R/W): Disable system protection
Other than 0x0096 (R/W): Enable system protection

While the system protection is enabled, any data will not be written to the affected control bits (bits with "WP" or "R/WP" appearing in the R/W column).

MISC IRAM Size Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MSCIRAMSZ	15–9	–	0x00	–	R	–
	8	(reserved)	0	H0	R/WP	Always set to 0.
	7	–	0	–	R	–
	6–4	(reserved)	0x4	–	R	
	3	–	0	–	R	
	2–0	IRAMSZ[2:0]	0x4	H0	R/WP	

Bits 15–3 Reserved

Bits 2–0 IRAMSZ[2:0]

These bits set the internal RAM size that can be used.

Table 4.7.1 Internal RAM Size Selections

MSCIRAMSZ.IRAMSZ[2:0] bits	Internal RAM size
0x7	Reserved
0x6	(16KB)*
0x5	(12KB)*
0x4	6KB
0x3	4KB
0x2	2KB
0x1	1KB
0x0	512B

* Setting prohibited in this IC

FLASHC Flash Read Cycle Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
FLASHCWAIT	15–8	–	0x00	–	R	–
	7	XBUSY	0	H0	R	
	6–2	–	0x00	–	R	
	1–0	RDWAIT[1:0]	0x0	H0	R/WP	

Bits 15–8 Reserved

Bit 7 XBUSY

This bit indicates whether the Flash memory can be accessed or not.

1 (R): Flash memory ready to access

0 (R): Flash access prohibited

The Flash memory can always be accessed during normal operation.

Bits 6–2 Reserved

Bits 1–0 RDWAIT[1:0]

These bits set the number of bus access cycles for reading from the Flash memory.

Table 4.7.2 Setting Number of Bus Access Cycles for Flash Read

FLASHCWAIT.RDWAIT[1:0] bits	Number of bus access cycles	System clock frequency
0x3	4	20.0 MHz (max.)
0x2	3	20.0 MHz (max.)
0x1	2	16.3 MHz (max.)
0x0	1	8.2 MHz (max.)

Note: Be sure to set the FLASHCWAIT.RDWAIT[1:0] bits before the system clock is configured.

5 Interrupt Controller (ITC)

5.1 Overview

The features of the ITC are listed below.

- Honors interrupt requests from the peripheral circuits and outputs the interrupt request, interrupt level and vector number signals to the CPU.
- The interrupt level of each interrupt source is selectable from among eight levels.
- Priorities of the simultaneously generated interrupts are established from the interrupt level.
- Handles the simultaneously generated interrupts with the same interrupt level as smaller vector number has higher priority.

Figure 5.1.1 shows the configuration of the ITC.

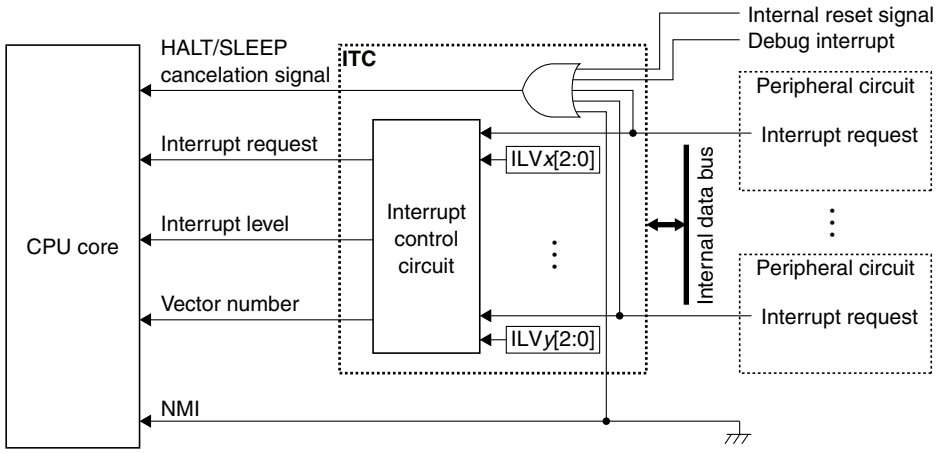


Figure 5.1.1 ITC Configuration

5.2 Vector Table

The vector table contains the vectors to the interrupt handler routines (handler routine start address) that will be read by the CPU to execute the handler when an interrupt occurs.
Table 5.2.1 shows the vector table.

Table 5.2.1 Vector Table

TTBR initial value = 0x8000

Vector number/ Software interrupt number	Vector address	Hardware interrupt name	Cause of hardware interrupt	Priority
0 (0x00)	TTBR + 0x00	Reset	• Low input to the #RESET pin • Power-on reset • Brownout reset • Key entry reset • Watchdog timer overflow • Supply voltage detector reset	1
1 (0x01)	TTBR + 0x04	Address misaligned interrupt	Memory access instruction	2
–	(0xfffc00)	Debugging interrupt	brk instruction, etc.	3
2 (0x02)	TTBR + 0x08	NMI	–	4
3 (0x03)	TTBR + 0x0c	Reserved for C compiler	–	–

Vector number/ Software interrupt number	Vector address	Hardware interrupt name	Hardware interrupt flag	Priority
4 (0x04)	TTBR + 0x10	Supply voltage detector interrupt	Low power supply voltage detection	High *1 ↑
5 (0x05)	TTBR + 0x14	Port interrupt	Port input	
6 (0x06)	TTBR + 0x18	Clock generator interrupt	• OSC3A oscillation stabilization waiting completion • OSC1 oscillation stabilization waiting completion • OSC3B oscillation stabilization waiting completion	
7 (0x07)	TTBR + 0x1c	Real-time clock interrupt	• 1 day • Half day • 1 hour • 10 minutes • 1 minute • 10 seconds • 1 Hz • 4 Hz • 8 Hz • 32 Hz	
8 (0x08)	TTBR + 0x20	16-bit timer Ch.0 interrupt	Underflow	
9 (0x09)	TTBR + 0x24	UART interrupt	• End of transmission • Framing error • Parity error • Overrun error • Receive buffer two bytes full • Receive buffer one byte full • Transmit buffer empty	
10 (0x0a)	TTBR + 0x28	16-bit timer Ch.1 interrupt	Underflow	
11 (0x0b)	TTBR + 0x2c	SPI Ch.0 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty	
12 (0x0c)	TTBR + 0x30	I^2C interrupt	• End of data transfer • General call address reception • NACK reception • STOP condition • START condition • Error detection • Receive buffer full • Transmit buffer empty	
13 (0x0d)	TTBR + 0x34	Clock timer interrupt	• 32 Hz • 8 Hz • 2 Hz • 1 Hz	
14 (0x0e)	TTBR + 0x38	16-bit timer Ch.2 interrupt	Underflow	
15 (0x0f)	TTBR + 0x3c	SPI Ch.1 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty	
16 (0x10)	TTBR + 0x40	16-bit timer Ch.3 interrupt	Underflow	
17 (0x11)	TTBR + 0x44	SPI Ch.2 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty	
18 (0x12)	TTBR + 0x48	16-bit PWM timer Ch.0 interrupt	• Capture B overwrite • Capture A overwrite • Capture B • Capture A • Compare B • Compare A	
19 (0x13)	TTBR + 0x4c	16-bit PWM timer Ch.1 interrupt	• Capture B overwrite • Capture A overwrite • Capture B • Capture A • Compare B • Compare A	
20 (0x14)	TTBR + 0x50	R/F Ch.0 converter interrupt	• Reference oscillation completion • Sensor A oscillation completion • Sensor B oscillation completion • Measurement counter overflow error • Time base counter overflow error	

Vector number/ Software interrupt number	Vector address	Hardware interrupt name	Hardware interrupt flag	Priority
21 (0x15)	TTBR + 0x54	R/F Ch.1 converter interrupt	• Reference oscillation completion • Sensor A oscillation completion • Sensor B oscillation completion • Measurement counter overflow error • Time base counter overflow error	
22 (0x16)	TTBR + 0x58	EPD timing controller interrupt	Display refresh completion	
23 (0x17)	TTBR + 0x5c	Temperature detection circuit interrupt	Conversion completion	
24 (0x18) : 31 (0x1f)	TTBR + 0x60 : TTBR + 0x7c	reserved : reserved	– : –	↓ Low *1

*1 When the same interrupt level is set

5.2.1 Vector Table Base Address (TTBR)

The MSCTTBRL and MSCTTBRH registers are provided to set the base (start) address of the vector table in which interrupt vectors are programmed. "TTBR" described in Table 5.2.1 means the value set to these registers. After an initial reset, the MSCTTBRL and MSCTTBRH registers are set to address 0x8000. Therefore, even when the vector table location is changed, it is necessary that at least the reset vector be written to the above address. Bits 7 to 0 in the MSCTTBRL register are fixed at 0, so the vector table always begins from a 256-byte boundary address.

5.3 Initialization

The following shows an example of the initial setting procedure related to interrupts:

1. Execute the di instruction to set the CPU into interrupt disabled state.
2. If the vector table start address is different from the default address, set it to the MSCTTBRL and MSCTTBRH registers after removing system protection by writing 0x0096 to the MSCPROT.PROT[15:0] bits. Then, write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits to set system protection.
3. Set the interrupt enable bit of the peripheral circuit to 0 (interrupt disabled).
4. Set the interrupt level for the peripheral circuit using the ITCLV*x*.ILV*x*[2:0] bits in the ITC.
5. Configure the peripheral circuit and start its operation.
6. Clear the interrupt factor flag of the peripheral circuit.
7. Set the interrupt enable bit of the peripheral circuit to 1 (interrupt enabled).
8. Execute the ei instruction to set the CPU into interrupt enabled state.

5.4 Maskable Interrupt Control and Operations

5.4.1 Peripheral Circuit Interrupt Control

The peripheral circuit that generates interrupts includes an interrupt enable bit and an interrupt flag for each interrupt cause.

Interrupt flag: The flag is set to 1 when the interrupt cause occurs. The clear condition depends on the peripheral circuit.

Interrupt enable bit: By setting this bit to 1 (interrupt enabled), an interrupt request will be sent to the ITC when the interrupt flag is set to 1. When this bit is set to 0 (interrupt disabled), no interrupt request will be sent to the ITC even if the interrupt flag is set to 1. An interrupt request is also sent to the ITC if the status is changed to interrupt enabled when the interrupt flag is 1.

For specific information on causes of interrupts, interrupt flags, and interrupt enable bits, refer to the respective peripheral circuit descriptions.

Note: To prevent occurrence of unnecessary interrupts, the corresponding interrupt flag should be cleared before setting the interrupt enable bit to 1 (interrupt enabled) and before terminating the interrupt handler routine.

5.4.2 ITC Interrupt Request Processing

On receiving an interrupt signal from a peripheral circuit, the ITC sends an interrupt request, the interrupt level, and the vector number to the CPU. Vector numbers are determined by the ITC internal hardware for each interrupt cause, as shown in Table 5.2.1. The interrupt level is a value to configure the priority, and it can be set to between 0 (low) and 7 (high) using the ITCLV*x*.ILV*x*[2:0] bits provided for each interrupt source. The default ITC settings are level 0 for all maskable interrupts. Interrupt requests are not accepted by the CPU if the level is 0.
The ITC outputs the interrupt request with the highest priority to the CPU in accordance with the following conditions if interrupt requests are input to the ITC simultaneously from two or more peripheral circuits.

- The interrupt with the highest interrupt level takes precedence.
- If multiple interrupt requests are input with the same interrupt level, the interrupt with the lowest vector number takes precedence.

The other interrupts occurring at the same time are held until all interrupts with higher priority levels have been accepted by the CPU.
If an interrupt cause with higher priority occurs while the ITC is outputting an interrupt request signal to the CPU (before being accepted by the CPU), the ITC alters the vector number and interrupt level signals to the setting information on the more recent interrupt. The previously occurring interrupt is held. The held interrupt is canceled and no interrupt is generated if the interrupt flag in the peripheral circuit is cleared via software.

Note: Before changing the interrupt level, make sure that no interrupt of which the level is changed can be generated (the interrupt enable bit of the peripheral circuit is set to 0 or the peripheral circuit is deactivated).

5.4.3 Conditions to Accept Interrupt Requests by the CPU

The CPU accepts an interrupt request sent from the ITC when all of the following conditions are met:

- The IE (Interrupt Enable) bit of the PSR has been set to 1.
- The interrupt request that has occurred has a higher interrupt level than the value set in the IL[2:0] (Interrupt Level) bits of the PSR.
- No other interrupt request having higher priority, such as NMI, has occurred.

5.5 NMI

This IC cannot generate non-maskable interrupts (NMI).

5.6 Software Interrupts

The CPU provides the “int *imm5*” and “intl *imm5*, *imm3*” instructions allowing the software to generate any interrupts. The operand *imm5* specifies a vector number (0–31) in the vector table. In addition to this, the intl instruction has the operand *imm3* to specify the interrupt level (0–7) to be set to the IL[2:0] bits in the PSR. The software interrupt cannot be disabled (non-maskable interrupt). The processor performs the same interrupt processing operation as that of the hardware interrupt.

5.7 Interrupt Processing by the CPU

The CPU samples interrupt requests for each cycle. On accepting an interrupt request, the CPU switches to interrupt processing immediately after execution of the current instruction has been completed.
Interrupt processing involves the following steps:

1. The PSR and current program counter (PC) values are saved to the stack.
2. The PSR IE bit is cleared to 0 (disabling subsequent maskable interrupts).
3. The PSR IL[2:0] bits are set to the received interrupt level. (The NMI does not affect the IL bits.)
4. The vector for the interrupt occurred is loaded to the PC to execute the interrupt handler routine.

When an interrupt is accepted, Step 2 prevents subsequent maskable interrupts. Setting the IE bit to 1 in the interrupt handler routine allows handling of multiple interrupts. In this case, since the IL[2:0] bits are changed by Step 3, only an interrupt with a higher level than that of the currently processed interrupt will be accepted.
Ending interrupt handler routines using the reti instruction returns the PSR to the state before the interrupt occurred. The program resumes processing following the instruction being executed at the time the interrupt occurred.

Note: When HALT or SLEEP mode is canceled, the CPU jumps to the interrupt handler routine after executing one instruction. To execute the interrupt handler routine immediately after HALT or SLEEP mode is canceled, place the nop instruction at just behind the halt/slp instruction.

5.8 Control Registers

MISC Vector Table Address Low Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MSCTTBRL	15–8	TTBR[15:8]	0x80	H0	R/WP	–
	7–0	TTBR[7:0]	0x00	H0	R	

Bits 15–0 TTBR[15:0]
These bits set the vector table base address (16 low-order bits).

MISC Vector Table Address High Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MSCTTBRH	15–8	–	0x00	–	R	–
	7–0	TTBR[23:16]	0x00	H0	R/WP	

Bits 15–8 Reserved

Bits 7–0 TTBR[23:16]
These bits set the vector table base address (eight high-order bits).

ITC Interrupt Level Setup Register *x*

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ITCLV*x*	15–11	–	0x00	–	R	–
	10–8	ILVy_1[2:0]	0x0	H0	R/W	
	7–3	–	0x00	–	R	
	2–0	ILVy_0[2:0]	0x0	H0	R/W	

Bits 15–11 Reserved
Bits 7–3 Reserved

Bits 10–8 ILVy_1[2:0] ($y_1 = 2x + 1$)
Bits 2–0 ILVy_0[2:0] ($y_0 = 2x$)
These bits set the interrupt level of each interrupt.

Table 5.8.1 Interrupt Level and Priority Settings

ITCLV*x*.ILV*y*[2:0] bits	Interrupt level	Priority
0x7	7	High
0x6	6	↑
. . .	. . .	
0x1	1	↓
0x0	0	Low

The following shows the ITCLV*x* register configuration in this IC.

Table 5.8.2 List of ITCLV*x* Registers

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ITCLV0 (ITC Interrupt Level Setup Register 0)	15–11	–	0x00	–	R	–
	10–8	ILV1[2:0]	0x0	H0	R/W	Port interrupt (ILVPPORT)
	7–3	–	0x00	–	R	–
	2–0	ILV0[2:0]	0x0	H0	R/W	Supply voltage detector interrupt (ILVSVD)
ITCLV1 (ITC Interrupt Level Setup Register 1)	15–11	–	0x00	–	R	–
	10–8	ILV3[2:0]	0x0	H0	R/W	Real-time clock interrupt (ILVRTC)
	7–3	–	0x00	–	R	–
	2–0	ILV2[2:0]	0x0	H0	R/W	Clock generator interrupt (ILVCLG)
ITCLV2 (ITC Interrupt Level Setup Register 2)	15–11	–	0x00	–	R	–
	10–8	ILV5[2:0]	0x0	H0	R/W	UART interrupt (ILVUART_0)
	7–3	–	0x00	–	R	–
	2–0	ILV4[2:0]	0x0	H0	R/W	16-bit timer Ch.0 interrupt (ILVT16_0)
ITCLV3 (ITC Interrupt Level Setup Register 3)	15–11	–	0x00	–	R	–
	10–8	ILV7[2:0]	0x0	H0	R/W	SPI Ch.0 interrupt (ILVSPI_0)
	7–3	–	0x00	–	R	–
	2–0	ILV6[2:0]	0x0	H0	R/W	16-bit timer Ch.1 interrupt (ILVT16_1)
ITCLV4 (ITC Interrupt Level Setup Register 4)	15–11	–	0x00	–	R	–
	10–8	ILV9[2:0]	0x0	H0	R/W	Clock timer interrupt (ILVCT)
	7–3	–	0x00	–	R	–
	2–0	ILV8[2:0]	0x0	H0	R/W	I^2C interrupt (ILVI2C_0)
ITCLV5 (ITC Interrupt Level Setup Register 5)	15–11	–	0x00	–	R	–
	10–8	ILV11[2:0]	0x0	H0	R/W	SPI Ch.1 interrupt (ILVSPI_1)
	7–3	–	0x00	–	R	–
	2–0	ILV10[2:0]	0x0	H0	R/W	16-bit timer Ch.2 interrupt (ILVT16_2)
ITCLV6 (ITC Interrupt Level Setup Register 6)	15–11	–	0x00	–	R	–
	10–8	ILV13[2:0]	0x0	H0	R/W	SPI Ch.2 interrupt (ILVSPI_2)
	7–3	–	0x00	–	R	–
	2–0	ILV12[2:0]	0x0	H0	R/W	16-bit timer Ch.3 interrupt (ILVT16_3)
ITCLV7 (ITC Interrupt Level Setup Register 7)	15–11	–	0x00	–	R	–
	10–8	ILV15[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.1 interrupt (ILVT16A3_1)
	7–3	–	0x00	–	R	–
	2–0	ILV14[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.0 interrupt (ILVT16A3_0)
ITCLV8 (ITC Interrupt Level Setup Register 8)	15–11	–	0x00	–	R	–
	10–8	ILV17[2:0]	0x0	H0	R/W	R/F converter Ch.1 interrupt (ILVRFC_1)
	7–3	–	0x00	–	R	–
	2–0	ILV16[2:0]	0x0	H0	R/W	R/F converter Ch.0 interrupt (ILVRFC_0)
ITCLV9 (ITC Interrupt Level Setup Register 9)	15–11	–	0x00	–	R	–
	10–8	ILV19[2:0]	0x0	H0	R/W	Temperature detection circuit interrupt (ILVTEM)
	7–3	–	0x00	–	R	–
	2–0	ILV18[2:0]	0x0	H0	R/W	EPD timing controller interrupt (ILVEPD_Tcon)

6 I/O Ports (PPORT)

6.1 Overview

PPORT controls the I/O ports. The main features are outlined below.

- Allows port-by-port function configurations.
  - Each port can be configured with or without a pull-up or pull-down resistor.
  - Each port can be configured with or without a chattering filter.
  - Allows selection of the function (general-purpose I/O port (GPIO) function, up to four peripheral I/O functions) to be assigned to each port.
- Ports, except for those shared with debug pins, are initially placed into Hi-Z state.
  (No current passes through the pin during this Hi-Z state.)
- Over voltage tolerant fail-safe design allowing interface with the signal without passing unnecessary current even if a voltage exceeding VDD is applied.

Note: '*x*', which is used in the port names P*xy*, register names, and bit names, refers to a port group (*x* = 0, 1, 2, ⋯ , d) and '*y*' refers to a port number (*y* = 0, 1, 2, ⋯ , 7).

Figure 6.1.1 shows the configuration of PPORT.

Port configuration in this IC	
• Port groups included:	P0[7:0], P1[7:0], P2[7:0], P3[7:0], P4[1:0], Pd[2:0]
• Ports with general-purpose I/O function (GPIO):	P0[7:0], P1[7:0], P2[7:0], P3[7:0], P4[1:0], Pd[2:0] (Pd2: output only)
• Ports with interrupt function:	P0[7:0], P1[7:0]
• Ports for debug function:	Pd[2:0]
• Key-entry reset function:	Supported (P0[3:0])

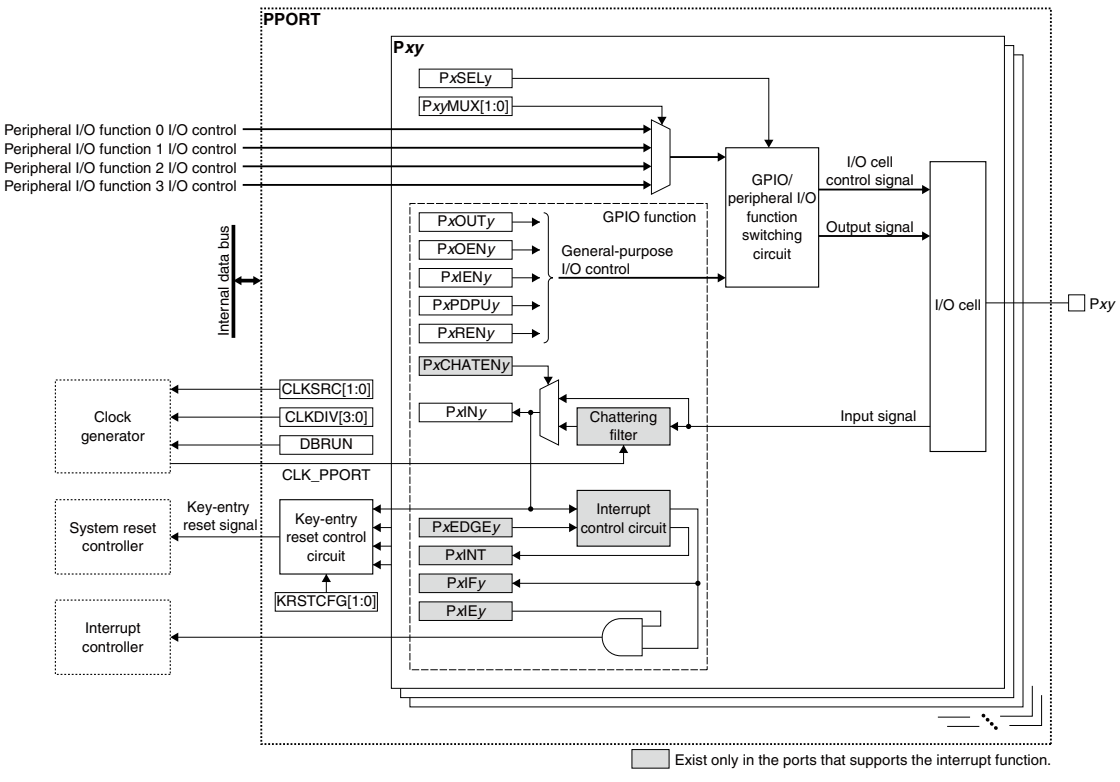


Figure 6.1.1 PPORT Configuration

6.2 I/O Cell Structure and Functions

Figure 6.2.1 shows the I/O cell Configuration.

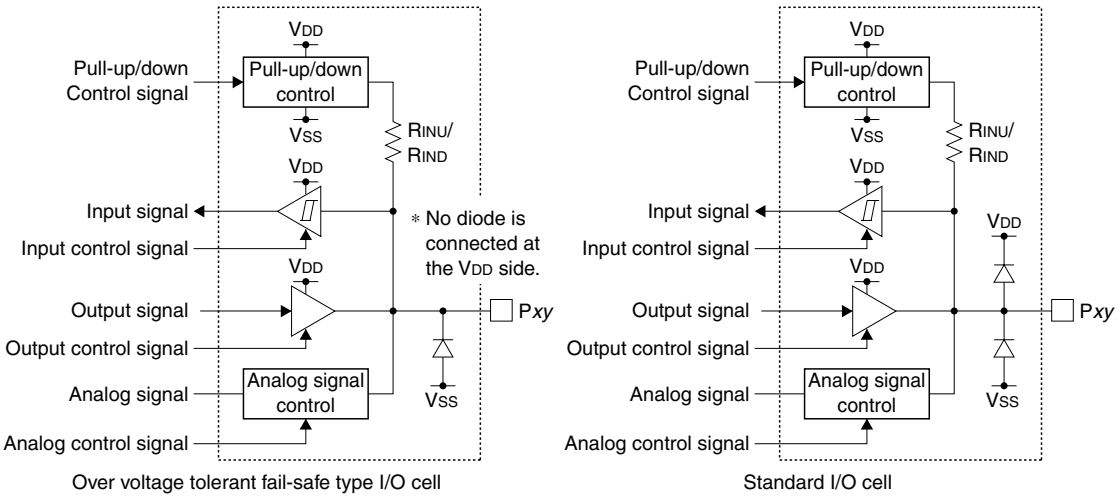


Figure 6.2.1 I/O Cell Configuration

Refer to "Pin Descriptions" in the "Overview" chapter for the cell type, either the over voltage tolerant fail-safe type I/O cell or the standard I/O cell, included in each port.

6.2.1 Schmitt Input

The input functions are all configured with the Schmitt interface level. When a port is set to input disable status (P*x*IOEN.P*x*IEN*y* bit = 0), unnecessary current is not consumed if the P*xy* pin is placed into floating status.

6.2.2 Over Voltage Tolerant Fail-Safe Type I/O Cell

The over voltage tolerant fail-safe type I/O cell allows interfacing without passing unnecessary current even if a voltage exceeding V_{DD} is applied to the port. Also unnecessary current is consumed when the port is externally biased without supplying V_{DD}. However, be sure to avoid applying a voltage exceeding the recommended maximum operating power supply voltage to the port.

6.2.3 Pull-Up/Pull-Down

The GPIO port has a pull-up/pull-down function. Either pull-up or pull-down may be selected for each port individually. This function may also be disabled for the port that does not require pulling up/down.
When the port level is switched from low to high through the pull-up resistor included in the I/O cell or from high to low through the pull-down resistor, a delay will occur in the waveform rising/falling edge depending on the time constant by the pull-up/pull-down resistance and the pin load capacitance. The rising/falling time is commonly determined by the following equation:

$$t_{PR} = -R_{INU} \times (C_{IN} + C_{BOARD}) \times \ln(1 - V_{T+}/V_{DD}) \quad \text{(Eq. 6.1)}$$
$$t_{PF} = -R_{IND} \times (C_{IN} + C_{BOARD}) \times \ln(1 - V_{T-}/V_{DD})$$

Where

- t_{PR}: Rising time (port level = low → high) [second]
- t_{PF}: Falling time (port level = high → low) [second]
- V_{T+}: High level Schmitt input threshold voltage [V]
- V_{T-}: Low level Schmitt input threshold voltage [V]
- R_{INU}/R_{IND}: Pull-up/pull-down resistance [Ω]
- C_{IN}: Pin capacitance [F]
- C_{BOARD}: Parasitic capacitance on the board [F]

6.2.4 CMOS Output and High Impedance State

The I/O cells except for analog output can output signals in the VDD and VSS levels. Also the GPIO ports may be put into high-impedance (Hi-Z) state.

6.3 Clock Settings

6.3.1 PPORT Operating Clock

When using the chattering filter for entering external signals to PPORT, the PPORT operating clock CLK_PPORT must be supplied to PPORT from the clock generator.
The CLK_PPORT supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
3. Set the following PCLK register bits:
   - PCLK.CLKSRC[1:0] bits (Clock source selection)
   - PCLK.CLKDIV[3:0] bits (Clock division ratio selection = Clock frequency setting)
4. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Settings in Step 3 determine the input sampling time of the chattering filter.

6.3.2 Clock Supply in SLEEP Mode

When using the chattering filter function during SLEEP mode, the PPORT operating clock CLK_PPORT must be configured so that it will keep suppling by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_PPORT clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_PPORT clock source is 1, the CLK_PPORT clock source is deactivated during SLEEP mode and it disables the chattering filter function regardless of the P*x*CHATEN.P*x*CHATEN*y* bit setting (chattering filter enabled/disabled).

6.3.3 Clock Supply in DEBUG Mode

The CLK_PPORT supply during DEBUG mode should be controlled using the PCLK.DBRUN bit.
The CLK_PPORT supply to PPORT is suspended when the CPU enters DEBUG mode if the PCLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_PPORT supply resumes. The PPORT chattering filter stops operating when the CLK_PPORT supply is suspended. If the chattering filter is enabled in PPORT, the input port function is also deactivated. However, the control registers can be altered. If the PCLK.DBRUN bit = 1, the CLK_PPORT supply is not suspended and the chattering filter will keep operating in DEBUG mode.

6.4 Operations

6.4.1 Initialization

After a reset, the ports except for the debugging function are configured as shown below.

- Port input: Disabled
- Port output: Disabled
- Pull-up: Off
- Pull-down: Off
- Port pins: High impedance state
- Port function: Configured to GPIO

This status continues until the ports are configured via software. The debugging function ports are configured for debug signal input/output.

Initial settings when using a port for a peripheral I/O function

When using the P*xy* port for a peripheral I/O function, perform the following software initial settings:

1. Set the following P*x*IOEN register bits:
   Set the P*x*IOEN.P*x*IEN*y* bit to 0. (Disable input)
   Set the P*x*IOEN.P*x*OEN*y* bit to 0. (Disable output)
2. Set the P*x*MODSEL.P*x*SEL*y* bit to 0. (Disable peripheral I/O function)
3. Initialize the peripheral circuit that uses the pin.
4. Set the P*x*FNCSEL.P*xy*MUX[1:0] bits. (Select peripheral I/O function)
5. Set the P*x*MODSEL.P*x*SEL*y* bit to 1. (Enable peripheral I/O function)

For the list of the peripheral I/O functions that can be assigned to each port of this IC, refer to "Control Register and Port Function Configuration of this IC." For the specific information on the peripheral I/O functions, refer to the respective peripheral circuit chapter.

Initial settings when using a port as a general-purpose output port (only for the ports with GPIO function)

When using the P*xy* port pin as a general-purpose output pin, perform the following software initial settings:

1. Set the P*x*IOEN.P*x*OEN*y* bit to 1. (Enable output)
2. Set the P*x*MODSEL.P*x*SEL*y* bit to 0. (Enable GPIO function)

Initial settings when using a port as a general-purpose input port (only for the ports with GPIO function)

When using the P*xy* port pin as a general-purpose input pin, perform the following software initial settings:

1. Write 0 to the P*x*INTCTL.P*x*IE*y* bit. * (Disable interrupt)
2. When using the chattering filter, configure the PPORT operating clock (see "PPORT Operating Clock") and set the P*x*CHATEN.P*x*CHATEN*y* bit to 1. *

   When the chattering filter is not used, set the P*x*CHATEN.P*x*CHATEN*y* bit to 0 (supply of the PPORT operating clock is not required).
3. Configure the following P*x*RCTL register bits when pulling up/down the port using the internal pull-up or down resistor:
   - P*x*RCTL.P*x*PDPU*y* bit (Select pull-up or pull-down resistor)
   - Set the P*x*RCTL.P*x*REN*y* bit to 1. (Enable pull-up/down)

   Set the P*x*RCTL.P*x*REN*y* bit to 0 if the internal pull-up/down resistors are not used.
4. Set the P*x*MODSEL.P*x*SEL*y* bit to 0. (Enable GPIO function)
5. Configure the following bits when using the port input interrupt: *
   - Write 1 to the P*x*INTF.P*x*IF*y* bit. (Clear interrupt flag)
   - P*x*INTCTL.P*x*EDGE*y* bit (Select interrupt edge (input rising edge/falling edge))
   - Set the P*x*INTCTL.P*x*IE*y* bit to 1. (Enable interrupt)
6. Set the following P*x*IOEN register bits:
   - Set the P*x*IOEN.P*x*OEN*y* bit to 0. (Disable output)
   - Set the P*x*IOEN.P*x*IEN*y* bit to 1. (Enable input)

∗ Steps 1 and 5 are required for the ports with an interrupt function. Step 2 is required for the ports with a chattering filter function.

Table 6.4.1.1 lists the port status according to the combination of data input/output control and pull-up/down control.

Table 6.4.1.1 GPIO Port Control List

PxIOEN. PxIENy bit	PxIOEN. PxOENy bit	PxRCTL. PxRENy bit	PxRCTL. PxPDPUy bit	Input	Output	Pull-up/pull-down condition
0	0	0	×	Disabled		Off (Hi-Z) *1
0	0	1	0	Disabled		Pulled down
0	0	1	1	Disabled		Pulled up
1	0	0	×	Enabled	Disabled	Off (Hi-Z) *2
1	0	1	0	Enabled	Disabled	Pulled down
1	0	1	1	Enabled	Disabled	Pulled up
0	1	0	×	Disabled	Enabled	Off
0	1	1	0	Disabled	Enabled	Off
0	1	1	1	Disabled	Enabled	Off
1	1	1	0	Enabled	Enabled	Off
1	1	1	1	Enabled	Enabled	Off

*1: Initial status. Current does not flow if the pin is placed into floating status.
*2: Use of the pull-up or pull-down function is recommended, as undesired current will flow if the port input is set to floating status.

Note: If the PxMODSEL.PxSELy bit for the port without a GPIO function is set to 0, the port goes into initial status (refer to "Initial Settings"). The GPIO control bits are configured to a read-only bit always read out as 0.

6.4.2 Port Input/Output Control

Peripheral I/O function control

The port for which a peripheral I/O function is selected is controlled by the peripheral circuit. For more information, refer to the respective peripheral circuit chapter.

Setting output data to a GPIO port

Write data (1 = high output, 0 = low output) to be output from the P*xy* pin to the PxDAT.P*x*OUT*y* bit.

Reading input data from a GPIO port

The data (1 = high input, 0 = low input) input from the P*xy* pin can be read out from the PxDAT.P*x*IN*y* bit.

Note: The PxDAT.PxIN*y* bit retains the input port status at 1 clock before being read from the CPU.

Chattering filter function

Some GPIO ports have a chattering filter function and it can be controlled in each port. This function is enabled by setting the PxCHATEN.P*x*CHATEN*y* bit to 1. The input sampling time to remove chattering is determined by the CLK_PPORT frequency configured using the PCLK register in common to all ports. The chattering filter removes pulses with a shorter width than the input sampling time.

$$\text{Input sampling time} = \frac{2 \text{ to } 3}{\text{CLK\_PPORT frequency [Hz]}} \text{ [second]} \qquad \text{(Eq.6.2)}$$

Make sure the P*xy* port interrupt is disabled before altering the PCLK register and P*x*CHATEN.P*x*CHATEN*y* bit settings. A P*xy* port interrupt may erroneously occur if these settings are altered in an interrupt enabled status. Furthermore, enable the interrupt after a lapse of four or more CLK_PPORT cycles from enabling the chattering filter function.
If the clock generator is configured so that it will supply CLK_PPORT to PPORT in SLEEP mode, the chattering filter of the port will function even in SLEEP mode. If CLK_PPORT is configured to stop in SLEEP mode, PPORT inactivates the chattering filter during SLEEP mode to input pin status transitions directly to itself.

Key-entry reset function

This function issues a reset request when low-level pulses are input to all the specified ports simultaneously. Make the following settings when using this function:

1. Configure the ports to be used for key-entry reset as general-purpose input ports (refer to "Initial settings when using a port as a general-purpose input port (only for the ports with GPIO function)").
2. Configure the input pin combination for key-entry reset using the PCLK.KRSTCFG[1:0] bits.

Note: When enabling the key-entry reset function, be sure to configure the port pins to be used for it as general-purpose input pins before setting the PCLK.KRSTCFG[1:0] bits.

PPORT issues a reset request immediately after all the input pins specified by the PCLK.KRSTCFG[1:0] are set to a low level if the chattering filter function is disabled (initial status). To issue a reset request only when low-level signals longer than the time configured are input, enable the chattering filter function for all the ports used for key-entry reset.
The pins configured for key-entry reset can also be used as general-purpose input pins.

6.5 Interrupts

When the GPIO function is selected for the port with an interrupt function, the port input interrupt function can be used.

Table 6.5.1 Port Input Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Port input interrupt	P*x*INTF.P*x*IF*y*	Rising or falling edge of the input signal	Writing 1
	PINTFGRP.P*x*INT	Setting an interrupt flag in the port group	Clearing P*x*INTF.P*x*IF*y*

Interrupt edge selection

Port input interrupts will occur at the falling edge of the input signal when setting the P*x*INTCTL.P*x*EDGE*y* bit to 1, or the rising edge when setting to 0.

Interrupt enable

PPORT provides interrupt enable bits (P*x*INTCTL.P*x*IE*y* bit) corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

Interrupt check in port group unit

When interrupts are enabled in two or more port groups, check the PINTFGRP.P*x*INT bit in the interrupt handler first. It helps minimize the handler codes for finding the port that has generated an interrupt. If this bit is set to 1, an interrupt has occurred in the port group. Next, check the P*x*INTF.P*x*IF*y* bit set to 1 in the port group to determine the port that has generated an interrupt. Clearing the P*x*INTF.P*x*IF*y* bit also clears the PINTFGRP.P*x*INT bit. If the port is set to interrupt disabled status by the P*x*INTCTL.P*x*IE*y* bit, the PINTFGRP.P*x*INT bit will not be set even if the P*x*INTF.P*x*IF*y* bit is set to 1.

6.6 Control Registers

This section describes the same control registers of all port groups as a single register. For the register and bit configurations in each port group and their initial values, refer to "Control Register and Port Function Configuration of this IC."

P*x* Port Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*DAT	15–8	P*x*OUT[7:0]	0x00	H0	R/W	–
	7–0	P*x*IN[7:0]	0x00	H0	R	

∗1: This register is effective when the GPIO function is selected.
∗2: The bit configuration differs depending on the port group.
∗3: The initial value may be changed by the port.

Bits 15–8 P*x*OUT[7:0]

These bits are used to set data to be output from the GPIO port pins.
1 (R/W): Output high level from the port pin
0 (R/W): Output low level from the port pin

When output is enabled (P*x*IOEN.P*x*OEN*y* bit = 1), the port pin outputs the data set here. Although data can be written when output is disabled (P*x*IOEN.P*x*OEN*y* bit = 0), it does not affect the pin status.
These bits do not affect the outputs when the port is used as a peripheral I/O function.

Bits 7–0 P*x*IN[7:0]

The GPIO port pin status can be read out from these bits.
1 (R): Port pin = High level
0 (R): Port pin = Low level

The port pin status can be read out when input is enabled (P*x*IOEN.P*x*IEN*y* bit = 1). When input is disabled (P*x*IOEN.P*x*IEN*y* bit = 0), these bits are always read as 0.
When the port is used for a peripheral I/O function, the input value cannot be read out from these bits.

P*x* Port Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PxIOEN	15–8	PxIEN[7:0]	0x00	H0	R/W	–
	7–0	PxOEN[7:0]	0x00	H0	R/W	

∗1: This register is effective when the GPIO function is selected.
∗2: The bit configuration differs depending on the port group.

Bits 15–8 P*x*IEN[7:0]

These bits enable/disable the GPIO port input.
1 (R/W): Enable (The port pin status is input.)
0 (R/W): Disable (Input data is fixed at 0.)

When both data output and data input are enabled, the pin output status controlled by this IC can be read.
These bits do not affect the input control when the port is used as a peripheral I/O function.

Bits 7–0 P*x*OEN[7:0]

These bits enable/disable the GPIO port output.
1 (R/W): Enable (Data is output from the port pin.)
0 (R/W): Disable (The port is placed into Hi-Z.)

These bits do not affect the output control when the port is used as a peripheral I/O function.

P*x* Port Pull-up/down Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PxRCTL	15–8	PxPDPU[7:0]	0x00	H0	R/W	–
	7–0	PxREN[7:0]	0x00	H0	R/W	

∗1: This register is effective when the GPIO function is selected.
∗2: The bit configuration differs depending on the port group.

Bits 15–8 P*x*PDPU[7:0]

These bits select either the pull-up resistor or the pull-down resistor when using a resistor built into the port.
1 (R/W): Pull-up resistor
0 (R/W): Pull-down resistor

The selected pull-up/down resistor is enabled when the P*x*RCTL.P*x*REN*y* bit = 1.

Bits 7–0 P*x*REN[7:0]

These bits enable/disable the port pull-up/down control.
1 (R/W): Enable (The built-in pull-up/down resistor is used.)
0 (R/W): Disable (No pull-up/down control is performed.)

Enabling this function pulls up or down the port when output is disabled (P*x*IOEN.P*x*OEN*y* bit = 0). When output is enabled (P*x*IOEN.P*x*OEN*y* bit = 1), the P*x*RCTL.P*x*REN*y* bit setting is ineffective regardless of how the P*x*IOEN.P*x*IEN*y* bit is set and the port is not pulled up/down.
These bits do not affect the pull-up/down control when the port is used as a peripheral I/O function.

P*x* Port Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*INTF	15–8	–	0x00	–	R	–
	7–0	P*x*IF[7:0]	0x00	H0	R/W	Cleared by writing 1.

∗1: This register is effective when the GPIO function is selected.
∗2: The bit configuration differs depending on the port group.

Bits 15–8 Reserved

Bits 7–0 P*x*IF[7:0]

These bits indicate the port input interrupt cause occurrence status.

1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

P*x* Port Interrupt Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*INTCTL	15–8	P*x*EDGE[7:0]	0x00	H0	R/W	–
	7–0	P*x*IE[7:0]	0x00	H0	R/W	

∗1: This register is effective when the GPIO function is selected.
∗2: The bit configuration differs depending on the port group.

Bits 15–8 P*x*EDGE[7:0]

These bits select the input signal edge to generate a port input interrupt.

1 (R/W): An interrupt will occur at a falling edge.
0 (R/W): An interrupt will occur at a rising edge.

Bits 7–0 P*x*IE[7:0]

These bits enable port input interrupts.

1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

Note: To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.

P*x* Port Chattering Filter Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*CHATEN	15–8	–	0x00	–	R	–
	7–0	P*x*CHATEN[7:0]	0x00	H0	R/W	

∗1: The bit configuration differs depending on the port group.

Bits 15–8 Reserved

Bits 7–0 P*x*CHATEN[7:0]

These bits enable/disable the chattering filter function.

1 (R/W): Enable (The chattering filter is used.)
0 (R/W): Disable (The chattering filter is bypassed.)

P*x* Port Mode Select Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*MODSEL	15–8	–	0x00	–	R	–
	7–0	P*x*SEL[7:0]	0x00	H0	R/W	

∗1: The bit configuration differs depending on the port group.
∗2: The initial value may be changed by the port.

Bits 15–8 Reserved

Bits 7–0 P*x*SEL[7:0]

These bits select whether each port is used for the GPIO function or a peripheral I/O function.

1 (R/W): Use peripheral I/O function

0 (R/W): Use GPIO function

P*x* Port Function Select Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*FNCSEL	15–14	P*x*7MUX[1:0]	0x0	H0	R/W	–
	13–12	P*x*6MUX[1:0]	0x0	H0	R/W	
	11–10	P*x*5MUX[1:0]	0x0	H0	R/W	
	9–8	P*x*4MUX[1:0]	0x0	H0	R/W	
	7–6	P*x*3MUX[1:0]	0x0	H0	R/W	
	5–4	P*x*2MUX[1:0]	0x0	H0	R/W	
	3–2	P*x*1MUX[1:0]	0x0	H0	R/W	
	1–0	P*x*0MUX[1:0]	0x0	H0	R/W	

∗1: The bit configuration differs depending on the port group.

∗2: The initial value may be changed by the port.

Bits 15–14 P*x*7MUX[1:0]

: :

Bits 1–0 P*x*0MUX[1:0]

These bits select the peripheral I/O function to be assigned to each port pin.

Table 6.6.1 Selecting Peripheral I/O Function

P*x*FNCSEL.P*xy*MUX[1:0] bits	Peripheral I/O function
0x3	Function 3
0x2	Function 2
0x1	Function 1
0x0	Function 0

This selection takes effect when the P*x*MODSEL.P*x*SEL*y* bit = 1.

P Port Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PCLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/WP	
	7–4	CLKDIV[3:0]	0x0	H0	R/WP	
	3–2	KRSTCFG[1:0]	0x0	H0	R/WP	
	1–0	CLKSRC[1:0]	0x0	H0	R/WP	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the PPORT operating clock is supplied in DEBUG mode or not.

1 (R/WP): Clock supplied in DEBUG mode

0 (R/WP): No clock supplied in DEBUG mode

Bits 7–4 CLKDIV[3:0]

These bits select the division ratio of the PPORT operating clock (chattering filter clock).

Bits 3–2 KRSTCFG[1:0]

These bits configure the key-entry reset function.

Table 6.6.2 Key-Entry Reset Function Settings

PCLK.KRSTCFG[1:0] bits	key-entry reset
0x3	Reset when P0[3:0] inputs = all low
0x2	Reset when P0[2:0] inputs = all low
0x1	Reset when P0[1:0] inputs = all low
0x0	Disable

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of PPORT (chattering filter).

The PPORT operating clock should be configured by selecting the clock source using the PCLK.CLKSRC[1:0] bits and the clock division ratio using the PCLK.CLKDIV[3:0] bits as shown in Table 6.6.3. These settings determine the input sampling time of the chattering filter.

Table 6.6.3 Clock Source and Division Ratio Settings

PCLK.CLKDIV[3:0] bits	PCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0xf	1/32,768			1/1
0xe	1/16,384			
0xd	1/8,192			
0xc	1/4,096			
0xb	1/2,048			
0xa	1/1,024			
0x9	1/512			
0x8	1/256			
0x7	1/128			
0x6	1/64			
0x5	1/32			
0x4	1/16			
0x3	1/8			
0x2	1/4			
0x1	1/2			
0x0	1/1			

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

P Port Interrupt Flag Group Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PINTFGRP	15–13	–	0x0	–	R	–
	12	PcINT	0	H0	R	
	11	PbINT	0	H0	R	
	10	PaINT	0	H0	R	
	9	P9INT	0	H0	R	
	8	P8INT	0	H0	R	
	7	P7INT	0	H0	R	
	6	P6INT	0	H0	R	
	5	P5INT	0	H0	R	
	4	P4INT	0	H0	R	
	3	P3INT	0	H0	R	
	2	P2INT	0	H0	R	
	1	P1INT	0	H0	R	
	0	P0INT	0	H0	R	

∗1: Only the bits corresponding to the port groups that support interrupts are provided.

Bits 15–13 Reserved

Bits 12–0 PxINT

These bits indicate that P*x* port group includes a port that has generated an interrupt.

1 (R): A port generated an interrupt

0 (R): No port generated an interrupt

The PINTFGRP.P*x*INT bit is cleared when the interrupt flag for the port that has generated an interrupt is cleared.

6.7 Control Register and Port Function Configuration of this IC

This section shows the PPORT control register/bit configuration in this IC and the list of peripheral I/O functions selectable for each port.

6.7.1 P0 Port Group

The P0 port group supports the GPIO, interrupt, and chattering filter functions.

Table 6.7.1.1 Control Registers for P0 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P0DAT (P0 Port Data Register)	15–8	P0OUT[7:0]	0x00	H0	R/W	–
	7–0	P0IN[7:0]	0x00	H0	R	
P0IOEN (P0 Port Enable Register)	15–8	P0IEN[7:0]	0x00	H0	R/W	–
	7–0	P0OEN[7:0]	0x00	H0	R/W	
P0RCTL (P0 Port Pull-up/down Control Register)	15–8	P0PDPU[7:0]	0x00	H0	R/W	–
	7–0	P0REN[7:0]	0x00	H0	R/W	
P0INTF (P0 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
	7–0	P0IF[7:0]	0x00	H0	R/W	Cleared by writing 1.
P0INTCTL (P0 Port Interrupt Control Register)	15–8	P0EDGE[7:0]	0x00	H0	R/W	–
	7–0	P0IE[7:0]	0x00	H0	R/W	
P0CHATEN (P0 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–
	7–0	P0CHATEN[7:0]	0x00	H0	R/W	
P0MODSEL (P0 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–0	P0SEL[7:0]	0x00	H0	R/W	
P0FNCSEL (P0 Port Function Select Register)	15–14	P07MUX[1:0]	0x0	H0	R/W	Valid settings: 0x0, 0x1
	13–12	P06MUX[1:0]	0x0	H0	R/W	
	11–10	P05MUX[1:0]	0x0	H0	R/W	
	9–8	P04MUX[1:0]	0x0	H0	R/W	
	7–6	P03MUX[1:0]	0x0	H0	R/W	
	5–4	P02MUX[1:0]	0x0	H0	R/W	
	3–2	P01MUX[1:0]	0x0	H0	R/W	
	1–0	P00MUX[1:0]	0x0	H0	R/W	

Table 6.7.1.2 P0 Port Group Function Assignment

Port name	P0SEL*y* = 0	P0SEL*y* = 1							
	GPIO	P0*y*MUX = 0x0 (Function 0)		P0*y*MUX = 0x1 (Function 1)		P0*y*MUX = 0x2 (Function 2)		P0*y*MUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
P00	P00	T16A3 Ch.0	TOUTA0/CAPA0	CLG	FOUT	–	–	–	–
P01	P01	T16A3 Ch.0	TOUTB0/CAPB0	PIO	#PIOWR	–	–	–	–
P02	P02	T16A3 Ch.0	EXCL0	PIO	#PIORD	–	–	–	–
P03	P03	CLG	EXOSC	PIO	#PIOCE	–	–	–	–
P04	P04	SPI Ch.0	#SPISS0	PIO	PIOA0	–	–	–	–
P05	P05	SPI Ch.0	SPICLK0	PIO	PIOA1	–	–	–	–
P06	P06	SPI Ch.0	SDI0	PIO	PIOA2	–	–	–	–
P07	P07	SPI Ch.0	SDO0	PIO	PIOA3	–	–	–	–

6.7.2 P1 Port Group

The P1 port group supports the GPIO, interrupt, and chattering filter functions.

Table 6.7.2.1 Control Registers for P1 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P1DAT (P1 Port Data Register)	15–8	P1OUT[7:0]	0x00	H0	R/W	–
	7–0	P1IN[7:0]	x	H0	R	
P1IOEN (P1 Port Enable Register)	15–8	P1IEN[7:0]	0x00	H0	R/W	–
	7–0	P1OEN[7:0]	0x00	H0	R/W	
P1RCTL (P1 Port Pull-up/down Control Register)	15–8	P1PDPU[7:0]	0x00	H0	R/W	–
	7–0	P1REN[7:0]	0x00	H0	R/W	
P1INTF (P1 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
	7–0	P1IF[7:0]	0x00	H0	R/W	Cleared by writing 1.
P1INTCTL (P1 Port Interrupt Control Register)	15–8	P1EDGE[7:0]	0x00	H0	R/W	–
	7–0	P1IE[7:0]	0x00	H0	R/W	
P1CHATEN (P1 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–
	7–0	P1CHATEN[7:0]	0x00	H0	R/W	
P1MODSEL (P1 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–0	P1SEL[7:0]	0x00	H0	R/W	
P1FNCSEL (P1 Port Function Select Register)	15–14	P17MUX[1:0]	0x0	H0	R/W	Valid settings: 0x0, 0x1
	13–12	P16MUX[1:0]	0x0	H0	R/W	
	11–10	P15MUX[1:0]	0x0	H0	R/W	
	9–8	P14MUX[1:0]	0x0	H0	R/W	
	7–6	P13MUX[1:0]	0x0	H0	R/W	
	5–4	P12MUX[1:0]	0x0	H0	R/W	
	3–2	P11MUX[1:0]	0x0	H0	R/W	
	1–0	P10MUX[1:0]	0x0	H0	R/W	

Table 6.7.2.2 P1 Port Group Function Assignment

Port name	P1SEL*y* = 0	P1SEL*y* = 1							
	GPIO	P1*y*MUX = 0x0 (Function 0)		P1*y*MUX = 0x1 (Function 1)		P1*y*MUX = 0x2 (Function 2)		P1*y*MUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
P10	P10	RFC Ch.0	SENB0	PIO	PIOA4	–	–	–	–
P11	P11	RFC Ch.0	SENA0	PIO	PIOA5	–	–	–	–
P12	P12	RFC Ch.0	REF0	PIO	PIOA6	–	–	–	–
P13	P13	RFC Ch.0	RFIN0	PIO	PIOA7	–	–	–	–
P14	P14	RFC Ch.1	SENB1	PIO	PIOD0	–	–	–	–
P15	P15	RFC Ch.1	SENA1	PIO	PIOD1	–	–	–	–
P16	P16	RFC Ch.1	REF1	PIO	PIOD2	–	–	–	–
P17	P17	RFC Ch.1	RFIN1	PIO	PIOD3	–	–	–	–

6.7.3 P2 Port Group

The P2 port group supports the GPIO function.

Table 6.7.3.1 Control Registers for P2 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P2DAT (P2 Port Data Register)	15–8	P2OUT[7:0]	0x00	H0	R/W	–
	7–0	P2IN[7:0]	x	H0	R	
P2IOEN (P2 Port Enable Register)	15–8	P2IEN[7:0]	0x00	H0	R/W	–
	7–0	P2OEN[7:0]	0x00	H0	R/W	

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P2RCTL (P2 Port Pull-up/down Control Register)	15–8	P2PDPU[7:0]	0x00	H0	R/W	–
	7–0	P2REN[7:0]	0x00	H0	R/W	
P2INTF P2INTCTL P2CHATEN	15–0	–	0x0000	–	R	–
P2MODSEL (P2 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–0	P2SEL[7:0]	0x00	H0	R/W	
P2FNCSEL (P2 Port Function Select Register)	15–14	P27MUX[1:0]	0x0	H0	R	Valid settings: 0x0
	13–12	P26MUX[1:0]	0x0	H0	R	
	11–10	P25MUX[1:0]	0x0	H0	R	
	9–8	P24MUX[1:0]	0x0	H0	R	
	7–6	P23MUX[1:0]	0x0	H0	R	Valid settings: 0x0, 0x1
	5–4	P22MUX[1:0]	0x0	H0	R	
	3–2	P21MUX[1:0]	0x0	H0	R/W	
	1–0	P20MUX[1:0]	0x0	H0	R/W	

Table 6.7.3.2 P2 Port Group Function Assignment

Port name	P2SEL*y* = 0	P2SEL*y* = 1							
	GPIO	P2*y*MUX = 0x0 (Function 0)		P2*y*MUX = 0x1 (Function 1)		P2*y*MUX = 0x2 (Function 2)		P2*y*MUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
P20	P20	SPI Ch.1	SDO1	PIO	PIOD4	–	–	–	–
P21	P21	SPI Ch.1	SDI1	PIO	PIOD5	–	–	–	–
P22	P22	SPI Ch.1	SPICLK1	PIO	PIOD6	–	–	–	–
P23	P23	SPI Ch.1	#SPISS1	PIO	PIOD7	–	–	–	–
P24	P24	SPI Ch.2	#SPISS2	–	–	–	–	–	–
P25	P25	SPI Ch.2	SPICLK2	–	–	–	–	–	–
P26	P26	SPI Ch.2	SDI2	–	–	–	–	–	–
P27	P27	SPI Ch.2	SDO2	–	–	–	–	–	–

6.7.4 P3 Port Group

The P3 port group supports the GPIO function.

Table 6.7.4.1 Control Registers for P3 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P3DAT (P3 Port Data Register)	15–8	P3OUT[7:0]	0x00	H0	R/W	–
	7–0	P3IN[7:0]	x	H0	R	
P3IOEN (P3 Port Enable Register)	15–8	P3IEN[7:0]	0x00	H0	R/W	–
	7–0	P3OEN[7:0]	0x00	H0	R/W	
P3RCTL (P3 Port Pull-up/down Control Register)	15–8	P3PDPU[7:0]	0x00	H0	R/W	–
	7–0	P3REN[7:0]	0x00	H0	R/W	
P3INTF P3INTCTL P3CHATEN	15–0	–	0x0000	–	R	–
P3MODSEL (P3 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–0	P3SEL[7:0]	0x00	H0	R/W	

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P3FNCSEL (P3 Port Function Select Register)	15–14	P37MUX[1:0]	0x0	H0	R	Valid settings: 0x0, 0x1
	13–12	P36MUX[1:0]	0x0	H0	R/W	
	11–10	P35MUX[1:0]	0x0	H0	R/W	
	9–8	P34MUX[1:0]	0x0	H0	R/W	
	7–6	P33MUX[1:0]	0x0	H0	R/W	
	5–4	P32MUX[1:0]	0x0	H0	R/W	
	3–2	P31MUX[1:0]	0x0	H0	R/W	Valid settings: 0x0, 0x1, 0x2
	1–0	P30MUX[1:0]	0x0	H0	R/W	Valid settings: 0x0, 0x1

Table 6.7.4.2 P3 Port Group Function Assignment

Port name	P3SEL*y* = 0	P3SEL*y* = 1							
	GPIO	P3*y*MUX = 0x0 (Function 0)		P3*y*MUX = 0x1 (Function 1)		P3*y*MUX = 0x2 (Function 2)		P3*y*MUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
P30	P30	T16A3 Ch.1	EXCL1	RFC Ch.0	RFCLKO0	–	–	–	–
P31	P31	TR	REGMON	RFC Ch.1	RFCLKO1	SVD	EXSVD	–	–
P32	P32	T16A3 Ch.1	TOUTB1/CAPB1	PIO	PIOA0	–	–	–	–
P33	P33	T16A3 Ch.1	TOUTA1/CAPA1	PIO	PIOA1	–	–	–	–
P34	P34	UART	USIN0	PIO	PIOA2	–	–	–	–
P35	P35	UART	USOUT0	PIO	PIOA3	–	–	–	–
P36	P36	I2C	SCL0	PIO	PIOD0	–	–	–	–
P37	P37	I2C	SDA0	PIO	PIOD1	–	–	–	–

6.7.5 P4 Port Group

The P4 port group supports the GPIO function.

Table 6.7.5.1 Control Registers for P4 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P4DAT (P4 Port Data Register)	15–10	–	0x00	–	R	–
	9–8	P4OUT[1:0]	0x0	H0	R/W	
	7–2	–	0x00	–	R	
	1–0	P4IN[1:0]	x	H0	R	
P4IOEN (P4 Port Enable Register)	15–10	–	0x00	–	R	–
	9–8	P4IEN[1:0]	0x0	H0	R/W	
	7–2	–	0x00	–	R	
	1–0	P4OEN[1:0]	0x0	H0	R/W	
P4RCTL (P4 Port Pull-up/down Control Register)	15–10	–	0x00	–	R	–
	9–8	P4PDPU[1:0]	0x0	H0	R/W	
	7–2	–	0x00	–	R	
	1–0	P4REN[1:0]	0x0	H0	R/W	
P4INTF P4INTCTL P4CHATEN	15–0	–	0x0000	–	R	–
P4MODSEL (P4 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1–0	P4SEL[1:0]	0x0	H0	R/W	
P4FNCSEL (P4 Port Function Select Register)	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3–2	P41MUX[1:0]	0x0	H0	R/W	Valid settings: 0x0, 0x1
	1–0	P40MUX[1:0]	0x0	H0	R/W	

Table 6.7.5.2 P4 Port Group Function Assignment

Port name	P4SEL*y* = 0	P4SEL*y* = 1							
	GPIO	P4*y*MUX = 0x0 (Function 0)		P4*y*MUX = 0x1 (Function 1)		P4*y*MUX = 0x2 (Function 2)		P4*y*MUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
P40	P40	UART	USIN0	PIO	PIOD2	–	–	–	–
P41	P41	UART	USOUT0	PIO	PIOD3	–	–	–	–

6.7.6 Pd Port Group

The Pd port group consists of three ports Pd0–Pd2 and they are configured as a debugging function port at initialization. These three ports support the GPIO function. The GPIO function of the Pd2 port supports output only, therefore, the pull-up/down function cannot be used.

Table 6.7.6.1 Control Registers for Pd Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PDDAT (Pd Port Data Register)	15–11	–	0x00	–	R	–
	10–8	PDOUT[2:0]	0x0	H0	R/W	
	7–2	–	0x00	–	R	
	1–0	PDIN[1:0]	x	H0	R	
PDIOEN (Pd Port Enable Register)	15–11	–	0x00	–	R	–
	10	reserved	0	H0	R/W	
	9–8	PDIEN[1:0]	0x0	H0	R/W	
	7–3	–	0x00	–	R	
	2	reserved	0	H0	R/W	
	1–0	PDOEN[1:0]	0x0	H0	R/W	
PDRCTL (Pd Port Pull-up/ down Control Register)	15–11	–	0x00	–	R	–
	10	reserved	0	H0	R/W	
	9–8	PDPDPU[1:0]	0x0	H0	R/W	
	7–3	–	0x00	–	R	
	2	reserved	0	H0	R/W	
	1–0	PDREN[1:0]	0x0	H0	R/W	
PDINTF PDINTCTL PDCHATEN	15–0	–	0x0000	–	R	–
PDMODSEL (Pd Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–3	–	0x00	–	R	
	2–0	PDSEL[2:0]	0x7	H0	R/W	
PDFNCSEL (Pd Port Function Select Register)	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5–4	PD2MUX[1:0]	0x0	H0	R/W	Valid settings: 0x0
	3–2	PD1MUX[1:0]	0x0	H0	R/W	
	1–0	PD0MUX[1:0]	0x0	H0	R/W	

Table 6.7.6.2 Pd Port Group Function Assignment

Port name	PdSEL*y* = 0	PdSEL*y* = 1							
	GPIO	Pd*y*MUX = 0x0 (Function 0)		Pd*y*MUX = 0x1 (Function 1)		Pd*y*MUX = 0x2 (Function 2)		Pd*y*MUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
Pd0	Pd0	DBG	DST2	–	–	–	–	–	–
Pd1	Pd1	DBG	DSIO	–	–	–	–	–	–
Pd2	Pd2	DBG	DCLK	–	–	–	–	–	–

6.7.7 Common Registers between Port Groups

Table 6.7.7.1 Control Registers for Common Use with Port Groups

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PCLK (P Port Clock Control Register)	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/WP	
	7–4	CLKDIV[3:0]	0x0	H0	R/WP	
	3–2	KRSTCFG[1:0]	0x0	H0	R/WP	
	1–0	CLKSRC[1:0]	0x0	H0	R/WP	
PINTFGRP (P Port Interrupt Flag Group Register)	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	P1INT	0	H0	R	
	0	P0INT	0	H0	R	

7 Watchdog Timer (WDT)

7.1 Overview

WDT restarts the system if a problem occurs, such as when the program cannot be executed normally.
The features of WDT are listed below.

- Includes a 10-bit up counter to count reset generation cycle.
- A counter clock source and clock division ratio are selectable.
- Counter overflow generates a reset.

Figure 7.1.1 shows the configuration of WDT.

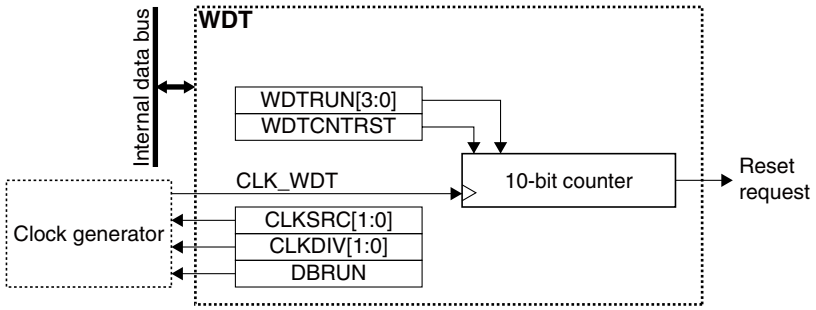


Figure 7.1.1 WDT Configuration

7.2 Clock Settings

7.2.1 WDT Operating Clock

When using WDT, the WDT operating clock CLK_WDT must be supplied to WDT from the clock generator.
The CLK_WDT supply should be controlled as in the procedure shown below.

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
3. Set the following WDTCLK register bits:
   WDTCLK.CLKSRC[1:0] bits (Clock source selection)
   WDTCLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)
4. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Use the following equation to calculate the WDT counter overflow cycle (reset generation cycle).

$$t_{WDT} = \frac{1{,}024}{CLK\_WDT} \qquad \text{(Eq. 7.1)}$$

Where

t_{WDT}: Counter overflow cycle [second]
CLK_WDT: WDT operating clock frequency [Hz]

Example) t_{WDT} = 4 seconds when CLK_WDT = 256 Hz

7.2.2 Clock Supply in DEBUG Mode

The CLK_WDT supply during DEBUG mode should be controlled using the WDTCLK.DBRUN bit.
The CLK_WDT supply to WDT is suspended when the CPU enters DEBUG mode if the WDTCLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_WDT supply resumes. Although WDT stops operating when the CLK_WDT supply is suspended, the register retains the status before DEBUG mode was entered.
If the WDTCLK.DBRUN bit = 1, the CLK_WDT supply is not suspended and WDT will keep operating in DEBUG mode.

7.3 Operations

7.3.1 WDT Control

Starting up WDT

WDT should be initialized and started up with the procedure listed below.

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Configure the WDT operating clock.
3. Write 1 to the WDTCTL.WDTCNTRST bit. (Reset WDT counter)
4. Write a value other than 0xa to the WDTCTL.WDTRUN[3:0] bits. (Start up WDT)
5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Resetting WDT

WDT generates a system reset when the counter overflows. To avert system restart by WDT, its embedded counter must be reset periodically via software while WDT is running.

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Write 1 to the WDTCTL.WDTCNTRST bit. (Reset WDT counter)
3. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

A location should be provided for periodically processing this routine. Process this routine within the t_{WDT} cycle. After resetting, WDT starts counting with a new reset generation cycle.
If WDT is not reset within the t_{WDT} cycle for any reason, a system reset is generated.

7.3.2 Operations in HALT and SLEEP Modes

During HALT mode

WDT operates in HALT mode. HALT mode is therefore cleared by a reset if it continues for more than the reset generation cycle and the reset handler is executed. To disable WDT in HALT mode, stop WDT by writing 0xa to the WDTCTL.WDTRUN[3:0] bits before executing the halt instruction. Reset WDT before resuming operations after HALT mode is cleared.

During SLEEP mode

WDT operates in SLEEP mode if the selected clock source is running. In this case SLEEP mode is cleared by a reset if it continues for more than the reset generation cycle and the reset handler is executed. Therefore, stop WDT by setting the WDTCTL.WDTRUN[3:0] bits before executing the slp instruction.
If the clock source stops in SLEEP mode, WDT stops. To prevent generation of an unnecessary reset after clearing SLEEP mode, reset WDT before executing the slp instruction. WDT should also be stopped as required using the WDTCTL.WDTRUN[3:0] bits.

7.4 Control Registers

WDT Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
WDTCLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/WP	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/WP	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/WP	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the WDT operating clock is supplied in DEBUG mode or not.
1 (R/WP): Clock supplied in DEBUG mode
0 (R/WP): No clock supplied in DEBUG mode

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits select the division ratio of the WDT operating clock (counter clock). The clock frequency should be set to around 256 Hz.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of WDT.

Table 7.4.1 Clock Source and Division Ratio Settings

WDTCLK. CLKDIV[1:0] bits	WDTCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x3	1/65,536	1/128	1/65,536	1/1
0x2	1/32,768		1/32,768	
0x1	1/16,384		1/16,384	
0x0	1/8,192		1/8,192	

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

WDT Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
WDTCTL	15–8	–	0x00	–	R	–
	7–5	–	0x0	–	R	
	4	WDTCNTRST	0	H0	WP	Always read as 0.
	3–0	WDTRUN[3:0]	0xa	H0	R/WP	–

Bits 15–5 Reserved

Bit 4 WDTCNTRST

This bit resets WDT.
1 (WP): Reset
0 (WP): Ignored
0 (R): Always 0 when being read

Bits 3–0 WDTRUN[3:0]

These bits control WDT to run and stop.
0xa (R/WP): Stop
Values other than 0xa (R/WP): Run

Always 0x0 is read if a value other than 0xa is written.
Since a reset may be generated immediately after running depending on the counter value, WDT should also be reset concurrently when running WDT.

8 Supply Voltage Detector (SVD)

8.1 Overview

SVD is a supply voltage detector to monitor the power supply voltage on the VDD pin or the voltage applied to an external pin. The main features are listed below.

- Power supply voltage to be detected: Selectable from VDD and an external power supply (EXSVD)
- Detectable voltage level: Selectable from among 19 levels (1.8 to 3.6 V)
- Detection results:
  - Can be read whether the power supply voltage is lower than the detection voltage level or not.
  - Can generate an interrupt or a reset when low power supply voltage is detected.
- Interrupt: 1 system (Low power supply voltage detection interrupt)
- Supports intermittent operations:
  - Three detection cycles are selectable.
  - Low power supply voltage detection count function to generate an interrupt/reset when low power supply voltage is successively detected the number of times specified.
  - Continuous operation is also possible.

Figure 8.1.1 shows the configuration of SVD.

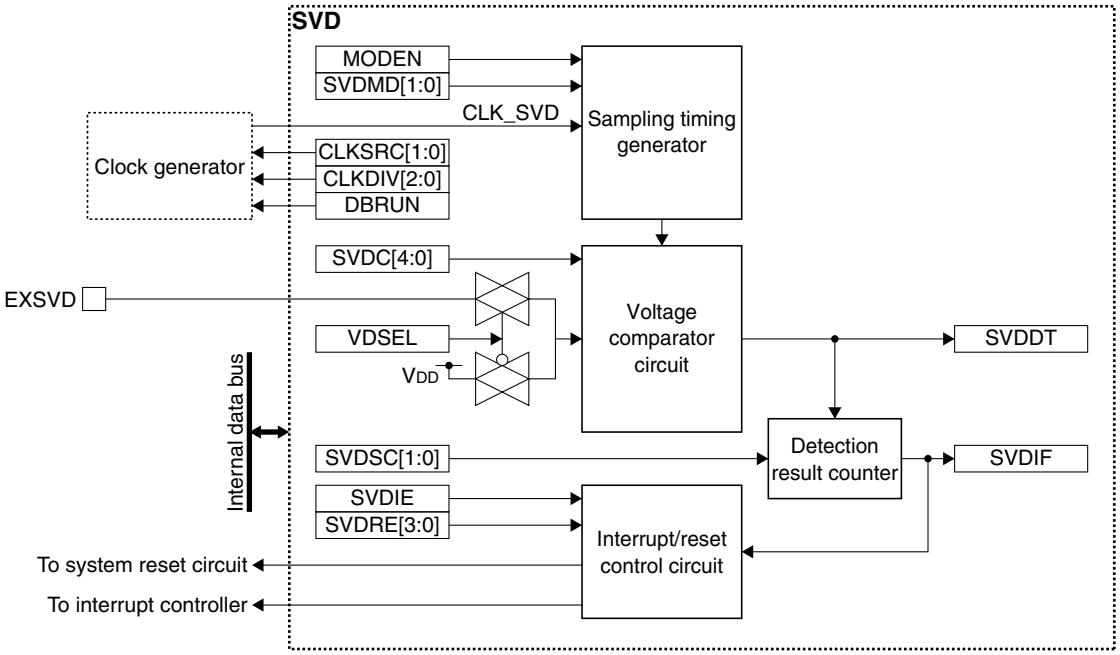


Figure 8.1.1 SVD Configuration

8.2 Input Pin and External Connection

8.2.1 Input Pin

Table 8.2.1.1 shows the SVD input pin.

Table 8.2.1.1 SVD Input Pin

Pin name	I/O*	Initial status*	Function
EXSVD	A	A (Hi-Z)	External power supply voltage detection pin

* Indicates the status when the pin is configured for SVD.

If the port is shared with the EXSVD pin and other functions, the EXSVD function must be assigned to the port before SVD can be activated. For more information, refer to the "I/O Ports" chapter.

8.2.2 External Connection

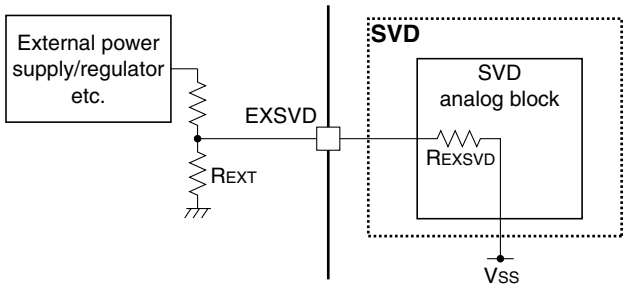


Figure 8.2.2.1 Connection between EXSVD Pin and External Power Supply

REXT resistance value must be determined so that it will be sufficiently smaller than the EXSVD input impedance REXSVD. For the EXSVD pin input voltage range and the EXSVD input impedance, refer to "Supply Voltage Detector Characteristics" in the "Electrical Characteristics" chapter.

8.3 Clock Settings

8.3.1 SVD Operating Clock

When using SVD, the SVD operating clock CLK_SVD must be supplied to SVD from the clock generator.
The CLK_SVD supply should be controlled as in the procedure shown below.

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
3. Set the following SVDCLK register bits:
   - SVDCLK.CLKSRC[1:0] bits (Clock source selection)
   - SVDCLK.CLKDIV[2:0] bits (Clock division ratio selection = Clock frequency setting)
4. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

The CLK_SVD frequency should be set to around 32 kHz.

8.3.2 Clock Supply in SLEEP Mode

When using SVD during SLEEP mode, the SVD operating clock CLK_SVD must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_SVD clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_SVD clock source is 1, the CLK_SVD clock source is deactivated during SLEEP mode and SVD stops with the register settings maintained at those before entering SLEEP mode. After the CPU returns to normal mode, CLK_SVD is supplied and the SVD operation resumes.

8.3.3 Clock Supply in DEBUG Mode

The CLK_SVD supply during DEBUG mode should be controlled using the SVDCLK.DBRUN bit.
The CLK_SVD supply to SVD is suspended when the CPU enters DEBUG mode if the SVDCLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_SVD supply resumes. Although SVD stops operating when the CLK_SVD supply is suspended, the registers retain the status before DEBUG mode was entered.
If the SVDCLK.DBRUN bit = 1, the CLK_SVD supply is not suspended and SVD will keep operating in DEBUG mode.

8.4 Operations

8.4.1 SVD Control

Starting detection

SVD should be initialized and activated with the procedure listed below.

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Configure the operating clock using the SVDCLK.CLKSRC[1:0] and SVDCLK.CLKDIV[2:0] bits.
3. Set the following SVDCTL register bits:
   - SVDCTL.VDSEL bit (Select detection voltage (V_{DD} or EXSVD))
   - SVDCTL.SVDSC[1:0] bits (Set low power supply voltage detection counter)
   - SVDCTL.SVDC[4:0] bits (Set SVD detection voltage V_{SVD})
   - SVDCTL.SVDRE[3:0] bits (Select reset/interrupt mode)
   - SVDCTL.SVDMD[1:0] bits (Set intermittent operation mode)
4. Set the following bits when using the interrupt:
   - Write 1 to the SVDINTF.SVDIF bit. (Clear interrupt flag)
   - Set the SVDINTE.SDVIE bit to 1. (Enable SVD interrupt)
5. Set the SVDCTL.MODEN bit to 1. (Enable SVD detection)
6. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Terminating detection

Follow the procedure shown below to stop SVD operation.

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Write 0 to the SVDCTL.MODEN bit. (Disable SVD detection)
3. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Reading detection results

The following two detection results can be obtained by reading the SVDINTF.SVDDT bit:

- Power supply voltage (V_{DD} or EXSVD) ≥ SVD detection voltage V_{SVD} when SVDINTF.SVDDT bit = 0
- Power supply voltage (V_{DD} or EXSVD) < SVD detection voltage V_{SVD} when SVDINTF.SVDDT bit = 1

Before reading the SVDINTF.SVDDT bit, wait for at least SVD circuit enable response time after 1 is written to the SVDCTL.MODEN bit (refer to "Supply Voltage Detector Characteristics, SVD circuit enable response time t_{SVDEN}" in the "Electrical Characteristics" chapter).
After the SVDCTL.SVDC[4:0] bits setting value is altered to change the SVD detection voltage V_{SVD} when the SVDCTL.MODEN bit = 1, wait for at least SVD circuit response time before reading the SVDINTF.SVDDT bit (refer to "Supply Voltage Detector Characteristics, SVD circuit response time t_{SVD}" in the "Electrical Characteristics" chapter).

8.4.2 SVD Operations

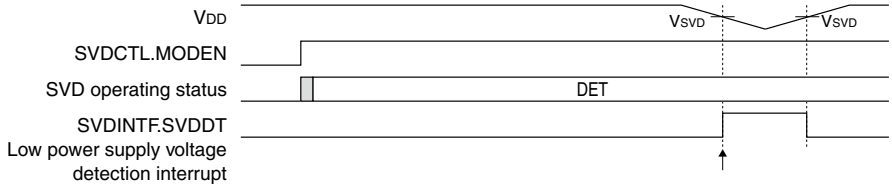
Continuous operation mode

SVD operates in continuous operation mode by default (SVDCTL.SVDMD[1:0] bits = 0x0). In this mode, SVD operates continuously while the SVDCTL.MODEN bit is set to 1 and it keeps loading the detection results to the SVDINTF.SVDDT bit. During this period, the current detection results can be obtained by reading the SVDINTF.SVDDT bit as necessary. Furthermore, an interrupt (if the SVDCTL.SVDRE[3:0] bits ≠ 0xa) or a reset (if the SVDCTL.SVDRE[3:0] bits = 0xa) can be generated when the SVDINTF.SVDDT bit is set to 1 (low power supply voltage is detected). This mode can keep detecting power supply voltage drop after the voltage detection masking time has elapsed even if the IC is placed into SLEEP status or accidental clock stoppage has occurred.

Intermittent operation mode

SVD operates in intermittent operation mode when the SVDCTL.SVDMD[1:0] bits are set to 0x1 to 0x3. In this mode, SVD turns on at an interval set using the SVDCTL.SVDMD[1:0] bits to perform detection operation and then it turns off while the SVDCTL.MODEN bit is set to 1. During this period, the latest detection results can be obtained by reading the SVDINTF.SVDDT bit as necessary. Furthermore, an interrupt or a reset can be generated when SVD has successively detected low power supply voltage the number of times specified by the SVDCTL.SVDSC[1:0] bits.

(1) When the SVDCTL.SVDMD[1:0] bits = 0x0 (continuous operation mode)



(2) When the SVDCTL.SVDMD[1:0] bits ≠ 0x0 (intermittent operation mode)

VSVD: Level set using the SVDCTL.SVDC[4:0] bits
: Voltage detection masking time
DET: Voltage detection operation

Figure 8.4.2.1 SVD Operations

8.5 SVD Interrupt and Reset

8.5.1 SVD Interrupt

Setting the SVDCTL.SVDRE[3:0] bits to a value other than 0xa allows use of the low power supply voltage detection interrupt function.

Table 8.5.1.1 Low Power Supply Voltage Detection Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Low power supply voltage detection	SVDINTF.SVDIF	In continuous operation mode When the SVDINTF.SVDDT bit is 1 In intermittent operation mode When low power supply voltage is successively detected the specified number of times	Writing 1

SVD provides the interrupt enable bit (SVDINTE.SVDIE bit) corresponding to the interrupt flag (SVDINTF.SVDIF bit). An interrupt request is sent to the interrupt controller only when the SVDINTF.SVDIF bit is set while the interrupt is enabled by the SVDINTE.SVDIE bit. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

Once the SVDINTF.SVDIF bit is set, it will not be cleared even if the power supply voltage subsequently returns to a value exceeding the SVD detection voltage VSVD. An interrupt may occur due to a temporary power supply voltage drop, check the power supply voltage status by reading the SVDINTF.SVDDT bit in the interrupt handler routine.

8.5.2 SVD Reset

Setting the SVDCTL.SVDRE[3:0] bits to 0xa allows use of the SVD reset issuance function.
The reset issuing timing is the same as that of the SVDINTF.SVDIF bit being set when a low voltage is detected.
After a reset has been issued, SVD enters continuous operation mode even if it was operating in intermittent operation mode, and continues operating. Issuing an SVD reset initializes the port assignment. However, when EXSVD is being detected, the input of the port for the EXSVD pin is sent to SVD so that SVD will continue the EXSVD detection operation.
If the power supply voltage reverts to the normal level, the SVDINTF.SVDDT bit goes 0 and the reset state is canceled. After that, SVD resumes operating in the operation mode set previously via the initialization routine.
During reset state, the SVD control bits are set as shown in Table 8.5.2.1.

Table 8.5.2.1 SVD Control Bits During Reset State

Control register	Control bit	Setting
SVDCLK	DBRUN	Reset to the initial values.
	CLKDIV[2:0]	
	CLKSRC[1:0]	
SVDCTL	VDSEL	The set value is retained.
	SVDSC[1:0]	Cleared to 0. (The set value becomes invalid as SVD enters continuous operation mode.)
	SVDC[4:0]	The set value is retained.
	SVDRE[3:0]	The set value (0xa) is retained.
	SVDMD[1:0]	Cleared to 0 to set continuous operation mode.
	MODEN	The set value (1) is retained.
SVDINTF	SVDIF	The status (1) before being reset is retained.
SVDINTE	SVDIE	Cleared to 0.

8.6 Control Registers

SVD Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVDCLK	15–9	–	0x00	–	R	–
	8	DBRUN	1	H0	R/WP	
	7	–	0	–	R	
	6–4	CLKDIV[2:0]	0x0	H0	R/WP	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/WP	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the SVD operating clock is supplied in DEBUG mode or not.
1 (R/WP): Clock supplied in DEBUG mode
0 (R/WP): No clock supplied in DEBUG mode

Bit 7 Reserved

Bits 6–4 CLKDIV[2:0]

These bits select the division ratio of the SVD operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of SVD.

Table 8.6.1 Clock Source and Division Ratio Settings

SVDCLK.CLKDIV[2:0] bits	SVDCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x6, 0x7	Reserved	1/1	Reserved	1/1
0x5	1/512		1/512	
0x4	1/256		1/256	
0x3	1/128		1/128	
0x2	1/64		1/64	
0x1	1/32		1/32	
0x0	1/16		1/16	

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The clock frequency should be set to around 32 kHz.

SVD Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVDCTL	15	VDSEL	0	H1	R/WP	–
	14–13	SVDSC[1:0]	0x0	H0	R/WP	Writing takes effect when the SVDCTL.SVDMD[1:0] bits are not 0x0.
	12–8	SVDC[4:0]	0x00	H1	R/WP	–
	7–4	SVDRE[3:0]	0x0	H1	R/WP	
	3	–	0	–	R	
	2–1	SVDMD[1:0]	0x0	H0	R/WP	
	0	MODEN	0	H1	R/WP	

Bit 15 VDSEL

This bit selects the power supply voltage to be detected by SVD.

1 (R/WP): Voltage applied to the EXSVD pin

0 (R/WP): V_{DD}

Bits 14–13 SVDSC[1:0]

These bits set the condition to generate an interrupt/reset (number of successive low voltage detections) in intermittent operation mode (SVDCTL.SVDMD[1:0] bits = 0x1 to 0x3).

Table 8.6.2 Interrupt/Reset Generating Condition in Intermittent Operation Mode

SVDCTL.SVDSC[1:0] bits	Interrupt/reset generating condition
0x3	Low power supply voltage is successively detected eight times.
0x2	Low power supply voltage is successively detected four times.
0x1	Low power supply voltage is successively detected twice.
0x0	Low power supply voltage is successively detected once.

This setting is ineffective in continuous operation mode (SVDCTL.SVDMD[1:0] bits = 0x0).

Bits 12–8 SVDC[4:0]

These bits select an SVD detection voltage V_{SVD} for detecting low voltage.

Table 8.6.3 Setting of SVD Detection Voltage V_{SVD}

SVDCTL.SVDC[4:0] bits	SVD detection voltage V_{SVD} [V]
0x1f	High
0x1e	↑
:	
0x0d	↓
0x0c	Low
0x0b–0x00	Use prohibited

For more information, refer to “Supply Voltage Detector Characteristics, SVD detection voltage V_{SVD}” in the “Electrical Characteristics” chapter.

Bits 7–4 SVDRE[3:0]

These bits enable/disable the reset issuance function when a low power supply voltage is detected.
0xa (R/WP): Enable (Issue reset)
Other than 0xa (R/WP): Disable (Generate interrupt)

For more information on the SVD reset issuance function, refer to "SVD Reset."

Bit 3 Reserved

Bits 2–1 SVDMD[1:0]

These bits select intermittent operation mode and its detection cycle.

Table 8.6.4 Intermittent Operation Mode Detection Cycle Selection

SVDCTL.SVDMD[1:0] bits	Operation mode (detection cycle)
0x3	Intermittent operation mode (CLK_SVD/512)
0x2	Intermittent operation mode (CLK_SVD/256)
0x1	Intermittent operation mode (CLK_SVD/128)
0x0	Continuous operation mode

For more information on intermittent and continuous operation modes, refer to "SVD Operations."

Bit 0 MODEN

This bit enables/disables for the SVD circuit to operate.
1 (R/WP): Enable (Start detection operations)
0 (R/WP): Disable (Stop detection operations)

After this bit has been altered, wait until the value written is read out from this bit without subsequent operations being performed.

Notes: • Writing 0 to the SVDCTL.MODEN bit resets the SVD hardware. However, the register values set and the interrupt flag are not cleared. The SVDCTL.MODEN bit is actually set to 0 after this processing has finished. If 1 is written to the SVDCTL.MODEN bit continuously without waiting for the bit being read as 0 at this time, writing 0 may be ignored and a malfunction may occur as the hardware restarts without resetting.

• The SVD internal circuit is initialized if the SVDCTL.SVDSC[1:0] bits, SVDCTL.SVDRE[3:0] bits, or SVDCTL.SVDMD[1:0] bits are altered while SVD is in operation after 1 is written to the SVDCTL.MODEN bit.

SVD Status and Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVDINTF	15–9	–	0x00	–	R	–
	8	SVDDT	x	–	R	
	7–1	–	0x00	–	R	
	0	SVDIF	0	H1	R/W	Cleared by writing 1.

Bits 15–9 Reserved

Bit 8 SVDDT

The power supply voltage detection results can be read out from this bit.
1 (R): Power supply voltage (V_{DD} or EXSVD) < SVD detection voltage V_{SVD}
0 (R): Power supply voltage (V_{DD} or EXSVD) ≥ SVD detection voltage V_{SVD}

Bits 7–1 Reserved

Bit 0 SVDIF

This bit indicates the low power supply voltage detection interrupt cause occurrence status.
1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

Note: The SVD internal circuit is initialized if the interrupt flag is cleared while SVD is in operation after 1 is written to the SVDCTL.MODEN bit.

SVD Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVDINTE	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	SVDIE	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 SVDIE

This bit enables low power supply voltage detection interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

Notes: • If the SVDCTL.SVDRE[3:0] bits are set to 0xa, no low power supply voltage detection interrupt will occur, as a reset is issued at the same timing as an interrupt.

• To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.

9 16-bit Timers (T16)

9.1 Overview

T16 is a 16-bit timer. The features of T16 are listed below.

- 16-bit presettable down counter
- Provides a reload data register for setting the preset value.
- A clock source and clock division ratio for generating the count clock are selectable.
- Repeat mode or one-shot mode is selectable.
- Can generate counter underflow interrupts.

Figure 9.1.1 shows the configuration of a T16 channel.

Channel configuration in this IC
- Number of channels: 4 channels (Ch.0 to Ch.3)
- Event counter function: Unavailable
- Peripheral clock output: Ch.1 → SPI Ch.0 master clock
  Ch.2 → SPI Ch.1 master clock
  Ch.3 → SPI Ch.2 master clock
  (Outputs the counter underflow signal)

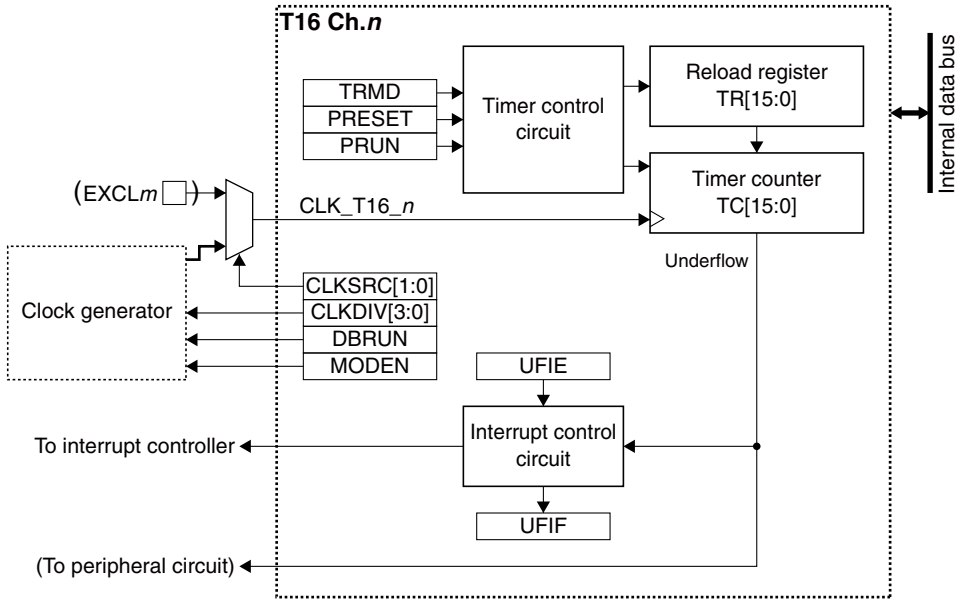


Figure 9.1.1 Configuration of a T16 Channel

9.2 Input Pin

Table 9.2.1 shows the T16 input pin.

Table 9.2.1 T16 Input Pin

Pin name	I/O*	Initial status*	Function
EXCL*m*	I	I (Hi-Z)	External event signal input pin

* Indicates the status when the pin is configured for T16.

If the port is shared with the EXCL*m* pin and other functions, the EXCL*m* input function must be assigned to the port before using the event counter function. For more information, refer to the "I/O Ports" chapter.

9.3 Clock Settings

9.3.1 T16 Operating Clock

When using T16 Ch.*n*, the T16 Ch.*n* operating clock CLK_T16_*n* must be supplied to T16 Ch.*n* from the clock generator. The CLK_T16_*n* supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Set the following T16_*n*CLK register bits:
   - T16_*n*CLK.CLKSRC[1:0] bits (Clock source selection)
   - T16_*n*CLK.CLKDIV[3:0] bits (Clock division ratio selection = Clock frequency setting)

9.3.2 Clock Supply in SLEEP Mode

When using T16 during SLEEP mode, the T16 operating clock CLK_T16_*n* must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_T16_*n* clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_T16_*n* clock source is 1, the CLK_T16_*n* clock source is deactivated during SLEEP mode and T16 stops with the register settings and counter value maintained at those before entering SLEEP mode. After the CPU returns to normal mode, CLK_T16_*n* is supplied and the T16 operation resumes.

9.3.3 Clock Supply in DEBUG Mode

The CLK_T16_*n* supply during DEBUG mode should be controlled using the T16_*n*CLK.DBRUN bit.
The CLK_T16_*n* supply to T16 Ch.*n* is suspended when the CPU enters DEBUG mode if the T16_*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_T16_*n* supply resumes. Although T16 Ch.*n* stops operating when the CLK_T16_*n* supply is suspended, the counter and registers retain the status before DEBUG mode was entered. If the T16_*n*CLK.DBRUN bit = 1, the CLK_T16_*n* supply is not suspended and T16 Ch.*n* will keep operating in DEBUG mode.

9.3.4 Event Counter Clock

The channel that supports the event counter function counts down at the rising edge of the EXCL*m* pin input signal when the T16_*n*CLK.CLKSRC[1:0] bits are set to 0x3.

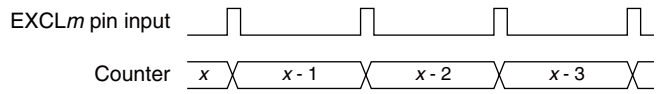


Figure 9.3.3.1 Count Down Timing

Note that the EXOSC clock is selected for the channel that does not support the event counter function.

9.4 Operations

9.4.1 Initialization

T16 Ch.*n* should be initialized and started counting with the procedure shown below.

1. Configure the T16 Ch.*n* operating clock (see "T16 Operating Clock").
2. Set the T16_*n*CTL.MODEN bit to 1. (Enable count operation clock)
3. Set the T16_*n*MOD.TRMD bit. (Select operation mode (Repeat mode or One-shot mode)).
4. Set the T16_*n*TR register. (Set reload data (counter preset data))
5. Set the following bits when using the interrupt:
   - Write 1 to the T16_*n*INTF.UFIF bit. (Clear interrupt flag)
   - Set the T16_*n*INTE.UFIE bit to 1. (Enable underflow interrupt)

6. Set the following T16_*n*CTL register bits:
   - Set the T16_*n*CTL.PRESET bit to 1. (Preset reload data to counter)
   - Set the T16_*n*CTL.PRUN bit to 1. (Start counting)

9.4.2 Counter Underflow

Normally, the T16 counter starts counting down from the reload data value preset and generates an underflow signal when an underflow occurs. This signal is used to generate an interrupt and may be output to a specific peripheral circuit as a clock (T16 Ch.*n* must be set to repeat mode to generate a clock). The underflow cycle is determined by the T16 Ch.*n* operating clock setting and reload data (counter initial value) set in the T16_*n*TR register.
The following shows the equations to calculate the underflow cycle and frequency:

$$T = \frac{TR + 1}{f_{CLK\_T16\_n}} \qquad f_T = \frac{f_{CLK\_T16\_n}}{TR + 1} \qquad \text{(Eq. 9.1)}$$

Where

T: Underflow cycle [s]
fT: Underflow frequency [Hz]
TR: T16_*n*TR register setting
$f_{CLK\_T16\_n}$: T16 Ch.*n* operating clock frequency [Hz]

9.4.3 Operations in Repeat Mode

T16 Ch.*n* enters repeat mode by setting T16_*n*MOD.TRMD bit to 0.
In repeat mode, the count operation starts by writing 1 to the T16_*n*CTL.PRUN bit and continues until 0 is written. A counter underflow presets the T16_*n*TR register value to the counter, so underflow occurs periodically. Select this mode to generate periodic underflow interrupts or when using the timer to output a trigger/clock to the peripheral circuit.

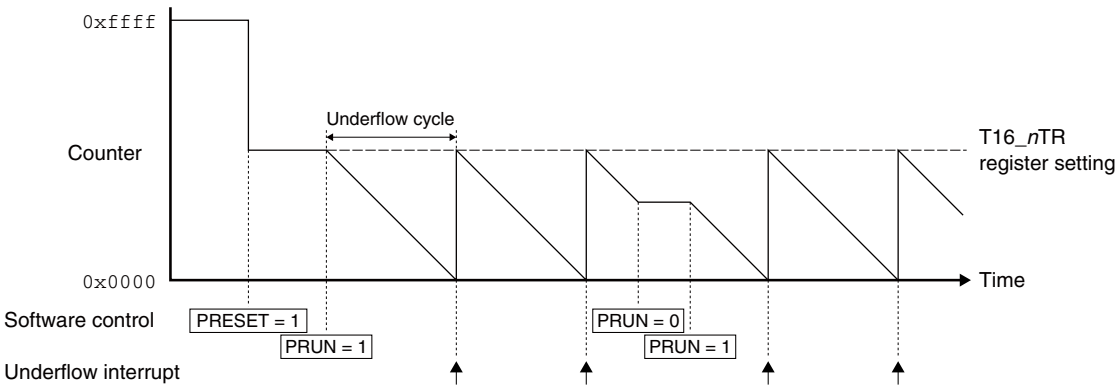


Figure 9.4.3.1 Count Operations in Repeat Mode

9.4.4 Operations in One-shot Mode

T16 Ch.*n* enters one-shot mode by setting the T16_*n*MOD.TRMD bit to 1.
In one-shot mode, the count operation starts by writing 1 to the T16_*n*CTL.PRUN bit and stops after the T16_*n*TR register value is preset to the counter when an underflow has occurred. At the same time the counter stops, the T16_*n*CTL.PRUN bit is cleared automatically. Select this mode to stop the counter after an interrupt has occurred once, such as for checking a specific lapse of time.

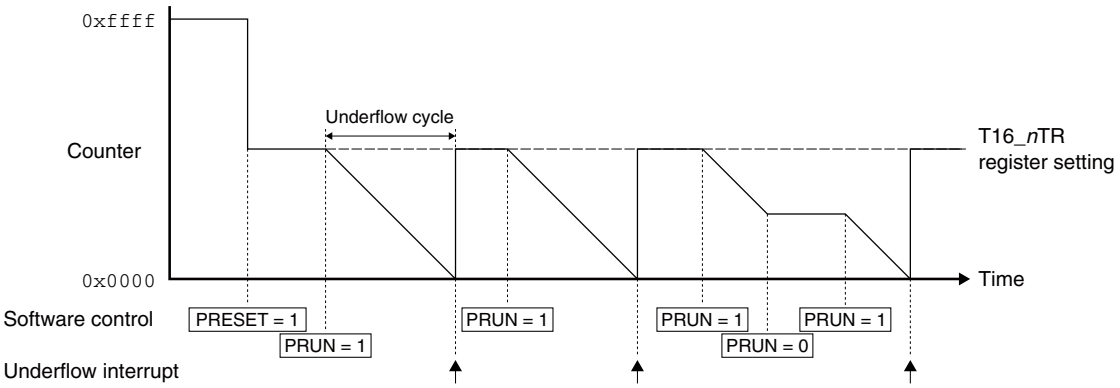


Figure 9.4.4.1 Count Operations in One-shot Mode

9.4.5 Counter Value Read

The counter value can be read out from the T16_*n*TC.TC[15:0] bits. However, since T16 operates on CLK_T16_*n*, one of the operations shown below is required to read correctly by the CPU.

- Read the counter value twice or more and check to see if the same value is read.
- Stop the timer and then read the counter value.

9.5 Interrupt

Each T16 channel has a function to generate the interrupt shown in Table 9.5.1.

Table 9.5.1 T16 Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Underflow	T16_*n*INTF.UFIF	When the counter underflows	Writing 1

T16 provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

9.6 Control Registers

T16 Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*CLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–4	CLKDIV[3:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the T16 Ch.*n* operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–4 CLKDIV[3:0]

These bits select the division ratio of the T16 Ch.*n* operating clock (counter clock).

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of T16 Ch.*n*.

Table 9.6.1 Clock Source and Division Ratio Settings

T16_*n*CLK.CLKDIV[3:0] bits	T16_*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC/EXCL*m*
0xf	1/32,768	1/1	1/32,768	1/1
0xe	1/16,384		1/16,384	
0xd	1/8,192		1/8,192	
0xc	1/4,096		1/4,096	
0xb	1/2,048		1/2,048	
0xa	1/1,024		1/1,024	
0x9	1/512		1/512	
0x8	1/256	1/256	1/256	
0x7	1/128	1/128	1/128	
0x6	1/64	1/64	1/64	
0x5	1/32	1/32	1/32	
0x4	1/16	1/16	1/16	
0x3	1/8	1/8	1/8	
0x2	1/4	1/4	1/4	
0x1	1/2	1/2	1/2	
0x0	1/1	1/1	1/1	

(Note 1) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.
(Note 2) When the T16_*n*CLK.CLKSRC[1:0] bits are set to 0x3, EXCL*m* is selected for the channel with an event counter function or EXOSC is selected for other channels.

T16 Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*MOD	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	TRMD	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 TRMD

This bit selects the T16 operation mode.
1 (R/W): One-shot mode
0 (R/W): Repeat mode

For detailed information on the operation mode, refer to "Operations in One-shot Mode" and "Operations in Repeat Mode."

T16 Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*CTL	15–9	–	0x00	–	R	–
	8	PRUN	0	H0	R/W	
	7–2	–	0x00	–	R	
	1	PRESET	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 PRUN

This bit starts/stops the timer.
1 (W): Start timer
0 (W): Stop timer
1 (R): Timer is running
0 (R): Timer is idle

By writing 1 to this bit, the timer starts count operations. However, the T16_*n*CTL.MODEN bit must be set to 1 in conjunction with this bit or it must be set in advance. While the timer is running, writing 0 to this bit stops count operations. When the counter stops due to a counter underflow in one-shot mode, this bit is automatically cleared to 0.

Bits 7–2 Reserved

Bit 1 PRESET

This bit presets the reload data stored in the T16_*n*TR register to the counter.

1 (W): Preset
0 (W): Ineffective
1 (R): Presetting in progress
0 (R): Presetting finished or normal operation

By writing 1 to this bit, the timer presets the T16_*n*TR register value to the counter. However, the T16_*n*CTL.MODEN bit must be set to 1 in conjunction with this bit or it must be set in advance. This bit retains 1 during presetting and is automatically cleared to 0 after presetting has finished.

Bit 0 MODEN

This bit enables the T16 Ch.*n* operations.

1 (R/W): Enable (Start supplying operating clock)
0 (R/W): Disable (Stop supplying operating clock)

T16 Ch.*n* Reload Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*TR	15–0	TR[15:0]	0xffff	H0	R/W	–

Bits 15–0 TR[15:0]

These bits are used to set the initial value to be preset to the counter.
The value set to this register will be preset to the counter when 1 is written to the T16_*n*CTL.PRESET bit or when the counter underflows.

Notes: • The T16_*n*TR register cannot be altered while the timer is running (T16_*n*CTL.PRUN bit = 1), as an incorrect initial value may be preset to the counter.

• When one-shot mode is set, the T16_*n*TR.TR[15:0] bits should be set to a value equal to or greater than 0x0001.

T16 Ch.*n* Counter Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*TC	15–0	TC[15:0]	0xffff	H0	R	–

Bits 15–0 TC[15:0]

The current counter value can be read out from these bits.

T16 Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*INTF	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	UFIF	0	H0	R/W	Cleared by writing 1.

Bits 15–1 Reserved

Bit 0 UFIF

This bit indicates the T16 Ch.*n* underflow interrupt cause occurrence status.

1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

T16 Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16_*n*INTE	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	UFIE	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 UFIE

This bit enables T16 Ch.*n* underflow interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

Note: To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.

10 UART (UART)

10.1 Overview

The UART is an asynchronous serial interface. The features of the UART are listed below.

- Includes a baud rate generator for generating the transfer clock.
- Supports 7- and 8-bit data length (LSB first).
- Odd parity, even parity, or non-parity mode is selectable.
- The start bit length is fixed at 1 bit.
- The stop bit length is selectable from 1 bit and 2 bits.
- Supports full-duplex communications.
- Includes a 2-byte receive data buffer and a 1-byte transmit data buffer.
- Includes an RZI modulator/demodulator circuit to support IrDA 1.0-compatible infrared communications.
- Can detect parity error, framing error, and overrun error.
- Can generate receive buffer full (1 byte/2 bytes), transmit buffer empty, end of transmission, parity error, framing error, and overrun error interrupts.
- Input pin can be pulled up with an internal resistor.
- The output pin is configurable as an open-drain output.

Figure 10.1.1 shows the UART configuration.

Channel configuration in this IC
- 1 channel (Ch.0)

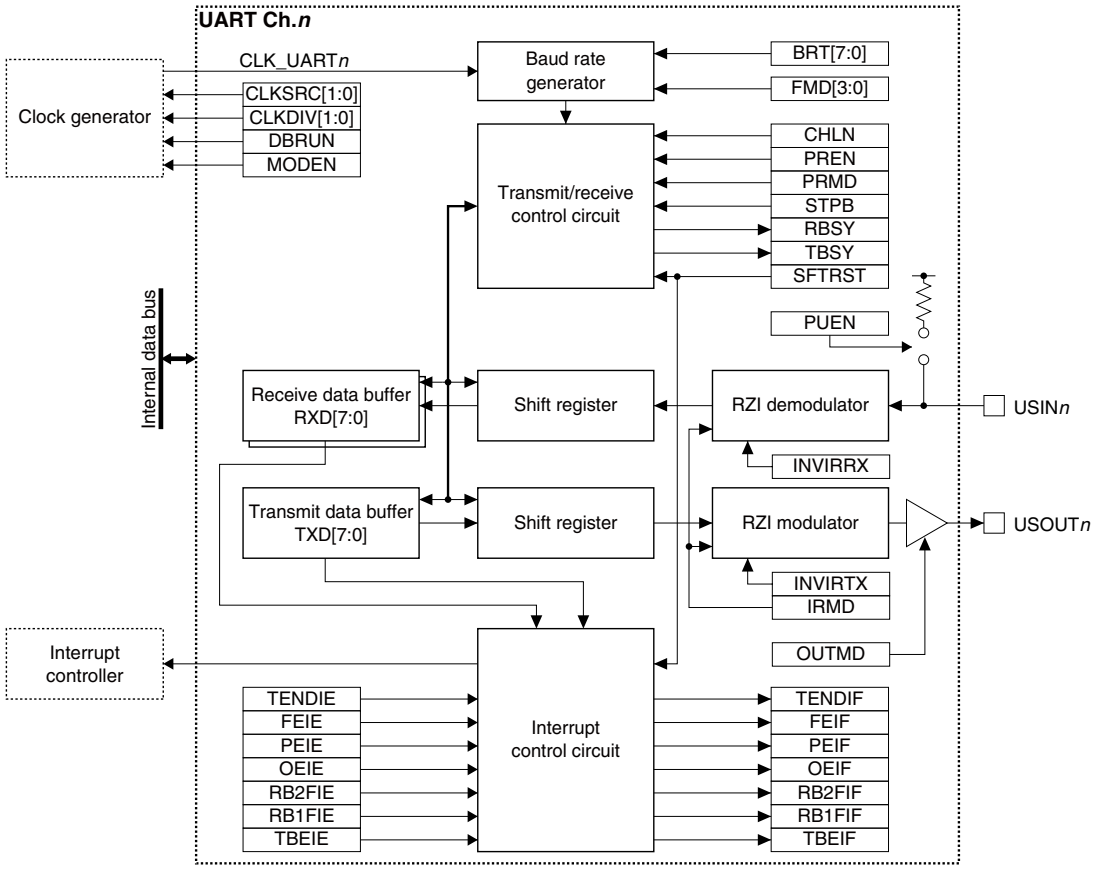


Figure 10.1.1 UART Configuration

10.2 Input/Output Pins and External Connections

10.2.1 List of Input/Output Pins

Table 10.2.1.1 lists the UART pins.

Table 10.2.1.1 List of UART Pins

Pin name	I/O*	Initial status*	Function
USIN*n*	I	I (Hi-Z)	UART Ch.*n* data input pin
USOUT*n*	O	O (High)	UART Ch.*n* data output pin

* Indicates the status when the pin is configured for the UART.

If the port is shared with the UART pin and other functions, the UART input/output function must be assigned to the port before activating the UART. For more information, refer to the "I/O Ports" chapter.

10.2.2 External Connections

Figure 10.2.2.1 shows a connection diagram between the UART in this IC and an external UART device.

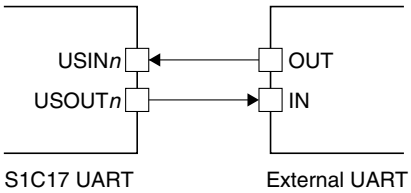


Figure 10.2.2.1 Connections between UART and an External UART Device

10.2.3 Input Pin Pull-Up Function

The UART includes a pull-up resistor for the USIN*n* pin. Setting the UA*n*MOD.PUEN bit to 1 enables the resistor to pull up the USIN*n* pin.

10.2.4 Output Pin Open-Drain Output Function

The USOUT*n* pin supports the open-drain output function. Default configuration is a push-pull output and it is switched to an open-drain output by setting the UA*n*MOD.OUTMD bit to 1.

10.3 Clock Settings

10.3.1 UART Operating Clock

When using the UART Ch.*n*, the UART Ch.*n* operating clock CLK_UART*n* must be supplied to the UART Ch.*n* from the clock generator. The CLK_UART*n* supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Set the following UA*n*CLK register bits:
   - UA*n*CLK.CLKSRC[1:0] bits (Clock source selection)
   - UA*n*CLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)

   The UART operating clock should be selected so that the baud rate generator will be configured easily.

10.3.2 Clock Supply in SLEEP Mode

When using the UART during SLEEP mode, the UART operating clock CLK_UART*n* must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_UART*n* clock source.

10.3.3 Clock Supply in DEBUG Mode

The CLK_UART*n* supply during DEBUG mode should be controlled using the UA*n*CLK.DBRUN bit.
The CLK_UART*n* supply to the UART Ch.*n* is suspended when the CPU enters DEBUG mode if the UA*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_UART*n* supply resumes. Although the UART Ch.*n* stops operating when the CLK_UART*n* supply is suspended, the output pin and registers retain the status before DEBUG mode was entered. If the UA*n*CLK.DBRUN bit = 1, the CLK_UART*n* supply is not suspended and the UART Ch.*n* will keep operating in DEBUG mode.

10.3.4 Baud Rate Generator

The UART includes a baud rate generator to generate the transfer (sampling) clock. The transfer rate is determined by the UA*n*BR.BRT[7:0] and UA*n*BR.FMD[3:0] bit settings. Use the following equations to calculate the setting values for obtaining the desired transfer rate.

$$bps = \frac{CLK\_UART}{\{(BRT + 1) \times 16 + FMD\}} \qquad BRT = \left(\frac{CLK\_UART}{bps} - FMD - 16\right) \div 16 \qquad \text{(Eq. 10.1)}$$

Where

CLK_UART: UART operating clock frequency [Hz]
bps: Transfer rate [bit/s]
BRT: UA*n*BR.BRT[7:0] setting value (0 to 255)
FMD: UA*n*BR.FMD[3:0] setting value (0 to 15)

For the transfer rate range configurable in the UART, refer to "UART Characteristics, Transfer baud rates U_{BRT1} and U_{BRT2}" in the "Electrical Characteristics" chapter.

10.4 Data Format

The UART allows setting of the data length, stop bit length, and parity function. The start bit length is fixed at one bit.

Data length

With the UA*n*MOD.CHLN bit, the data length can be set to seven bits (UA*n*MOD.CHLN bit = 0) or eight bits (UA*n*MOD.CHLN bit = 1).

Stop bit length

With the UA*n*MOD.STPB bit, the stop bit length can be set to one bit (UA*n*MOD.STPB bit = 0) or two bits (UA*n*MOD.STPB bit = 1).

Parity function

The parity function is configured using the UA*n*MOD.PREN and UA*n*MOD.PRMD bits.

Table 10.4.1 Parity Function Setting

UA*n*MOD.PREN bit	UA*n*MOD.PRMD bit	Parity function
1	1	Odd parity
1	0	Even parity
0	*	Non parity

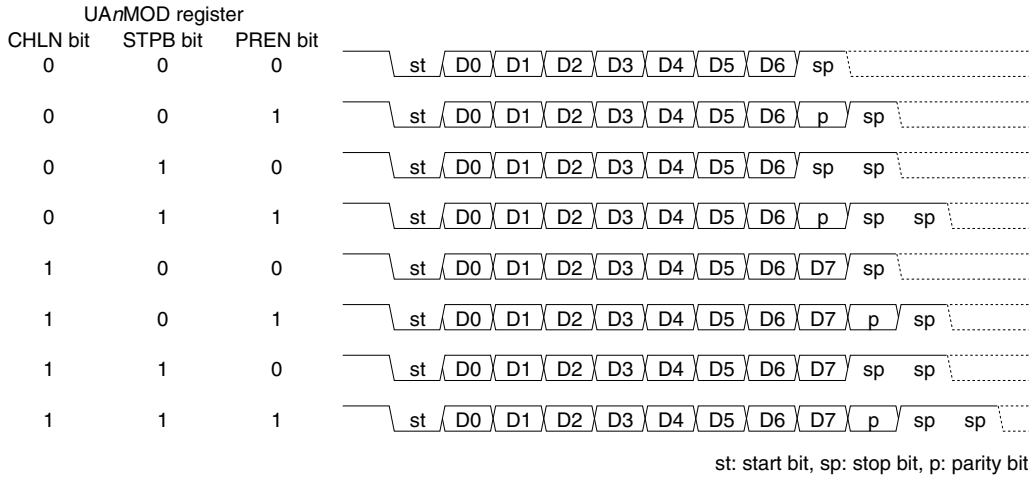


Figure 10.4.1 Data Format

10.5 Operations

10.5.1 Initialization

The UART Ch.*n* should be initialized with the procedure shown below.

1. Assign the UART Ch.*n* input/output function to the ports. (Refer to the "I/O Ports" chapter.)
2. Set the UA*n*CLK.CLKSRC[1:0] and UA*n*CLK.CLKDIV[1:0] bits. (Configure operating clock)
3. Configure the following UA*n*MOD register bits:
   - UA*n*MOD.PUEN bit (Enable/disable USIN*n* pin pull-up)
   - UA*n*MOD.OUTMD bit (Enable/disable USOUT*n* pin open-drain output)
   - UA*n*MOD.IRMD bit (Enable/disable IrDA interface)
   - UA*n*MOD.CHLN bit (Set 7/8-bit data length)
   - UA*n*MOD.PREN bit (Enable/disable parity function)
   - UA*n*MOD.PRMD bit (Even/odd parity selection)
   - UA*n*MOD.STPB bit (Set 1/2-bit stop bit length)
4. Set the UA*n*BR.BRT[7:0] and UA*n*BR.FMD[3:0] bits. (Set transfer rate)
5. Set the following UA*n*CTL register bits:
   - Set the UA*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the UA*n*CTL.MODEN bit to 1. (Enable UART Ch.*n* operations)
6. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the UA*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the UA*n*INTE register to 1. * (Enable interrupts)

   * The initial value of the UA*n*INTF.TBEIF bit is 1, therefore, an interrupt will occur immediately after the UA*n*INTE.TBEIE bit is set to 1.

10.5.2 Data Transmission

A data sending procedure and the UART Ch.*n* operations are shown below. Figures 10.5.2.1 and 10.5.2.2 show a timing chart and a flowchart, respectively.

Data sending procedure

1. Check to see if the UA*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
2. Write transmit data to the UA*n*TXD register.
3. Wait for a UART interrupt when using the interrupt.
4. Repeat Steps 1 to 3 (or 1 and 2) until the end of transmit data.

UART data sending operations

The UART Ch.*n* starts data sending operations when transmit data is written to the UA*n*TXD register.
The transmit data in the UA*n*TXD register is automatically transferred to the shift register and the UA*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
The USOUT*n* pin outputs a start bit and the UA*n*INTF.TBSY bit is set to 1 (transmit busy). The shift register data bits are then output successively from the LSB. Following output of MSB, the parity bit (if parity is enabled) and the stop bit are output.

Even if transmit data is being output from the USOUT*n* pin, the next transmit data can be written to the UA*n*TXD register after making sure the UA*n*INTF.TBEIF bit is set to 1.

If no transmit data remains in the UA*n*TXD register after the stop bit has been output from the USOUT*n* pin, the UA*n*INTF.TBSY bit is cleared to 0 and the UA*n*INTF.TENDIF bit is set to 1 (transmission completed).

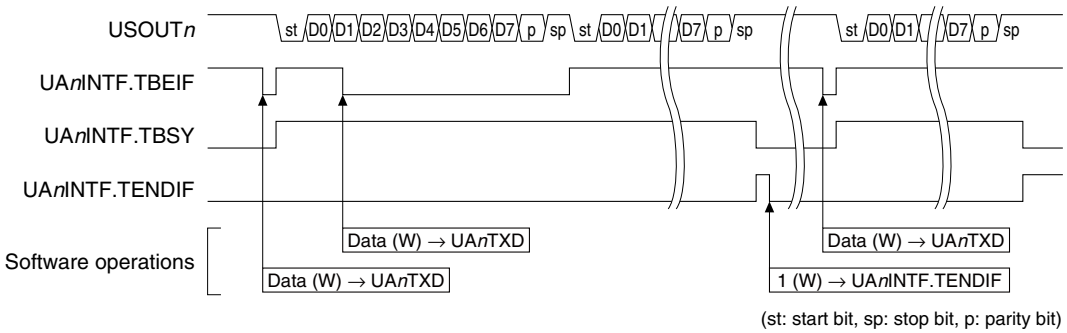


Figure 10.5.2.1 Example of Data Sending Operations

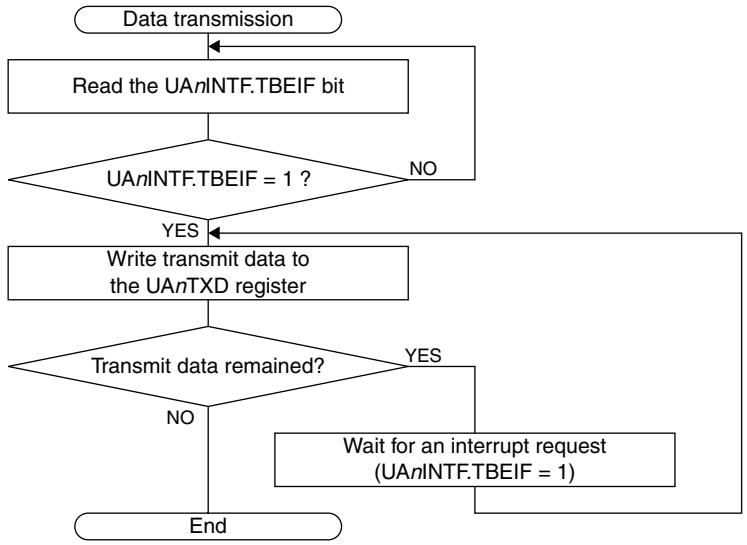


Figure 10.5.2.2 Data Transmission Flowchart

10.5.3 Data Reception

A data receiving procedure and the UART Ch.*n* operations are shown below. Figures 10.5.3.1 and 10.5.3.2 show a timing chart and flowcharts, respectively.

Data receiving procedure (read by one byte)

1. Wait for a UART interrupt when using the interrupt.
2. Check to see if the UA*n*INTF.RB1FIF bit is set to 1 (receive buffer one byte full).
3. Read the received data from the UA*n*RXD register.
4. Repeat Steps 1 to 3 (or 2 and 3) until the end of data reception.

Data receiving procedure (read by two bytes)

1. Wait for a UART interrupt when using the interrupt.
2. Check to see if the UA*n*INTF.RB2FIF bit is set to 1 (receive buffer two bytes full).
3. Read the received data from the UA*n*RXD register twice.
4. Repeat Steps 1 to 3 (or 2 and 3) until the end of data reception.

UART data receiving operations

The UART Ch.*n* starts data receiving operations when a start bit is input to the USIN*n* pin.
After the receive circuit has detected a low level as a start bit, it starts sampling the following data bits and loads the received data into the receive shift register. The UA*n*INTF.RBSY bit is set to 1 when the start bit is detected.
The UA*n*INTF.RBSY bit is cleared to 0 and the receive shift register data is transferred to the receive data buffer at the stop bit receive timing.
The receive data buffer consists of a 2-byte FIFO and receives data until it becomes full. When the receive data buffer receives the first data, it sets the UA*n*INTF.RB1FIF bit to 1 (receive buffer one byte full). If the second data is received without reading the first data, the UA*n*INTF.RB2FIF bit is set to 1 (receive buffer two bytes full).

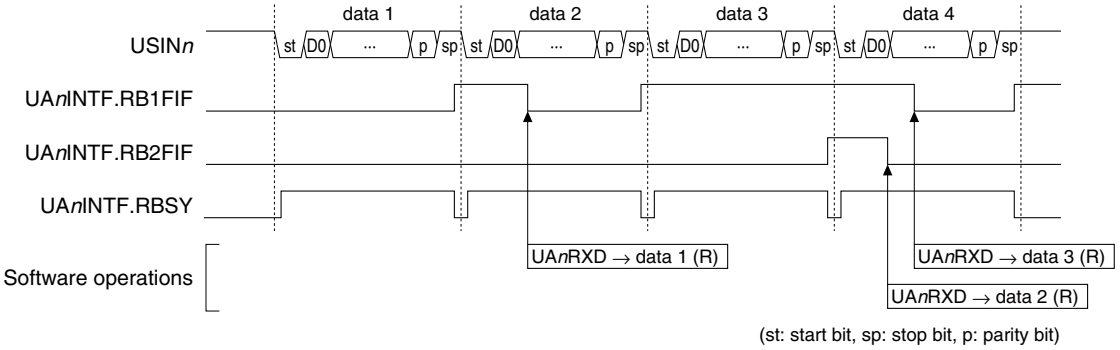


Figure 10.5.3.1 Example of Data Receiving Operations

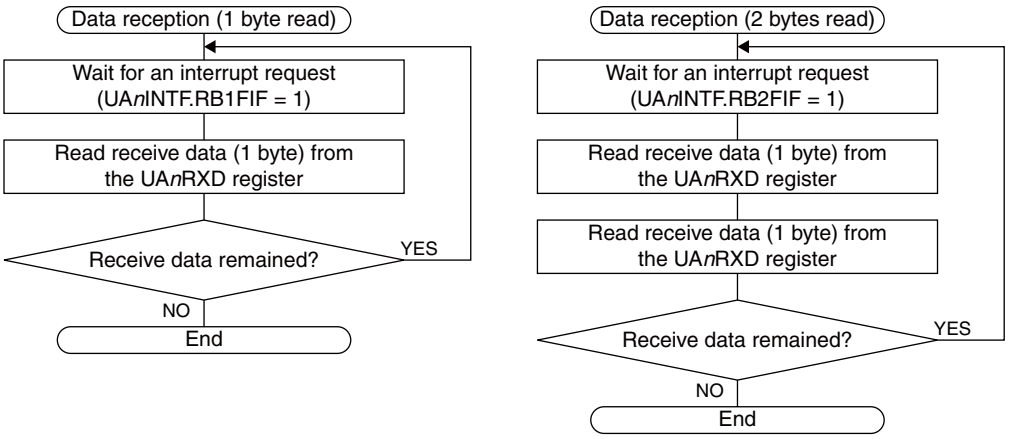


Figure 10.5.3.2 Data Reception Flowcharts

10.5.4 IrDA Interface

This UART includes an RZI modulator/demodulator circuit enabling implementation of IrDA 1.0-compatible infrared communication function simply by adding simple external circuits.
Set the UA*n*MOD.IRMD bit to 1 to use the IrDA interface.
Data transfer control is identical to that for normal interface even if the IrDA interface function is enabled.

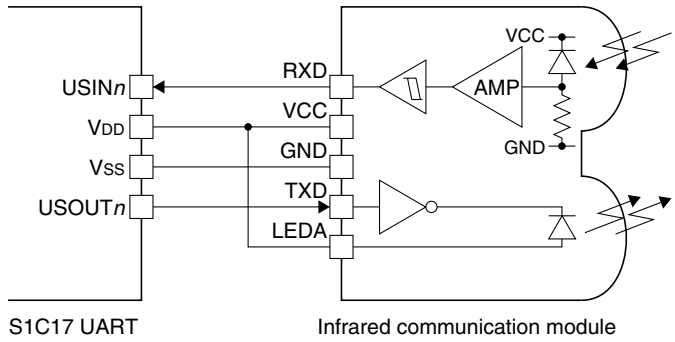


Figure 10.5.4.1 Example of Connections with an Infrared Communication Module

The transmit data output from the UART Ch.*n* transmit shift register is output from the USOUT*n* pin after the low pulse width is converted into 3/16 by the RZI modulator in SIR method and inverted. The USOUT*n* pin output signal can be inverted by setting the UA*n*MOD.INVIRTX bit to 1.

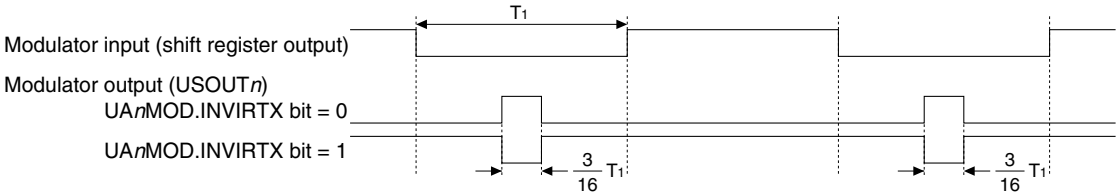


Figure 10.5.4.2 IrDA Transmission Signal Waveform

The received IrDA signal is input to the RZI demodulator and the low pulse width is converted into the normal width before input to the receive shift register. The USIN*n* pin input signal can be inverted prior to being demodulated by setting the UA*n*MOD.INVIRRX bit to 1.

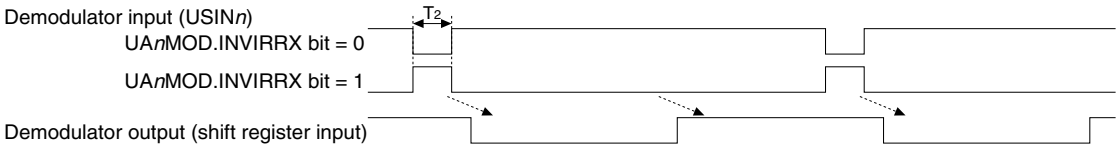


Figure 10.5.4.3 IrDA Receive Signal Waveform

Note: The low pulse width (T_2) of the IrDA signal input must be CLK_UART × 3 cycles or longer.

10.6 Receive Errors

Three different receive errors, framing error, parity error, and overrun error, may be detected while receiving data. Since receive errors are interrupt causes, they can be processed by generating interrupts.

10.6.1 Framing Error

The UART determines loss of sync if a stop bit is not detected (when the stop bit is received as 0) and assumes that a framing error has occurred. The received data that encountered an error is still transferred to the receive data buffer and the UA*n*INTF.FEIF bit (framing error interrupt flag) is set to 1 when the data becomes ready to read from the UA*n*RXD register.

Note: Framing error/parity error interrupt flag set timings
These interrupt flags will be set after the data that encountered an error is transferred to the receive data buffer. Note, however, that the set timing depends on the buffer status at that point.

- When the receive data buffer is empty
  The interrupt flag will be set when the data that encountered an error is transferred to the receive data buffer.

- When the receive data buffer has a one-byte free space
  The interrupt flag will be set when the first data byte already loaded is read out after the data that encountered an error is transferred to the second byte entry of the receive data buffer.

10.6.2 Parity Error

If the parity function is enabled, a parity check is performed when data is received. The UART checks matching between the data received in the shift register and its parity bit, and issues a parity error if the result is a non-match. The received data that encountered an error is still transferred to the receive data buffer and the UA*n*INTF.PEIF bit (parity error interrupt flag) is set to 1 when the data becomes ready to read from the UA*n*RXD register (see the Note on framing error).

10.6.3 Overrun Error

If the receive data buffer is still full (two bytes of received data have not been read) when a data reception to the shift register has completed, an overrun error occurs as the data cannot be transferred to the receive data buffer. When an overrun error occurs, the UA*n*INTF.OEIF bit (overrun error interrupt flag) is set to 1.

10.7 Interrupts

The UART has a function to generate the interrupts shown in Table 10.7.1.

Table 10.7.1 UART Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
End of transmission	UA*n*INTF.TENDIF	When the UA*n*INTF.TBEIF bit = 1 after the stop bit has been sent	Writing 1 or software reset
Framing error	UA*n*INTF.FEIF	Refer to “Receive Errors.”	Writing 1, reading received data that encountered an error, or software reset
Parity error	UA*n*INTF.PEIF	Refer to “Receive Errors.”	Writing 1, reading received data that encountered an error, or software reset
Overrun error	UA*n*INTF.OEIF	Refer to “Receive Errors.”	Writing 1 or software reset
Receive buffer two bytes full	UA*n*INTF.RB2FIF	When the second received data byte is loaded to the receive data buffer in which the first byte is already received	Reading received data or software reset
Receive buffer one byte full	UA*n*INTF.RB1FIF	When the first received data byte is loaded to the emptied receive data buffer	Reading data to empty the receive data buffer or software reset
Transmit buffer empty	UA*n*INTF.TBEIF	When transmit data written to the transmit data buffer is transferred to the shift register	Writing transmit data

The UART provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the “Interrupt Controller” chapter.

10.8 Control Registers

UART Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*CLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the UART operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits select the division ratio of the UART operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of the UART.

Table 10.8.1 Clock Source and Division Ratio Settings

UA*n*CLK.CLKDIV[1:0] bits	UA*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x3	1/8	1/1	1/8	1/1
0x2	1/4		1/4	
0x1	1/2		1/2	
0x0	1/1		1/1	

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The UA*n*CLK register settings can be altered only when the UA*n*CTL.MODEN bit = 0.

UART Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*MOD	15–10	–	0x00	–	R	–
	9	INVIRRX	0	H0	R/W	
	8	INVIRTX	0	H0	R/W	
	7	–	0	–	R	
	6	PUEN	0	H0	R/W	
	5	OUTMD	0	H0	R/W	
	4	IRMD	0	H0	R/W	
	3	CHLN	0	H0	R/W	
	2	PREN	0	H0	R/W	
	1	PRMD	0	H0	R/W	
	0	STPB	0	H0	R/W	

Bits 15–10 Reserved

Bit 9 INVIRRX

This bit enables the USIN*n* input inverting function when the IrDA interface function is enabled.

1 (R/W): Enable input inverting function

0 (R/W): Disable input inverting function

Bit 8 INVIRTX

This bit enables the USOUT*n* output inverting function when the IrDA interface function is enabled.

1 (R/W): Enable output inverting function

0 (R/W): Disable output inverting function

Bit 7 Reserved

Bit 6 PUEN

This bit enables pull-up of the USIN*n* pin.

1 (R/W): Enable pull-up

0 (R/W): Disable pull-up

Bit 5 OUTMD

This bit sets the USOUT*n* pin output mode.
1 (R/W): Open-drain output
0 (R/W): Push-pull output

Bit 4 IRMD

This bit enables the IrDA interface function.
1 (R/W): Enable IrDA interface function
0 (R/W): Disable IrDA interface function

Bit 3 CHLN

This bit sets the data length.
1 (R/W): 8 bits
0 (R/W): 7 bits

Bit 2 PREN

This bit enables the parity function.
1 (R/W): Enable parity function
0 (R/W): Disable parity function

Bit 1 PRMD

This bit selects either odd parity or even parity when using the parity function.
1 (R/W): Odd parity
0 (R/W): Even parity

Bit 0 STPB

This bit sets the stop bit length.
1 (R/W): 2 bits
0 (R/W): 1 bit

Note: The UA*n*MOD register settings can be altered only when the UA*n*CTL.MODEN bit = 0.

UART Ch.*n* Baud–Rate Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*BR	15–12	–	0x0	–	R	–
	11–8	FMD[3:0]	0x0	H0	R/W	
	7–0	BRT[7:0]	0x00	H0	R/W	

Bits 15–12 Reserved

Bits 11–8 FMD[3:0]
Bits 7–0 BRT[7:0]

These bits set the UART transfer rate. For more information, refer to "Baud Rate Generator."

Note: The UA*n*BR register settings can be altered only when the UA*n*CTL.MODEN bit = 0.

UART Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*CTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	SFTRST	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 SFTRST

This bit issues software reset to the UART.

1 (W): Issue software reset
0 (W): Ineffective
1 (R): Software reset is executing.
0 (R): Software reset has finished. (During normal operation)

Setting this bit resets the UART transmit/receive control circuit and interrupt flags. This bit is automatically cleared after the reset processing has finished.

Bit 0 MODEN

This bit enables the UART operations.

1 (R/W): Enable UART operations (The operating clock is supplied.)
0 (R/W): Disable UART operations (The operating clock is stopped.)

Note: If the UA*n*CTL.MODEN bit is altered from 1 to 0 during sending/receiving data, the data being sent/received cannot be guaranteed. When setting the UA*n*CTL.MODEN bit to 1 again after that, be sure to write 1 to the UA*n*CTL.SFTRST bit as well.

UART Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*TXD	15–8	–	0x00	–	R	–
	7–0	TXD[7:0]	0x00	H0	R/W	

Bits 15–8 Reserved

Bits 7–0 TXD[7:0]

Data can be written to the transmit data buffer through these bits. Make sure the UA*n*INTF.TBEIF bit is set to 1 before writing data.

UART Ch.*n* Receive Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*RXD	15–8	–	0x00	–	R	–
	7–0	RXD[7:0]	0x00	H0	R	

Bits 15–8 Reserved

Bits 7–0 RXD[7:0]

The receive data buffer can be read through these bits. The receive data buffer consists of a 2-byte FIFO, and older received data is read first.

UART Ch.*n* Status and Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*INTF	15–10	–	0x00	–	R	–
	9	RBSY	0	H0/S0	R	
	8	TBSY	0	H0/S0	R	
	7	–	0	–	R	
	6	TENDIF	0	H0/S0	R/W	Cleared by writing 1.
	5	FEIF	0	H0/S0	R/W	Cleared by writing 1 or reading the UA*n*RXD register.
	4	PEIF	0	H0/S0	R/W	
	3	OEIF	0	H0/S0	R/W	Cleared by writing 1.
	2	RB2FIF	0	H0/S0	R	Cleared by reading the UA*n*RXD register.
	1	RB1FIF	0	H0/S0	R	
	0	TBEIF	1	H0/S0	R	Cleared by writing to the UA*n*TXD register.

Bits 15–10 Reserved

Bit 9 RBSY

This bit indicates the receiving status. (See Figure 10.5.3.1.)

1 (R): During receiving

0 (R): Idle

Bit 8 TBSY

This bit indicates the sending status. (See Figure 10.5.2.1.)

1 (R): During sending

0 (R): Idle

Bit 7 Reserved

Bit 6 TENDIF
Bit 5 FEIF
Bit 4 PEIF
Bit 3 OEIF
Bit 2 RB2FIF
Bit 1 RB1FIF
Bit 0 TBEIF

These bits indicate the UART interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag

0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:

UA*n*INTF.TENDIF bit: End-of-transmission interrupt

UA*n*INTF.FEIF bit: Framing error interrupt

UA*n*INTF.PEIF bit: Parity error interrupt

UA*n*INTF.OEIF bit: Overrun error interrupt

UA*n*INTF.RB2FIF bit: Receive buffer two bytes full interrupt

UA*n*INTF.RB1FIF bit: Receive buffer one byte full interrupt

UA*n*INTF.TBEIF bit: Transmit buffer empty interrupt

UART Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UA*n*INTE	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6	TENDIE	0	H0	R/W	
	5	FEIE	0	H0	R/W	
	4	PEIE	0	H0	R/W	
	3	OEIE	0	H0	R/W	
	2	RB2FIE	0	H0	R/W	
	1	RB1FIE	0	H0	R/W	
	0	TBEIE	0	H0	R/W	

Bits 15–7 Reserved

Bit 6 TENDIE
Bit 5 FEIE
Bit 4 PEIE
Bit 3 OEIE
Bit 2 RB2FIE
Bit 1 RB1FIE
Bit 0 TBEIE

These bits enable UART interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
UA*n*INTE.TENDIE bit: End-of-transmission interrupt
UA*n*INTE.FEIE bit: Framing error interrupt
UA*n*INTE.PEIE bit: Parity error interrupt
UA*n*INTE.OEIE bit: Overrun error interrupt
UA*n*INTE.RB2FIE bit: Receive buffer two bytes full interrupt
UA*n*INTE.RB1FIE bit: Receive buffer one byte full interrupt
UA*n*INTE.TBEIE bit: Transmit buffer empty interrupt

11 Synchronous Serial Interface (SPI)

11.1 Overview

SPI is a synchronous serial interface. The features of the SPI are listed below.

- Supports both master and slave modes.
- Data length: 8 bits
- Either MSB first or LSB first can be selected for the data format.
- Clock phase and polarity are configurable.
- Supports full-duplex communications.
- Includes separated transmit data buffer and receive data buffer registers.
- Can generate receive buffer full, transmit buffer empty, and end of transmission interrupts.
- Master mode allows use of a 16-bit timer to set baud rate.
- Slave mode is capable of being operated with the external input clock SPICLK*n* only.
- Slave mode is capable of being operated in SLEEP mode allowing wake-up by an SPI interrupt.
- Input pins can be pulled up/down with an internal resistor.

Figure 11.1.1 shows the SPI configuration.

Channel configuration in this IC
- Number of channels: 3 channels (Ch.0, Ch.1, and Ch.2)
- Internal clock input: Ch.0 ← 16-bit timer Ch.1
  Ch.1 ← 16-bit timer Ch.2
  Ch.2 ← 16-bit timer Ch.3

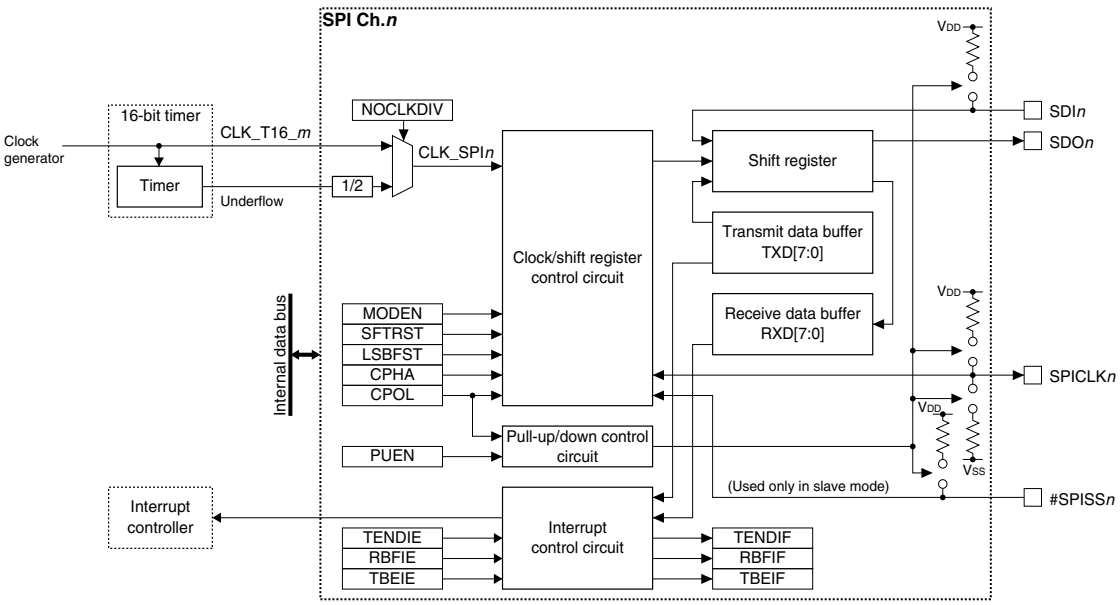


Figure 11.1.1 SPI Configuration

11.2 Input/Output Pins and External Connections

11.2.1 List of Input/Output Pins

Table 11.2.1.1 lists the SPI pins.

Table 11.2.1.1 List of SPI Pins

Pin name	I/O*	Initial status*	Function
SDI*n*	I	I (Hi-Z)	SPI Ch.*n* data input pin
SDO*n*	O or Hi-Z	Hi-Z	SPI Ch.*n* data output pin
SPICLK*n*	I or O	I (Hi-Z)	SPI Ch.*n* external clock input/output pin
#SPISS*n*	I	I (Hi-Z)	SPI Ch.*n* slave select signal input pin

* Indicates the status when the pin is configured for the SPI.

If the port is shared with the SPI pin and other functions, the SPI input/output function must be assigned to the port before activating the SPI. For more information, refer to the "I/O Ports" chapter.

11.2.2 External Connections

The SPI operates in master mode or slave mode. Figures 11.2.2.1 and 11.2.2.2 show connection diagrams between the SPI in each mode and external SPI devices.

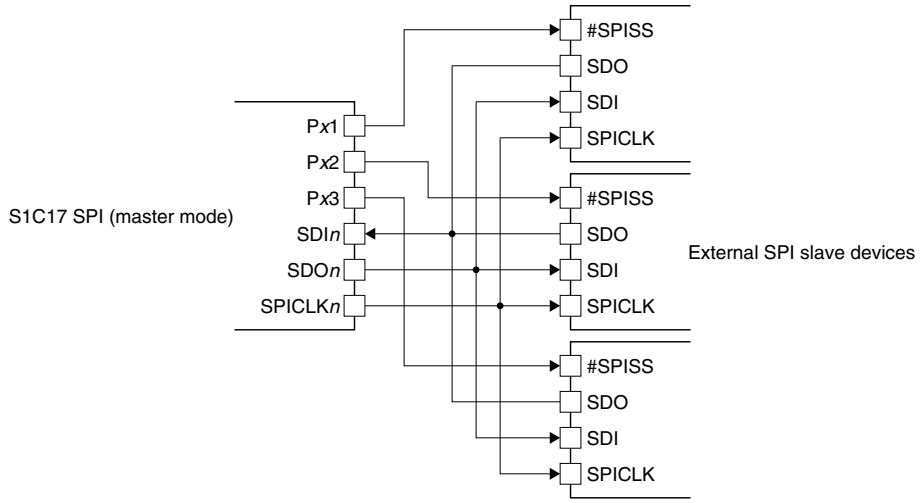


Figure 11.2.2.1 Connections between SPI in Master Mode and External SPI Slave Devices

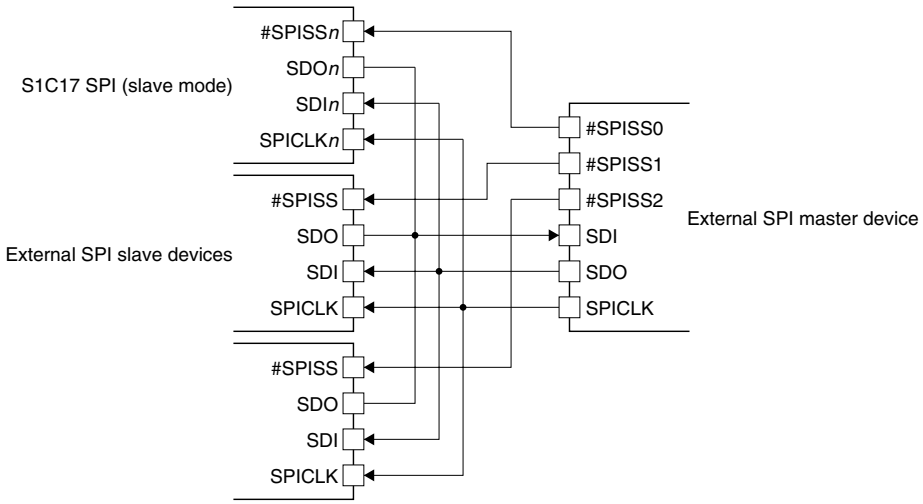


Figure 11.2.2.2 Connections between SPI in Slave Mode and External SPI Master Device

11.2.3 Pin Functions in Master Mode and Slave Mode

The pin functions are changed according to the master or slave mode selection. The differences in pin functions between the modes are shown in Table 11.2.3.1.

Table 11.2.3.1 Pin Function Differences between Modes

Pin	Function in master mode	Function in slave mode
SDI*n*	Always placed into input state.	
SDO*n*	Always placed into output state.	This pin is placed into output state while a low level is applied to the #SPISS*n* pin or placed into Hi-Z state while a high level is applied to the #SPISS*n* pin.
SPICLK*n*	Outputs the SPI clock to external devices. Output clock polarity and phase can be configured if necessary.	Inputs an external SPI clock. Clock polarity and phase can be designated according to the input clock.
#SPISS*n*	Not used. This input function is not required to be assigned to the port. To output the slave select signal in master mode, use a general-purpose I/O port function.	Applying a low level to the #SPISS*n* pin enables the SPI to transmit/receive data. While a high level is applied to this pin, the SPI is not selected as a slave device. Data input to the SDI*n* pin and the clock input to the SPICLK*n* pin are ignored. When a high level is applied, the transmit/receive bit count is cleared to 0 and the already received bits are discarded.

11.2.4 Input Pin Pull-Up/Pull-Down Function

The SPI input pins (SDI*n* in master mode or SDI*n*, SPICLK*n*, and #SPISS*n* pins in slave mode) have a pull-up or pull-down function as shown in Table 11.2.4.1. This function is enabled by setting the SPI*n*MOD.PUEN bit to 1.

Table 11.2.4.1 Pull-Up or Pull-Down of Input Pins

Pin	Master mode	Slave mode
SDI*n*	Pull-up	Pull-up
SPICLK*n*	–	SPI*n*MOD.CPOL bit = 1: Pull-up SPI*n*MOD.CPOL bit = 0: Pull-down
#SPISS*n*	–	Pull-up

11.3 Clock Settings

11.3.1 SPI Operating Clock

Operating clock in master mode

In master mode, the SPI operating clock is supplied from the 16-bit timer. The following two options are provided for the clock configuration.

Use the 16-bit timer operating clock without dividing

By setting the SPI*n*MOD.NOCLKDIV bit to 1, the operating clock CLK_T16_*m*, which is configured by selecting a clock source and a division ratio, for the 16-bit timer channel corresponding to the SPI channel is input to the SPI as CLK_SPI*n*. Since this clock is also used as the SPI clock SPICLK*n* without changing, the CLK_SPI*n* frequency becomes the baud rate.

To supply CLK_SPI*n* to the SPI, the 16-bit timer clock source must be enabled in the clock generator. Also the T16_*m*CTL.MODEN bit of the corresponding 16-bit timer channel must be set to 1. It does not matter how the T16_*m*CTL.PRUN bit is set (1 or 0).

When setting this mode, the timer function of the corresponding 16-bit timer channel may be used for another purpose.

Use the 16-bit timer as a baud rate generator

By setting the SPI*n*MOD.NOCLKDIV bit to 0, the SPI input the underflow signal generated by the corresponding 16-bit timer channel and converts it to the SPICLK*n*. The 16-bit timer must be run with an appropriate reload data set. The SPICLK*n* frequency (baud rate) and the 16-bit timer reload data are calculated by the equations shown below.

$$f_{SPICLK} = \frac{f_{CLK\_SPI}}{2 \times (RLD + 1)} \qquad RLD = \frac{f_{CLK\_SPI}}{f_{SPICLK} \times 2} - 1 \qquad \text{(Eq. 11.1)}$$

Where

f_{SPICLK}: SPICLK*n* frequency [Hz] (= baud rate [bps])
f_{CLK_SPI}: SPI operating clock frequency [Hz]
RLD: 16-bit timer reload data value

For controlling the 16-bit timer, refer to the "16-bit Timers" chapter.

Operating clock in slave mode

The SPI set in slave mode operates with the clock supplied from the external SPI master to the SPICLK*n* pin. The 16-bit timer channel (including the clock source selector and the divider) corresponding to the SPI channel is not used. Furthermore, the SPI*n*MOD.NOCLKDIV bit setting becomes ineffective.

The SPI keeps operating using the clock supplied from the external SPI master even if all the internal clocks halt during SLEEP mode, so the SPI can receive data and can generate receive buffer full interrupts.

11.3.2 Clock Supply in DEBUG Mode

In master mode, the operating clock supply during DEBUG mode should be controlled using the T16_*m*CLK.DBRUN bit.

The CLK_T16_*m* supply to the SPI Ch.*n* is suspended when the CPU enters DEBUG mode if the T16_*m*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_T16_*m* supply resumes. Although the SPI Ch.*n* stops operating when the CLK_T16_*m* supply is suspended, the output pins and registers retain the status before DEBUG mode was entered. If the T16_*m*CLK.DBRUN bit = 1, the CLK_T16_*m* supply is not suspended and the SPI Ch.*n* will keep operating in DEBUG mode.

The SPI in slave mode operates with the external SPI master clock input from the SPICLK*n* pin regardless of whether the CPU is placed into DEBUG mode or normal mode.

11.3.3 SPI Clock (SPICLK*n*) Phase and Polarity

The SPICLK*n* phase and polarity can be configured separately using the SPI*n*MOD.CPHA bit and the SPI*n*MOD.CPOL bit, respectively. Figure 11.3.3.1 shows the clock waveform and data input/output timing in each setting.

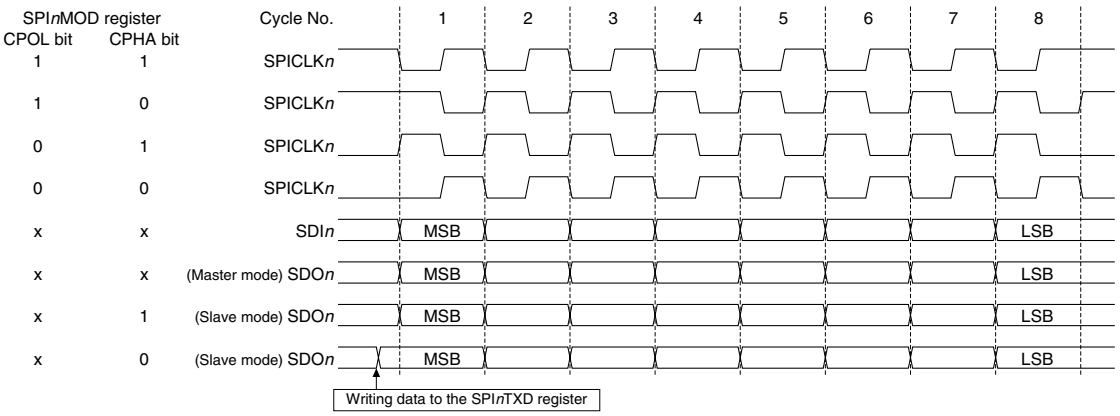


Figure 11.3.3.1 SPI Clock Phase and Polarity (SPI*n*MOD.LSBFST bit = 0)

11.4 Data Format

The data length is fixed at eight bits in the SPI. The input/output permutation is configurable to MSB first or LSB first using the SPI*n*MOD.LSBFST bit. Figure 11.4.1 shows a data format example when the SPI*n*MOD.CPOL bit = 0 and the SPI*n*MOD.CPHA bit = 0.

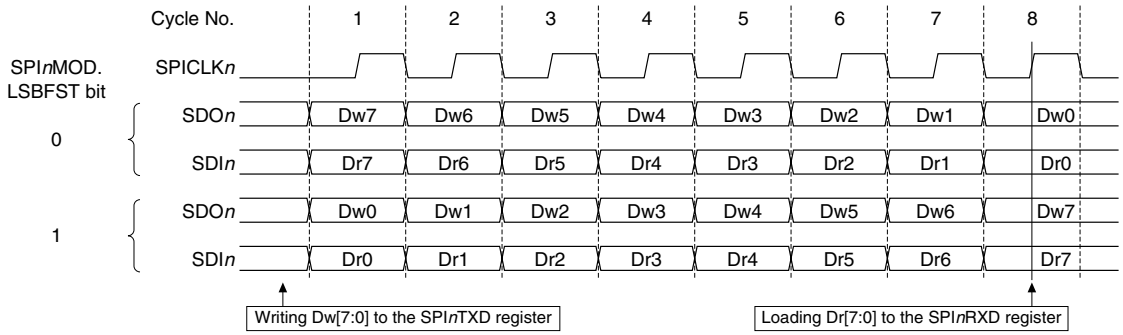


Figure 11.4.1 Data Format Selection Using the SPI*n*MOD.LSBFST Bit
(SPI*n*MOD.CPOL bit = 0, SPI*n*MOD.CPHA bit = 0)

11.5 Operations

11.5.1 Initialization

The SPI Ch.*n* should be initialized with the procedure shown below.

1. <Master mode only> Generate a clock by controlling the 16-bit timer and supply it to the SPI Ch.*n*.
2. Configure the following SPI*n*MOD register bits:
   - SPI*n*MOD.PUEN bit (Enable input pin pull-up/down)
   - SPI*n*MOD.NOCLKDIV bit (Select master mode operating clock)
   - SPI*n*MOD.LSBFST bit (Select MSB first/LSB first)
   - SPI*n*MOD.CPHA bit (Select clock phase)
   - SPI*n*MOD.CPOL bit (Select clock polarity)
   - SPI*n*MOD.MST bit (Select master/slave mode)
3. Assign the SPI Ch.*n* input/output function to the ports. (Refer to the "I/O Ports" chapter.)
4. Set the following SPI*n*CTL register bits:
   - Set the SPI*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the SPI*n*CTL.MODEN bit to 1. (Enable SPI Ch.*n* operations)
5. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the SPI*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the SPI*n*INTE register to 1. * (Enable interrupts)

   * The initial value of the SPI*n*INTF.TBEIF bit is 1, therefore, an interrupt will occur immediately after the SPI*n*INTE.TBEIE bit is set to 1.

11.5.2 Data Transmission in Master Mode

A data sending procedure and operations in master mode are shown below. Figures 11.5.2.1 and 11.5.2.2 show a timing chart and a flowchart, respectively.

Data sending procedure

1. Assert the slave select signal by controlling the general-purpose output port (if necessary).
2. Check to see if the SPI*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
3. Write transmit data to the SPI*n*TXD register.
4. Wait for an SPI interrupt when using the interrupt.

5. Repeat Steps 2 to 4 (or 2 and 3) until the end of transmit data.
6. Negate the slave select signal by controlling the general-purpose output port (if necessary).

Data sending operations

The SPI Ch.*n* starts data sending operations when transmit data is written to the SPI*n*TXD register.
The transmit data in the SPI*n*TXD register is automatically transferred to the shift register and the SPI*n*INTF.TBEIF bit is set to 1. If the SPI*n*INTE.TBEIE bit = 1 (transmit buffer empty interrupt enabled), a transmit buffer empty interrupt occurs at the same time.
The SPICLK*n* pin outputs eight clocks and the transmit data bits are output in sequence from the SDO*n* pin in sync with this clock.

Even if the clock is being output from the SPICLK*n* pin, the next transmit data can be written to the SPI*n*TXD register after making sure the SPI*n*INTF.TBEIF bit is set to 1.

If transmit data has not been written to the SPI*n*TXD register after the eighth clock is output from the SPICLK*n* pin, the clock output halts and the SPI*n*INTF.TENDIF bit is set to 1. At the same time the SPI issues an end-of-transmission interrupt request if the SPI*n*INTE.TENDIE bit = 1.

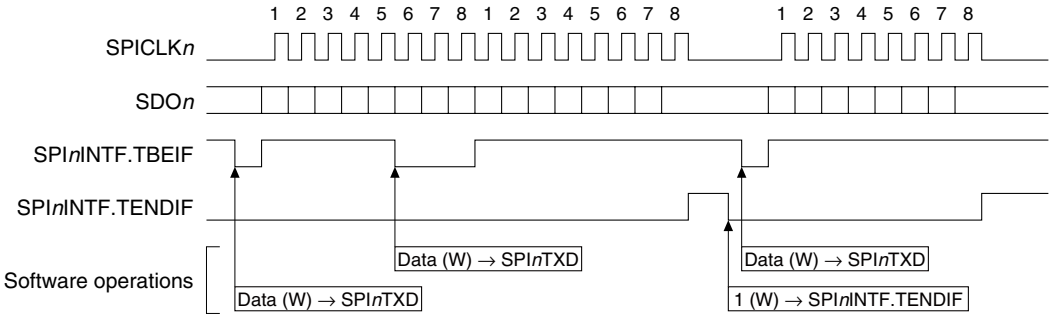


Figure 11.5.2.1 Example of Data Sending Operations in Master Mode

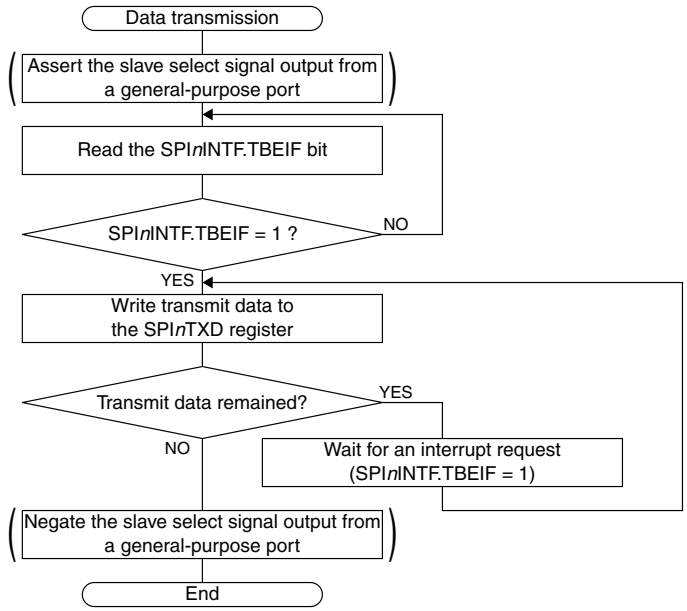


Figure 11.5.2.2 Data Transmission Flowchart in Master Mode

11.5.3 Data Reception in Master Mode

A data receiving procedure and operations in master mode are shown below. Figures 11.5.3.1 and 11.5.3.2 show a timing chart and flowcharts, respectively.

Data receiving procedure

1. Assert the slave select signal by controlling the general-purpose output port (if necessary).
2. Check to see if the SPI*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
3. Write dummy data (or transmit data) to the SPI*n*TXD register.
4. Wait for a transmit buffer empty interrupt (SPI*n*INTF.TBEIF bit = 1).
5. Write dummy data (or transmit data) to the SPI*n*TXD register.
6. Wait for a receive buffer full interrupt (SPI*n*INTF.RBFIF bit = 1).
7. Read the received data from the SPI*n*RXD register.
8. Repeat Steps 5 to 7 until the end of data reception.
9. Negate the slave select signal by controlling the general-purpose output port (if necessary).

Note: To perform continuous data reception without stopping SPICLK*n*, Steps 7 and 5 operations must be completed within seven SPICLK*n* cycles after Step 6.

Data receiving operations

The SPI Ch.*n* starts data receiving operations simultaneously with data sending operations when transmit data (may be dummy data if data transmission is not required) is written to the SPI*n*TXD register.
The SPICLK*n* pin outputs eight clocks. The transmit data bits are output in sequence from the SDO*n* pin in sync with this clock and the receive data bits input from the SDI*n* pin are shifted into the shift register.
When the eighth clock is output from the SPICLK*n* pin and 8-bit receive data is shifted into the shift register, the received data is transferred to the receive data buffer and the SPI*n*INTF.RBFIF bit is set to 1. At the same time the SPI issues a receive buffer full interrupt request if the SPI*n*INTE.RBFIE bit = 1. After that, the received data in the receive data buffer can be read through the SPI*n*RXD register.

Note: If 8-bit data is received when the SPI*n*INTF.RBFIF bit is set to 1, the SPI*n*RXD register is overwritten with the newly received 8-bit data and the previously received data is lost. There is no flag provided for indicating a loss of data.

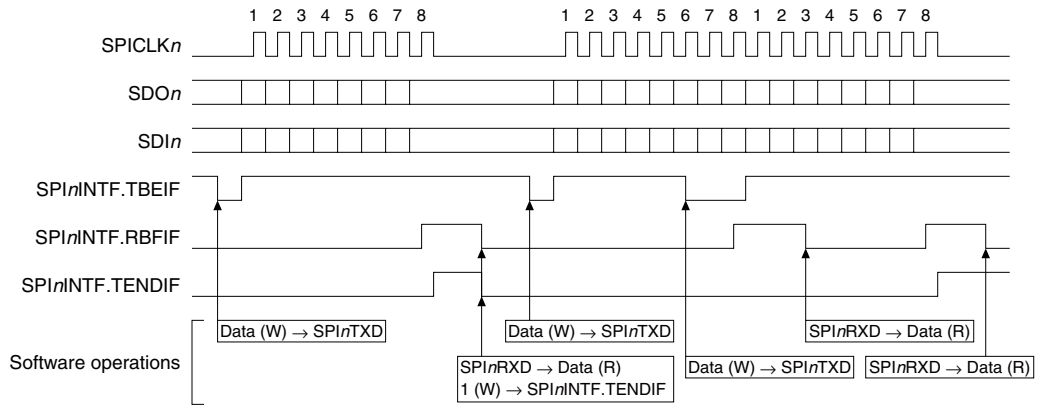


Figure 11.5.3.1 Example of Data Receiving Operations in Master Mode

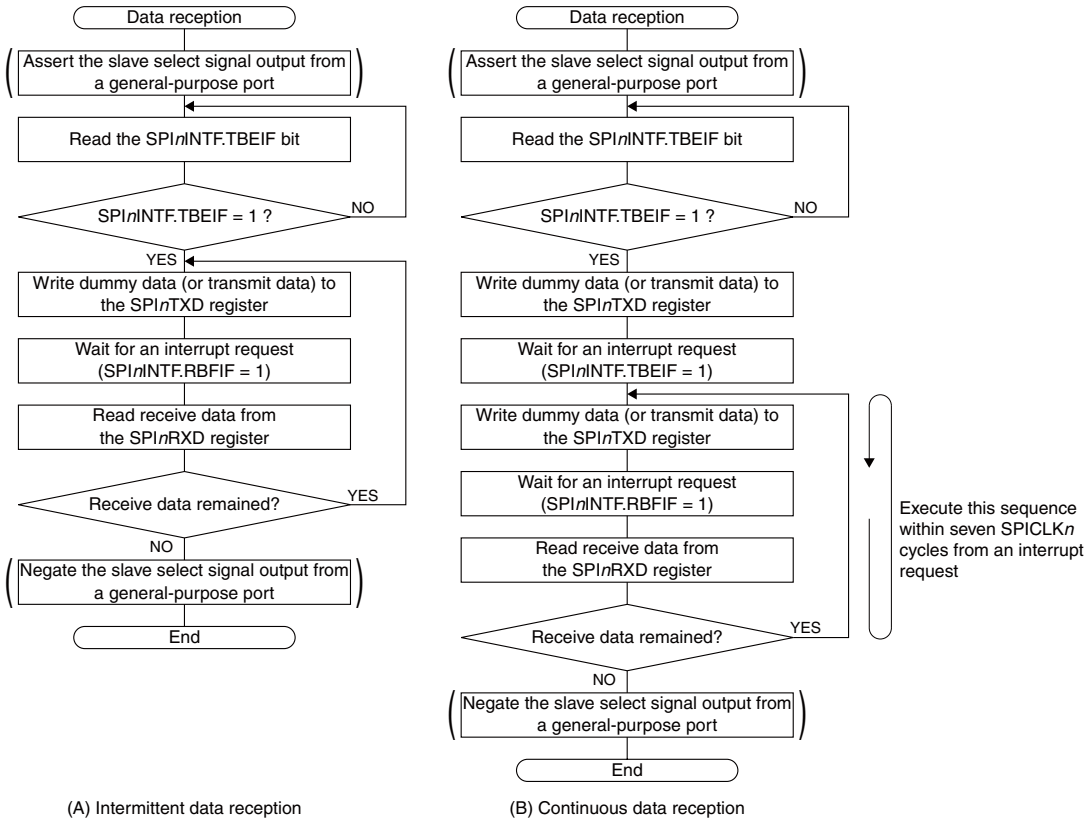


Figure 11.5.3.2 Data Reception Flowcharts in Master Mode

11.5.4 Terminating Data Transfer in Master Mode

A procedure to terminate data transfer in master mode is shown below.

1. Wait for an end-of-transmission interrupt (SPI*n*INTF.TENDIF bit = 1).
2. Set the SPI*n*CTL.MODEN bit to 0 to disable the SPI Ch.*n* operations.
3. Stop the 16-bit timer to disable the clock supply to the SPI Ch.*n*.

11.5.5 Data Transfer in Slave Mode

A data sending/receiving procedure and operations in slave mode are shown below. Figures 11.5.5.1 and 11.5.5.2 show a timing chart and flowcharts, respectively.

Data sending procedure

1. Check to see if the SPI*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
2. Write transmit data to the SPI*n*TXD register.
3. Wait for a transmit buffer empty interrupt (SPI*n*INTF.TBEIF bit = 1).
4. Repeat Steps 2 and 3 until the end of transmit data.

Note: Transmit data must be written to the SPI*n*TXD register after the SPI*n*INTF.TBEIF bit is set to 1 until the seventh SPICLK*n* clock is output. If no transmit data is written during this period, the data bits input from the SDI*n* pin are shifted and output from the SDO*n* pin without being modified.

Data receiving procedure

1. Wait for a receive buffer full interrupt (SPI*n*INTF.RBFIF bit = 1).
2. Read the received data from the SPI*n*RXD register.
3. Repeat Steps 1 and 2 until the end of data reception.

Data transfer operations

The following shows the slave mode operations different from master mode:

- Slave mode operates with the SPI clock supplied from the external SPI master to the SPICLK*n* pin.
  The data transfer rate is determined by the SPICLK*n* frequency. It is not necessary to control the 16-bit timer.
- The SPI can operate as a slave device only when the slave select signal input from the external SPI master to the #SPISS*n* pin is set to the active (low) level.
  If #SPISS*n* = high, the software transfer control, the SPICLK*n* pin input, and the SDI*n* pin input are all ineffective. If the #SPISS*n* signal goes high during data transfer, the transfer bit counter is cleared and data in the shift register is discarded.
- Slave mode starts data transfer when SPICLK*n* is input from the external SPI master after the #SPISS*n* signal is asserted. Writing transmit data is not a trigger to start data transfer. Therefore, it is not necessary to write dummy data to the transmit data buffer when performing data reception only.
- Data transmission/reception can be performed even in SLEEP mode, it makes it possible to wake the CPU up using an SPI interrupt.

Other operations are the same as master mode.

Notes: • If 8-bit data is received when the SPI*n*INTF.RBFIF bit is set to 1, the SPI*n*RXD register is overwritten with the newly received 8-bit data and the previously received data is lost. There is no flag provided for indicating a loss of data.

• When the clock for the first bit is input from the SPICLK*n* pin, the SPI starts sending the 8-bit data currently stored in the shift register even if the SPI*n*INTF.TBEIF bit is set to 1.

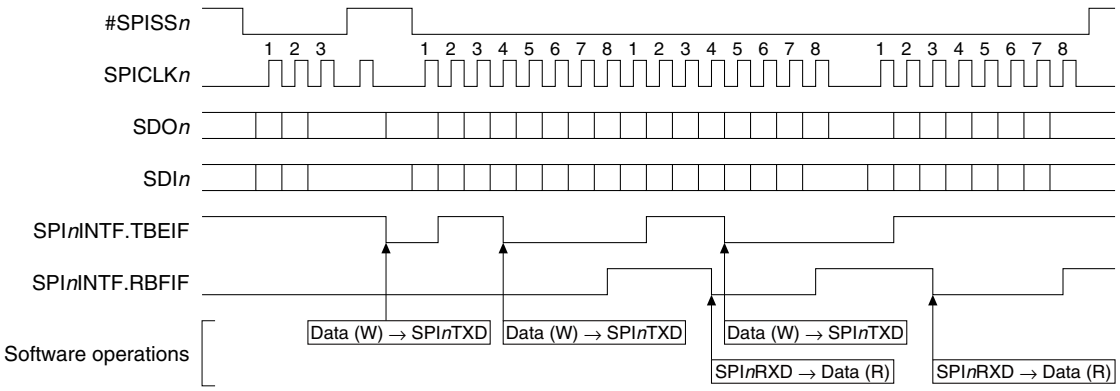


Figure 11.5.5.1 Example of Data Transfer Operations in Slave Mode

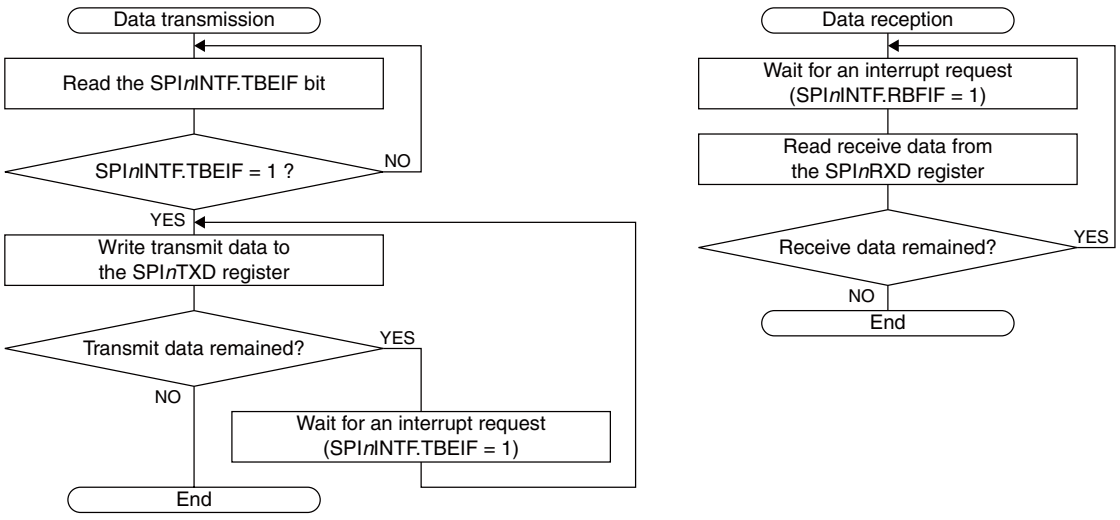


Figure 11.5.5.2 Data Transfer Flowcharts in Slave Mode

11.5.6 Terminating Data Transfer in Slave Mode

A procedure to terminate data transfer in slave mode is shown below.

1. Wait for an end-of-transmission interrupt (SPI*n*INTF.TENDIF bit = 1). Or determine end of transfer via the received data.
2. Set the SPI*n*CTL.MODEN bit to 0 to disable the SPI Ch.*n* operations.

11.6 Interrupts

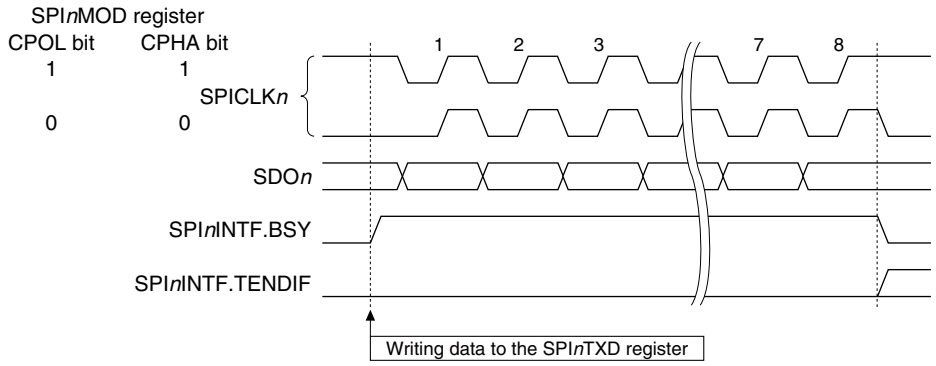
The SPI has a function to generate the interrupts shown in Table 11.6.1.

Table 11.6.1 SPI Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
End of transmission	SPI*n*INTF.TENDIF	When the SPI*n*INTF.TBEIF bit = 1 after the eighth data bit has been sent	Writing 1
Receive buffer full	SPI*n*INTF.RBFIF	When the eighth data bit is received and the received data is transferred from the shift register to the received data buffer	Reading the SPI*n*RXD register
Transmit buffer empty	SPI*n*INTF.TBEIF	When transmit data written to the transmit data buffer is transferred to the shift register	Writing to the SPI*n*TXD register

The SPI provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.
The SPI*n*INTF register also contains the BSY bit that indicates the SPI operating status.
Figure 11.6.1 shows the SPI*n*INTF.BSY and SPI*n*INTF.TENDIF bit set timings.

Master mode



Slave mode

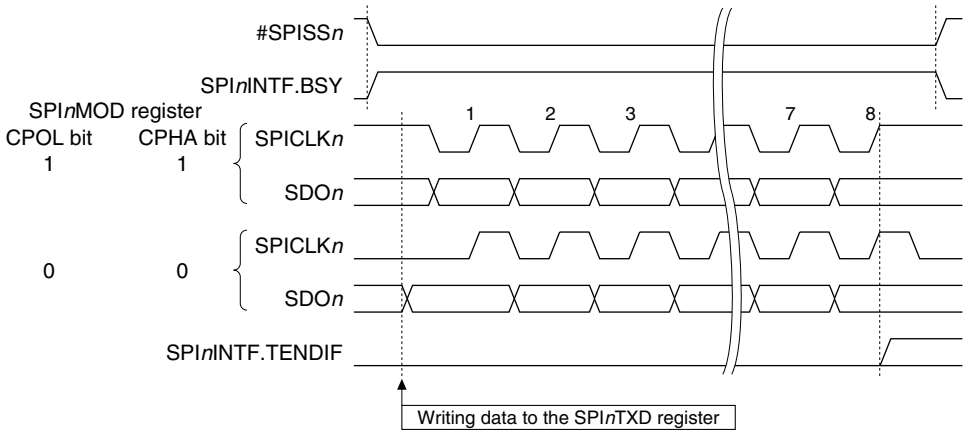


Figure 11.6.1 SPI*n*INTF.BSY and SPI*n*INTF.TENDIF Bit Set Timings

11.7 Control Registers

SPI Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*MOD	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5	PUEN	0	H0	R/W	
	4	NOCLKDIV	0	H0	R/W	
	3	LSBFST	0	H0	R/W	
	2	CPHA	0	H0	R/W	
	1	CPOL	0	H0	R/W	
	0	MST	0	H0	R/W	

Bits 15–6 Reserved

Bit 5 PUEN

This bit enables pull-up/down of the input pins.
1 (R/W): Enable pull-up/down
0 (R/W): Disable pull-up/down

For more information, refer to "Input Pin Pull-Up/Pull-Down Function."

Bit 4 NOCLKDIV

This bit selects SPICLK*n* in master mode. This setting is ineffective in slave mode.
1 (R/W): SPICLK*n* frequency = CLK_SPI*n* frequency (= 16-bit timer operating clock frequency)
0 (R/W): SPICLK*n* frequency = 16-bit timer output frequency / 2

For more information, refer to "SPI Operating Clock."

Bit 3 LSBFST

This bit configures the data format (input/output permutation).
1 (R/W): LSB first
0 (R/W): MSB first

Bit 2 CPHA
Bit 1 CPOL

These bits set the SPI clock phase and polarity. For more information, refer to "SPI Clock (SPICLK*n*) Phase and Polarity."

Bit 0 MST

This bit sets the SPI operating mode (master mode or slave mode).
1 (R/W): Master mode
0 (R/W): Slave mode

Note: The SPI*n*MOD register settings can be altered only when the SPI*n*CTL.MODEN bit = 0.

SPI Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*CTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	SFTRST	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 SFTRST

This bit issues software reset to the SPI.

1 (W): Issue software reset
0 (W): Ineffective
1 (R): Software reset is executing.
0 (R): Software reset has finished. (During normal operation)

Setting this bit resets the SPI shift register and transfer bit counter. This bit is automatically cleared after the reset processing has finished.

Bit 0 MODEN

This bit enables the SPI operations.

1 (R/W): Enable SPI operations (In master mode, the operating clock is supplied.)
0 (R/W): Disable SPI operations (In master mode, the operating clock is stopped.)

Note: If the SPI*n*CTL.MODEN bit is altered from 1 to 0 during sending/receiving data, the data being sent/received cannot be guaranteed. When setting the SPI*n*CTL.MODEN bit to 1 again after that, be sure to write 1 to the SPI*n*CTL.SFTRST bit as well.

SPI Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*TXD	15–8	–	0x00	–	R	–
	7–0	TXD[7:0]	0x00	H0	R/W	

Bits 15–8 Reserved

Bits 7–0 TXD[7:0]

Data can be written to the transmit data buffer through these bits.
In master mode, writing to these bits starts data transfer.
Transmit data can be written when the SPI*n*INTF.TBEIF bit = 1 regardless of whether data is being output from the SDO*n* pin or not.

Note: Be sure to avoid writing to the SPI*n*TXD register when the SPI*n*INTF.TBEIF bit = 0. Otherwise, transfer data cannot be guaranteed.

SPI Ch.*n* Receive Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*RXD	15–8	–	0x00	–	R	–
	7–0	RXD[7:0]	0x00	H0	R	

Bits 15–8 Reserved

Bits 7–0 RXD[7:0]

The receive data buffer can be read through these bits. Received data can be read when the SPI*n*INTF.RBFIF bit = 1 regardless of whether data is being input from the SDI*n* pin or not.

Note: The SPI*n*RXD.RXD[7:0] bits are cleared to 0x00 when 1 is written to the SPI*n*CTL.MODEN bit or the SPI*n*CTL.SFTRST bit.

SPI Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*INTF	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3	BSY	0	H0	R	
	2	TENDIF	0	H0/S0	R/W	Cleared by writing 1.
	1	RBFIF	0	H0/S0	R	Cleared by reading the SPI*n*RXD register.
	0	TBEIF	1	H0/S0	R	Cleared by writing to the SPI*n*TXD register.

Bits 15–4 Reserved

Bit 3 BSY

This bit indicates the SPI operating status.

1 (R): Transmit/receive busy (master mode), #SPISS*n* = Low level (slave mode)

0 (R): Idle

Bit 2 TENDIF

Bit 1 RBFIF

Bit 0 TBEIF

These bits indicate the SPI interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag (TENDIF)

0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:

SPI*n*INTF.TENDIF bit: End-of-transmission interrupt

SPI*n*INTF.RBFIF bit: Receive buffer full interrupt

SPI*n*INTF.TBEIF bit: Transmit buffer empty interrupt

SPI Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*INTE	15–8	–	0x00	–	R	–
	7–3	–	0x00	–	R	
	2	TENDIE	0	H0	R/W	
	1	RBFIE	0	H0	R/W	
	0	TBEIE	0	H0	R/W	

Bits 15–3 Reserved

Bit 2 TENDIE

Bit 1 RBFIE

Bit 0 TBEIE

These bits enable SPI interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:

SPI*n*INTE.TENDIE bit: End-of-transmission interrupt

SPI*n*INTE.RBFIE bit: Receive buffer full interrupt

SPI*n*INTE.TBEIE bit: Transmit buffer empty interrupt

12 I²C (I2C)

12.1 Overview

The I2C is a subset of the I²C bus interface. The features of the I2C are listed below.

- Functions as an I²C bus master (single master) or a slave device.
- Supports standard mode (up to 100 kbit/s) and fast mode (up to 400 kbit/s).
- Supports 7-bit and 10-bit address modes.
- Supports clock stretching.
- Includes a baud rate generator for generating the clock in master mode.
- No clock source is required to run the I2C in slave mode, as it can run with the I²C bus signals only.
- Slave mode is capable of being operated in SLEEP mode allowing wake-up by an interrupt when an address match is detected.
- Master mode supports automatic bus clear sending function.
- Can generate receive buffer full, transmit buffer empty, and other interrupts.

Figure 12.1.1 shows the I2C configuration.

Channel configuration in this IC
- 1 channel (Ch.0)

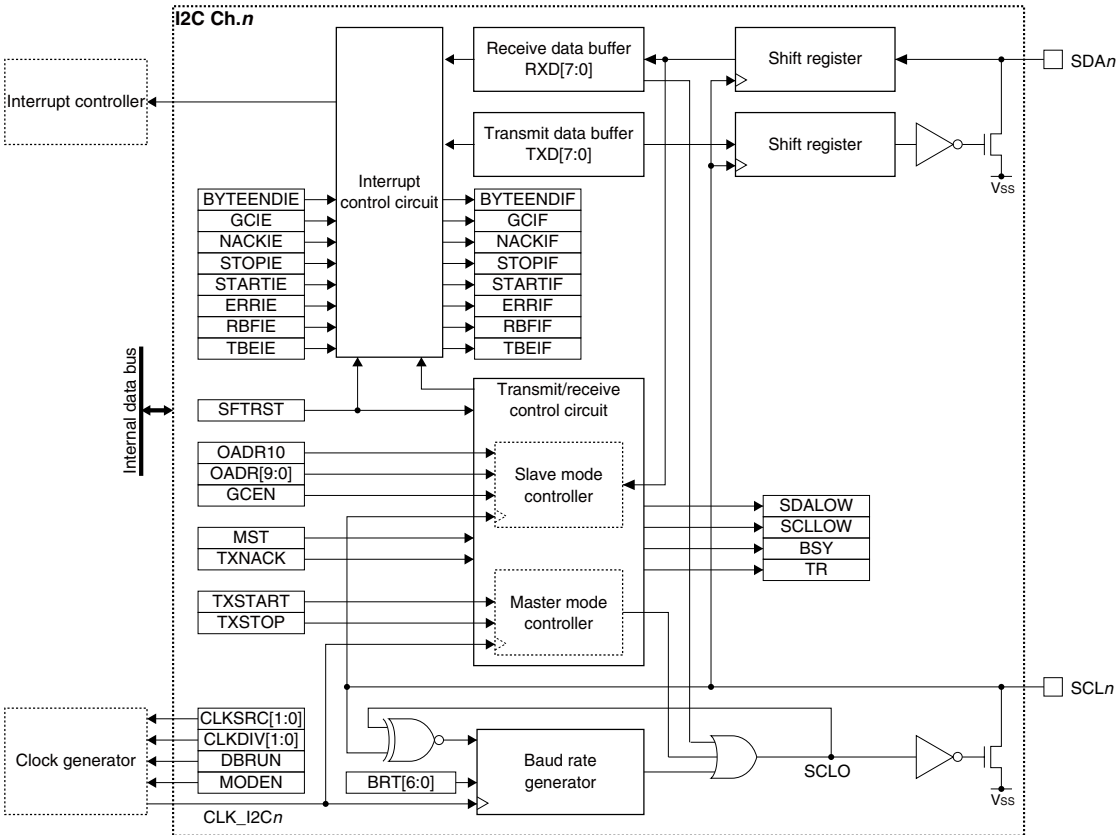


Figure 12.1.1 I2C Configuration

12.2 Input/Output Pins and External Connections

12.2.1 List of Input/Output Pins

Table 12.2.1.1 lists the I2C pins.

Table 12.2.1.1 List of I2C Pins

Pin name	I/O*	Initial status*	Function
SDA*n*	I/O	I	I2C bus serial data input/output pin
SCL*n*	I/O	I	I2C bus clock input/output pin

* Indicates the status when the pin is configured for the I2C.

If the port is shared with the I2C pin and other functions, the I2C input/output function must be assigned to the port before activating the I2C. For more information, refer to the "I/O Ports" chapter.

12.2.2 External Connections

Figure 12.2.2.1 shows a connection diagram between the I2C in this IC and external I2C devices.
The serial data (SDA) and serial clock (SCL) lines must be pulled up with an external resistor.
When the I2C is set into master mode, one or more slave devices that have a unique address may be connected to the I2C bus. When the I2C is set into slave mode, one or more master and slave devices that have a unique address may be connected to the I2C bus.

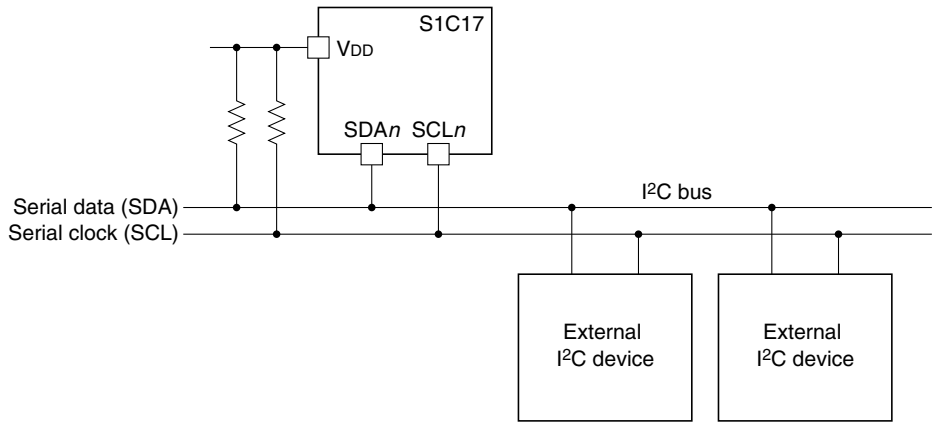


Figure 12.2.2.1 Connections between I2C and External I2C Devices

Notes: • The SDA and SCL lines must be pulled up to a VDD of this IC or lower voltage. However, if the I2C input/output ports are configured with the over voltage tolerant fail-safe type I/O, these lines can be pulled up to a voltage exceeding the VDD of this IC but within the recommended operating voltage range of this IC.

• The internal pull-up resistors for the I/O ports cannot be used for pulling up SDA and SCL.

• When the I2C is set into master mode, no other master device can be connected to the I2C bus.

12.3 Clock Settings

12.3.1 I2C Operating Clock

Master mode operating clock

When using the I2C Ch.*n* in master mode, the I2C Ch.*n* operating clock CLK_I2C*n* must be supplied to the I2C Ch.*n* from the clock generator. The CLK_I2C*n* supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Set the following I2C*n*CLK register bits:
   - I2C*n*CLK.CLKSRC[1:0] bits (Clock source selection)
   - I2C*n*CLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)

   When using the I2C in master mode during SLEEP mode, the I2C Ch.*n* operating clock CLK_I2C*n* must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_I2C*n* clock source.
   The I2C operating clock should be selected so that the baud rate generator will be configured easily.

Slave mode operating clock

The I2C set to slave mode uses the SCL supplied from the I²C master as its operating clock. The clock setting by the I2C*n*CLK register is ineffective.
The I2C keeps operating using the clock supplied from the external I²C master even if all the internal clocks halt during SLEEP mode, so the I2C can receive data and can generate receive buffer full interrupts.

12.3.2 Clock Supply in DEBUG Mode

In master mode, the CLK_I2C*n* supply during DEBUG mode should be controlled using the I2C*n*CLK.DBRUN bit. The CLK_I2C*n* supply to the I2C Ch.*n* is suspended when the CPU enters DEBUG mode if the I2C*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_I2C*n* supply resumes. Although the I2C Ch.*n* stops operating when the CLK_I2C*n* supply is suspended, the output pin and registers retain the status before DEBUG mode was entered. If the I2C*n*CLK.DBRUN bit = 1, the CLK_I2C*n* supply is not suspended and the I2C Ch.*n* will keep operating in DEBUG mode.
In slave mode, the I2C Ch.*n* operates with the external I²C master clock input from the SCL*n* pin regardless of whether the CPU is placed into DEBUG mode or normal mode.

12.3.3 Baud Rate Generator

The I2C includes a baud rate generator to generate the serial clock SCL used in master mode. The I2C set to slave mode does not use the baud rate generator, as it operates with the serial clock input from the SCL*n* pin.

Setting data transfer rate (for master mode)

The transfer rate is determined by the I2C*n*BR.BRT[6:0] bit settings. Use the following equations to calculate the setting values for obtaining the desired transfer rate.

$$bps = \frac{f_{CLK\_I2Cn}}{(BRT + 3) \times 2} \qquad BRT = \frac{f_{CLK\_I2Cn}}{bps \times 2} - 3 \qquad \text{(Eq. 12.1)}$$

Where
- bps: Data transfer rate [bit/s]
- f_{CLK_I2Cn}: I2C operating clock frequency [Hz]
- BRT: I2C*n*BR.BRT[6:0] bits setting value (1 to 127)

∗ The equations above do not include SCL rising/falling time and delay time by clock stretching (see Figure 12.3.3.1).

Note: The I²C bus transfer rate is limited to 100 kbit/s in standard mode or 400 kbit/s in fast mode. Do not set a transfer rate exceeding the limit.

Baud rate generator clock output and operations for supporting clock stretching

Figure 12.3.3.1 shows the clock generated by the baud rate generator and the clock waveform on the I²C bus.

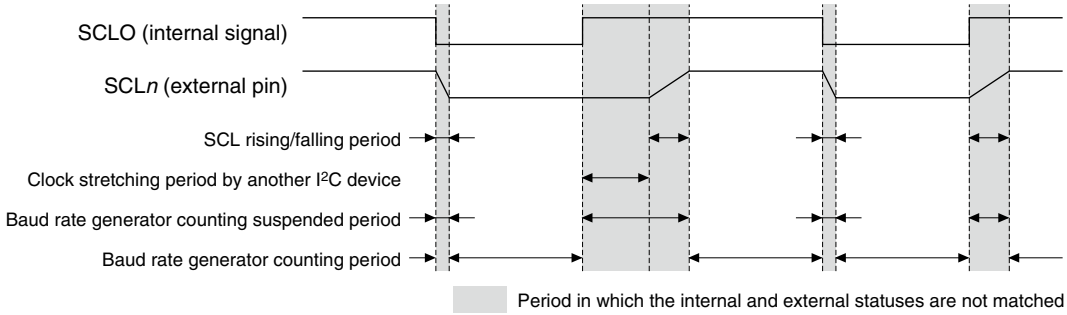


Figure 12.3.3.1 Baud Rate Generator Output Clock and SCL*n* Output Waveform

The baud rate generator output clock SCLO is compared with the SCL*n* pin status and the results are returned to the baud rate generator. If a mismatch has occurred between SCLO and SCL*n* pin levels, the baud rate generator suspends counting. This extends the clock to control data transfer during the SCL signal rising/falling period and clock stretching period in which SCL is fixed at low by a slave device.

12.4 Operations

12.4.1 Initialization

The I2C Ch.*n* should be initialized with the procedure shown below.

When using the I2C in master mode

1. Configure the operating clock and the baud rate generator using the I2C*n*CLK and I2C*n*BR registers.
2. Assign the I2C Ch.*n* input/output function to the ports. (Refer to the "I/O Ports" chapter.)
3. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the I2C*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the I2C*n*INTE register to 1. (Enable interrupts)
4. Set the following I2C*n*CTL register bits:
   - Set the I2C*n*CTL.MST bit to 1. (Set master mode)
   - Set the I2C*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the I2C*n*CTL.MODEN bit to 1. (Enable I2C Ch.*n* operations)

When using the I2C in slave mode

1. Set the following I2C*n*MOD register bits:
   - I2C*n*MOD.OADR10 bit (Set 10/7-bit address mode)
   - I2C*n*MOD.GCEN bit (Enable response to general call address)
2. Set its own address to the I2C*n*OADR.OADR[9:0] (or OADR[6:0]) bits.
3. Assign the I2C Ch.*n* input/output function to the ports. (Refer to the "I/O Ports" chapter.)
4. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the I2C*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the I2C*n*INTE register to 1. (Enable interrupts)
5. Set the following I2C*n*CTL register bits:
   - Set the I2C*n*CTL.MST bit to 0. (Set slave mode)
   - Set the I2C*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the I2C*n*CTL.MODEN bit to 1. (Enable I2C Ch.*n* operations)

12.4.2 Data Transmission in Master Mode

A data sending procedure in master mode and the I2C Ch.*n* operations are shown below. Figures 12.4.2.1 and 12.4.2.2 show an operation example and a flowchart, respectively.

Data sending procedure

1. Issue a START condition by setting the I2C*n*CTL.TXSTART bit to 1.
2. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1) or a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   Clear the I2C*n*INTF.STARTIF bit by writing 1 after the interrupt has occurred.
3. Write the 7-bit slave address to the I2C*n*TXD.TXD[7:1] bits and 0 that represents WRITE as the data transfer direction to the I2C*n*TXD.TXD0 bit.
4. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1) generated when an ACK is received or a NACK reception interrupt (I2C*n*INTF.NACKIF bit = 1) generated when a NACK is received.
   i. Go to Step 5 if transmit data remains when a transmit buffer empty interrupt has occurred.
   ii. Go to Step 7 or 1 after clearing the I2C*n*INTF.NACKIF bit when a NACK reception interrupt has occurred.
5. Write transmit data to the I2C*n*TXD register.
6. Repeat Steps 4 and 5 until the end of transmit data.
7. Issue a STOP condition by setting the I2C*n*CTL.TXSTOP bit to 1.
8. Wait for a STOP condition interrupt (I2C*n*INTF.STOPIF bit = 1).
   Clear the I2C*n*INTF.STOPIF bit by writing 1 after the interrupt has occurred.

Data sending operations

Generating a START condition

The I2C Ch.*n* starts generating a START condition when the I2C*n*CTL.TXSTART bit is set to 1. When the generating operation has completed, the I2C Ch.*n* clears the I2C*n*CTL.TXSTART bit to 0 and sets both the I2C*n*INTF.STARTIF and I2C*n*INTF.TBEIF bits to 1.

Sending slave address and data

If the I2C*n*INTF.TBEIF bit = 1, a slave address or data can be written to the I2C*n*TXD register. The I2C Ch.*n* pulls down SCL to low and enters standby state until data is written to the I2C*n*TXD register. The writing operation triggers the I2C Ch.*n* to send the data to the shift register automatically and to output eight clock pulses and data bits to the I²C bus.

When the slave device returns an ACK as the response, the I2C*n*INTF.TBEIF bit is set to 1. After this interrupt occurs, the subsequent data may be sent or a STOP/repeated START condition may be issued to terminate transmission. If the slave device returns NACK, the I2C*n*INTF.NACKIF bit is set to 1 without setting the I2C*n*INTF.TBEIF bit.

Generating a STOP/repeated START condition

After the I2C*n*INTF.TBEIF bit is set to 1 (transmit buffer empty) or the I2C*n*INTF.NACKIF bit is set to 1 (NACK received), setting the I2C*n*CTL.TXSTOP bit to 1 generates a STOP condition. When the bus free time (t_{BUF} defined in the I²C Specifications) has elapsed after the STOP condition has been generated, the I2C*n*CTL.TXSTOP bit is cleared to 0 and the I2C*n*INTF.STOPIF bit is set to 1.

When setting the I2C*n*CTL.TXSTART bit to 1 while the I2C*n*INTF.TBEIF bit = 1 (transmit buffer empty) or the I2C*n*INTF.NACKIF bit = 1 (NACK received), the I2C Ch.*n* generates a repeated START condition. When the repeated START condition has been generated, the I2C*n*INTF.STARTIF and I2C*n*INTF.TBEIF bits are both set to 1 same as when a START condition has been generated.

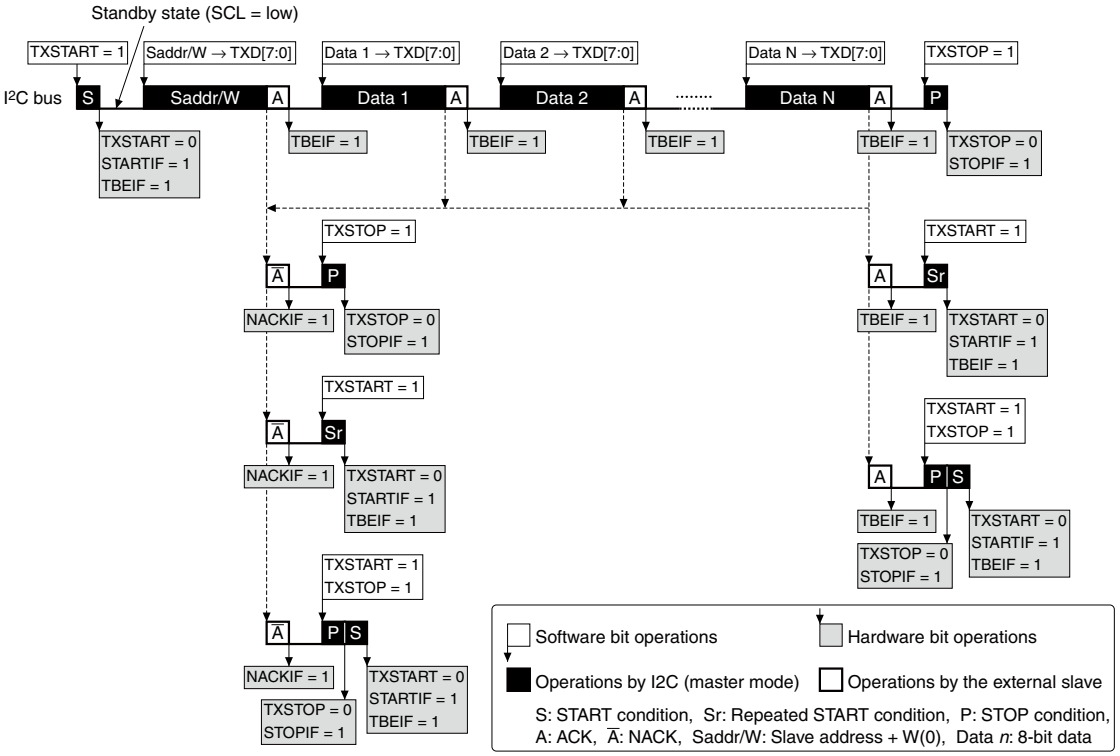


Figure 12.4.2.1 Example of Data Sending Operations in Master Mode

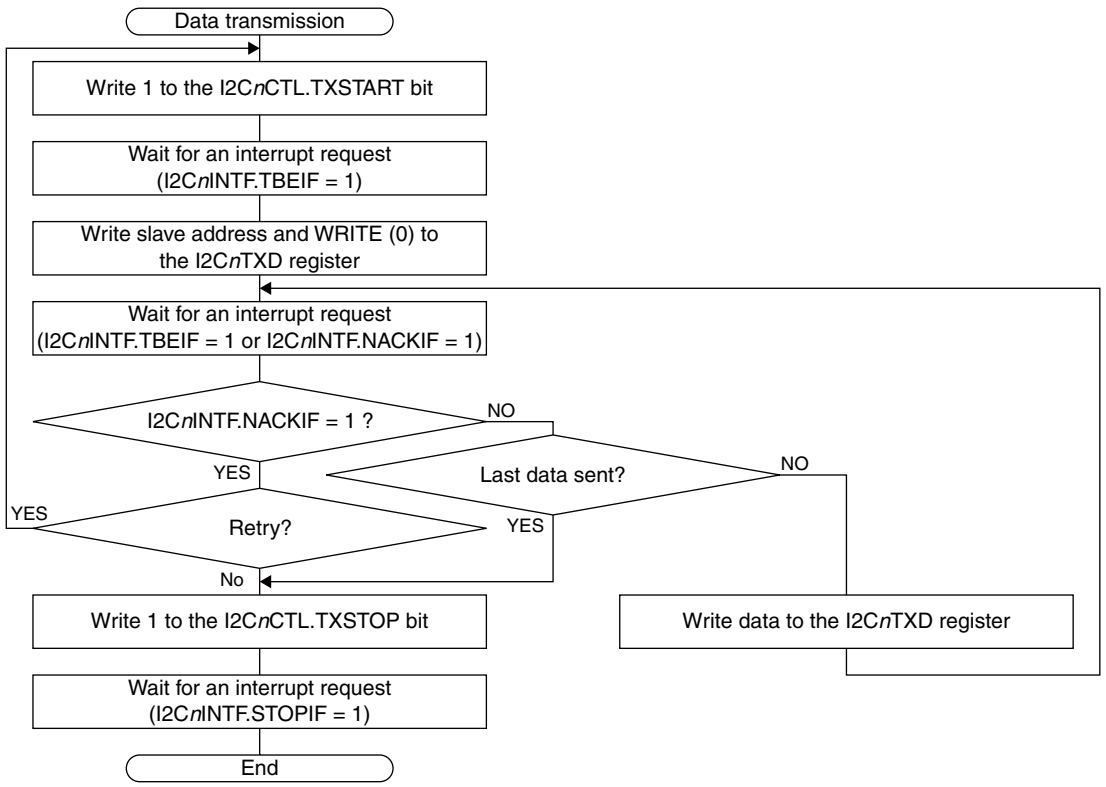


Figure 12.4.2.2 Master Mode Data Transmission Flowchart

12.4.3 Data Reception in Master Mode

A data receiving procedure in master mode and the I2C Ch.*n* operations are shown below. Figures 12.4.3.1 and 12.4.3.2 show an operation example and a flowchart, respectively.

Data receiving procedure

1. Issue a START condition by setting the I2C*n*CTL.TXSTART bit to 1.
2. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1) or a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   Clear the I2C*n*INTF.STARTIF bit by writing 1 after the interrupt has occurred.
3. Write the 7-bit slave address to the I2C*n*TXD.TXD[7:1] bits and 1 that represents READ as the data transfer direction to the I2C*n*TXD.TXD0 bit.
4. Wait for a receive buffer full interrupt (I2C*n*INTF.RBFIF bit = 1) generated when a one-byte reception has completed or a NACK reception interrupt (I2C*n*INTF.NACKIF bit = 1) generated when a NACK is received.
   i. Go to Step 5 when a receive buffer full interrupt has occurred.
   ii. Clear the I2C*n*INTF.NACKIF bit and issue a STOP condition by setting the I2C*n*CTL.TXSTOP bit to 1 when a NACK reception interrupt has occurred. Then go to Step 8 or Step 1 if making a retry.
5. Perform one of the operations below when the last or next-to-last data is received.
   i. When the next-to-last data is received, write 1 to the I2C*n*CTL.TXNACK bit to send a NACK after the last data is received, and then go to Step 6.
   ii. When the last data is received, read the received data from the I2C*n*RXD register and set the I2C*n*CTL.TXSTOP to 1 to generate a STOP condition. Then go to Step 8.
6. Read the received data from the I2C*n*RXD register.
7. Repeat Steps 4 to 6 until the end of data reception.
8. Wait for a STOP condition interrupt (I2C*n*INTF.STOPIF bit = 1).
   Clear the I2C*n*INTF.STOPIF bit by writing 1 after the interrupt has occurred.

Data receiving operations

Generating a START condition

It is the same as the data transmission in master mode.

Sending slave address

It is the same as the data transmission in master mode. Note, however, that the I2C*n*TXD.TXD0 bit must be set to 1 that represents READ as the data transfer direction to issue a request to the slave to send data.

Receiving data

After the slave address has been sent, the slave device sends an ACK and the first data. The I2C Ch.*n* sets the I2C*n*INTF.RBFIF bit to 1 after the data reception has completed. Furthermore, the I2C Ch.*n* returns an ACK. To return a NACK, such as for a response after the last data has been received, write 1 to the I2C*n*CTL.TXNACK bit before the I2C*n*INTF.RBFIF bit is set to 1.

The received data can be read out from the I2C*n*RXD register after a receive buffer full interrupt has occurred. The I2C Ch.*n* pulls down SCL to low and enters standby state until data is read out from the I2C*n*RXD register.

This reading triggers the I2C Ch.*n* to start subsequent data reception.

Generating a STOP or repeated START condition

It is the same as the data transmission in master mode.

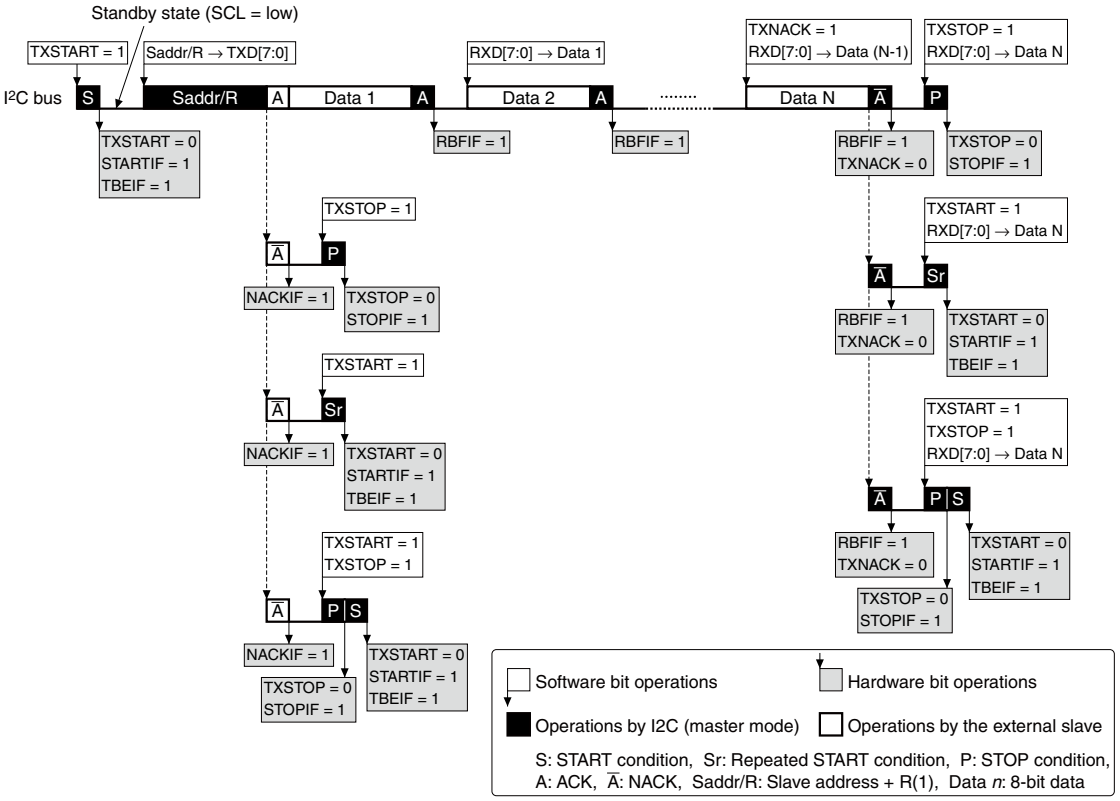


Figure 12.4.3.1 Example of Data Receiving Operations in Master Mode

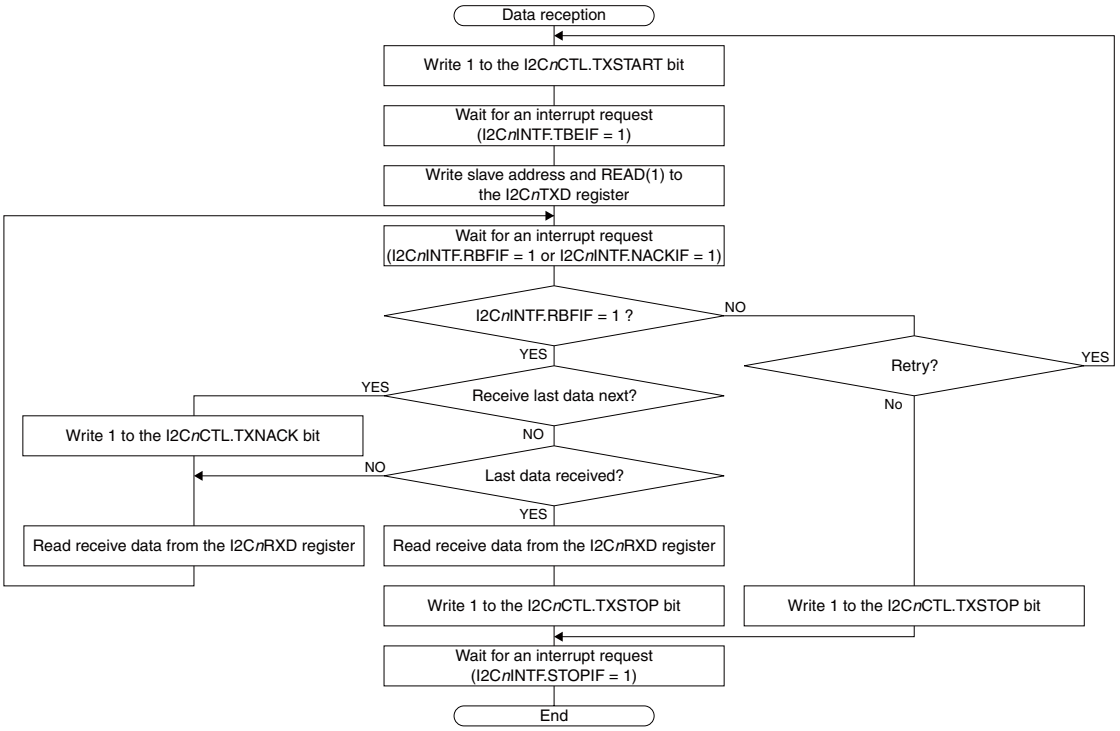


Figure 12.4.3.2 Master Mode Data Reception Flowchart

12.4.4 10-bit Addressing in Master Mode

A 10-bit address consists of the first address that contains two high-order bits and the second address that contains eight low-order bits.

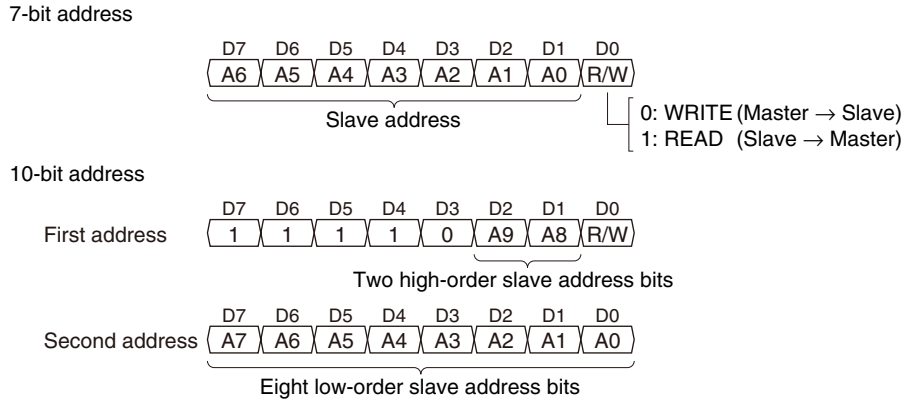


Figure 12.4.4.1 10-bit Address Configuration

The following shows a procedure to start data transfer in 10-bit address mode when the I2C Ch.*n* is placed into master mode (see the 7-bit mode descriptions above for control procedures when a NACK is received or sending/receiving data). Figure 12.4.4.2 shows an operation example.

Starting data transmission in 10-bit address mode

1. Issue a START condition by setting the I2C*n*CTL.TXSTART bit to 1.
2. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1) or a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   Clear the I2C*n*INTF.STARTIF bit by writing 1 after the interrupt has occurred.
3. Write the first address to the I2C*n*TXD.TXD[7:1] bits and 0 that represents WRITE as the data transfer direction to the I2C*n*TXD.TXD0 bit.
4. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1).
5. Write the second address to the I2C*n*TXD.TXD[7:0] bits.
6. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1).
7. Perform data transmission.

Starting data reception in 10-bit address mode

1 to 6. These steps are the same as the data transmission starting procedure described above.

7. Issue a repeated START condition by setting the I2C*n*CTL.TXSTART bit to 1.
8. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1) or a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   Clear the I2C*n*INTF.STARTIF bit by writing 1 after the interrupt has occurred.
9. Write the first address to the I2C*n*TXD.TXD[7:1] bits and 1 that represents READ as the data transfer direction to the I2C*n*TXD.TXD0 bit.
10. Perform data reception.

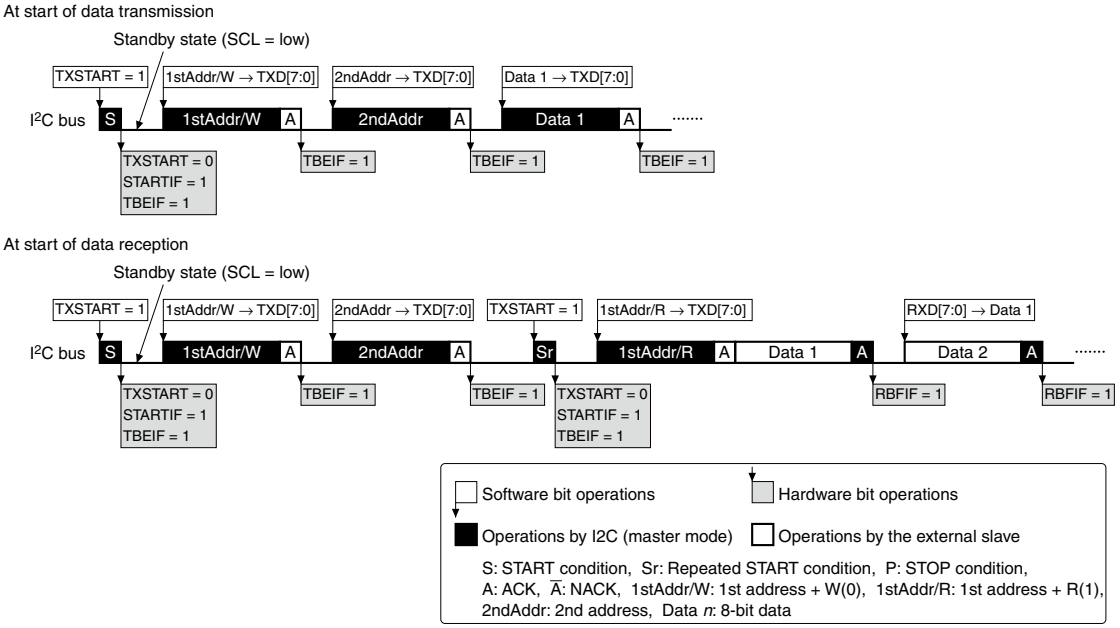


Figure 12.4.4.2 Example of Data Transfer Starting Operations in 10-bit Address Mode (Master Mode)

12.4.5 Data Transmission in Slave Mode

A data sending procedure in slave mode and the I2C Ch.*n* operations are shown below. Figures 12.4.5.1 and 12.4.5.2 show an operation example and a flowchart, respectively.

Data sending procedure

1. Wait for a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   Clear the I2C*n*INTF.STARTIF bit by writing 1 after the interrupt has occurred.
2. Check to see if the I2C*n*INTF.TR bit = 1 (transmission mode).
   (Start a data receiving procedure if the I2C*n*INTF.TR bit = 0.)
3. Write transmit data to the I2C*n*TXD register.
4. Wait for a transmit buffer empty interrupt (I2C*n*INTF.TBEIF bit = 1), a NACK reception interrupt (I2C*n*INTF.NACKIF bit = 1), or a STOP condition interrupt (I2C*n*INTF.STOPIF bit = 1).
   i. Go to Step 3 when a transmit buffer empty interrupt has occurred.
   ii. Go to Step 5 after clearing the I2C*n*INTF.NACKIF bit when a NACK reception interrupt has occurred.
   iii. Go to Step 6 when a STOP condition interrupt has occurred.
5. Wait for a STOP condition interrupt (I2C*n*INTF.STOPIF bit = 1) or a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   i. Go to Step 6 when a STOP condition interrupt has occurred.
   ii. Go to Step 2 when a START condition interrupt has occurred.
6. Clear the I2C*n*INTF.STOPIF bit and then terminate data sending operations.

Data sending operations

START condition detection and slave address check

While the I2C*n*CTL.MODEN bit = 1 and the I2C*n*CTL.MST bit = 0 (slave mode), the I2C Ch.*n* monitors the I²C bus. When the I2C Ch.*n* detects a START condition, it starts receiving of the slave address sent from the master. If the received address is matched with the own address set to the I2C*n*OADR.OADR[6:0] bits (when the I2C*n*MOD.OADR10 bit = 0 (7-bit address mode)) or the I2C*n*OADR.OADR[9:0] bits (when the I2C*n*MOD.OADR10 bit = 1 (10-bit address mode)), the I2C*n*INTF.STARTIF bit and the I2C*n*INTF.BSY bit are both set to 1. The I2C Ch.*n* sets the I2C*n*INTF.TR bit to the R/W bit value in the received address. If this value is 1, the I2C Ch.*n* sets the I2C*n*INTF.TBEIF bit to 1 and starts data sending operations.

Sending the first data byte

After the valid slave address has been received, the I2C Ch.*n* pulls down SCL to low and enters standby state until data is written to the I2C*n*TXD register. This puts the I²C bus into clock stretching state and the external master into standby state. When transmit data is written to the I2C*n*TXD register, the I2C Ch.*n* clears the I2C*n*INTF.TBEIF bit and sends an ACK to the master. The transmit data written in the I2C*n*TXD register is automatically transferred to the shift register and the I2C*n*INTF.TBEIF bit is set to 1. The data bits in the shift register are output in sequence to the I²C bus.

Sending subsequent data

If the I2C*n*INTF.TBEIF bit = 1, subsequent transmit data can be written during data transmission. If the I2C*n*INTF.TBEIF bit is still set to 1 when the data transmission from the shift register has completed, the I2C Ch.*n* pulls down SCL to low (sets the I²C bus into clock stretching state) until transmit data is written to the I2C*n*TXD register.

If the next transmit data already exists in the I2C*n*TXD register or data has been written after the above, the I2C Ch.*n* sends the subsequent eight-bit data when an ACK from the external master is received. At the same time, the I2C*n*INTF.BYTEENDIF bit is set to 1. If a NACK is received, the I2C*n*INTF.NACKIF bit is set to 1 without sending data.

STOP/repeated START condition detection

While the I2C*n*CTL.MST bit = 0 (slave mode) and the I2C*n*INTF.BSY = 1, the I2C Ch.*n* monitors the I²C bus. When the I2C Ch.*n* detects a STOP condition, it terminates data sending operations. At this time, the I2C*n*INTF.BSY bit is cleared to 0 and the I2C*n*INTF.STOPIF bit is set to 1. Also when the I2C Ch.*n* detects a repeated START condition, it terminates data sending operations. In this case, the I2C*n*INTF.STARTIF bit is set to 1.

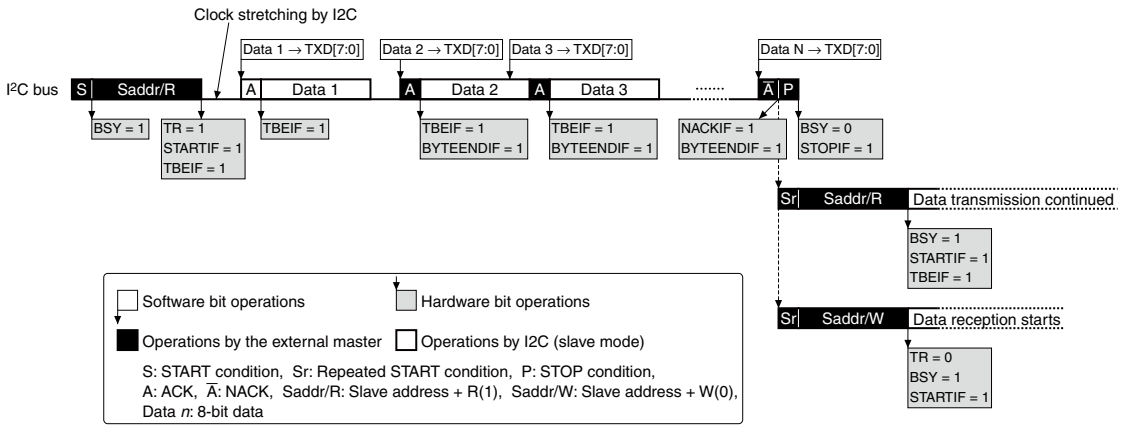


Figure 12.4.5.1 Example of Data Sending Operations in Slave Mode

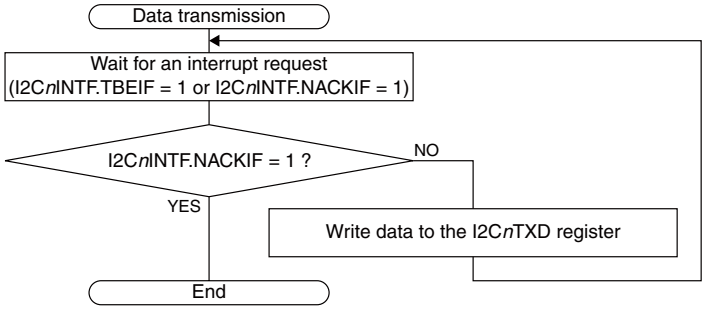


Figure 12.4.5.2 Slave Mode Data Transmission Flowchart

12.4.6 Data Reception in Slave Mode

A data receiving procedure in slave mode and the I2C Ch.*n* operations are shown below. Figures 12.4.6.1 and 12.4.6.2 show an operation example and a flowchart, respectively.

Data receiving procedure

1. Wait for a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
2. Check to see if the I2C*n*INTF.TR bit = 0 (reception mode).
   (Start a data sending procedure if I2C*n*INTF.TR bit = 1.)
3. Clear the I2C*n*INTF.STARTIF bit by writing 1.
4. Wait for a receive buffer full interrupt (I2C*n*INTF.RBFIF bit = 1) generated when a one-byte reception has completed or an end of transfer interrupt (I2C*n*INTF.BYTEENDIF bit = 1).
   Clear the I2C*n*INTF.BYTEENDIF bit by writing 1 after the interrupt has occurred.
5. If the next receive data is the last one, write 1 to the I2C*n*CTL.TXNACK bit to send a NACK after it is received.
6. Read the received data from the I2C*n*RXD register.
7. Repeat Steps 4 to 6 until the end of data reception.
8. Wait for a STOP condition interrupt (I2C*n*INTF.STOPIF bit = 1) or a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
   i. Go to Step 9 when a STOP condition interrupt has occurred.
   ii. Go to Step 2 when a START condition interrupt has occurred.
9. Clear the I2C*n*INTF.STOPIF bit and then terminate data receiving operations.

Data receiving operations

START condition detection and slave address check

It is the same as the data transmission in slave mode.
However, the I2C*n*INTF.TR bit is cleared to 0 and the I2C*n*INTF.TBEIF bit is not set.
If the I2C*n*MOD.GCEN bit is set to 1 (general call address response enabled), the I2C Ch.*n* starts data receiving operations when the general call address is received.
Slave mode can be operated even in SLEEP mode, it makes it possible to wake the CPU up using an interrupt when an address match is detected.

Receiving the first data byte

After the valid slave address has been received, the I2C Ch.*n* sends an ACK and pulls down SCL to low until 1 is written to the I2C*n*INTF.STARTIF bit. This puts the I²C bus into clock stretching state and the external master into standby state. When 1 is written to the I2C*n*INTF.STARTIF bit, the I2C Ch.*n* releases SCL and receives data sent from the external master into the shift register. After eight-bit data has been received, the I2C Ch.*n* sends an ACK and pulls down SCL to low. The received data in the shift register is transferred to the receive data buffer and the I2C*n*INTF.RBFIF and I2C*n*INTF.BYTEENDIF bits are both set to 1. After that, the received data can be read out from the I2C*n*RXD register.

Receiving subsequent data

When the received data is read out from the I2C*n*RXD register after the I2C*n*INTF.RBFIF bit has been set to 1, the I2C Ch.*n* clears the I2C*n*INTF.RBFIF bit to 0, releases SCL, and receives subsequent data sent from the external master. After eight-bit data has been received, the I2C Ch.*n* sends an ACK and pulls down SCL to low. The received data in the shift register is transferred to the receive data buffer and the I2C*n*INTF.RBFIF and I2C*n*INTF.BYTEENDIF bits are both set to 1.

To return a NACK after eight-bit data is received, such as when terminating data reception, write 1 to the I2C*n*CTL.TXNACK bit before the data reception is completed. The I2C*n*CTL.TXNACK bit is automatically cleared to 0 after a NACK has been sent.

STOP/repeated START condition detection

It is the same as the data transmission in slave mode.

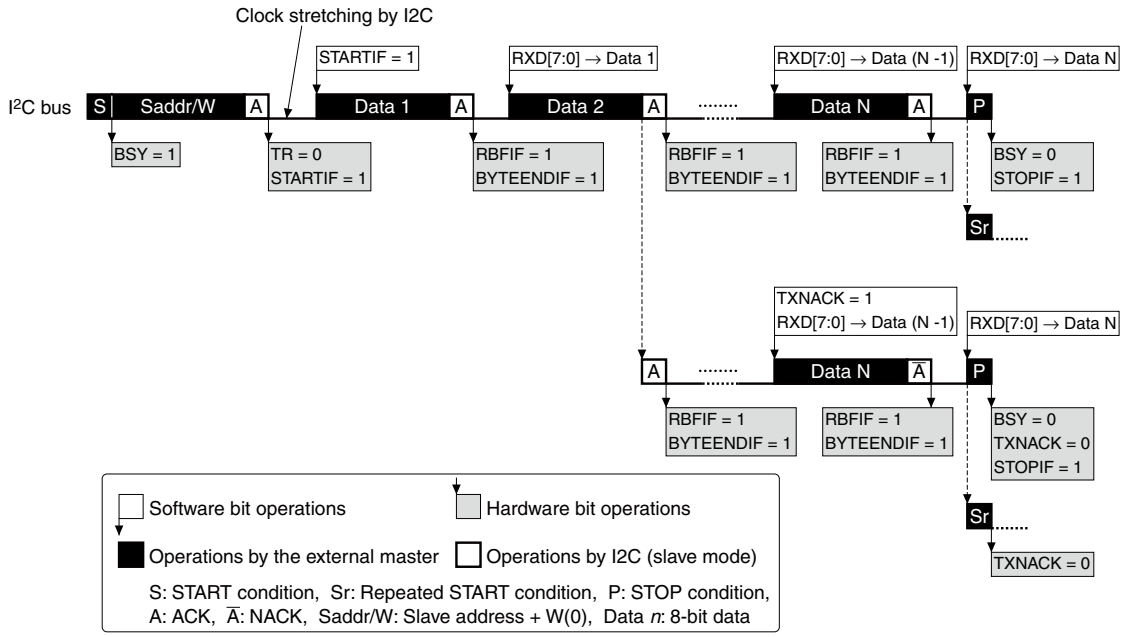


Figure 12.4.6.1 Example of Data Receiving Operations in Slave Mode

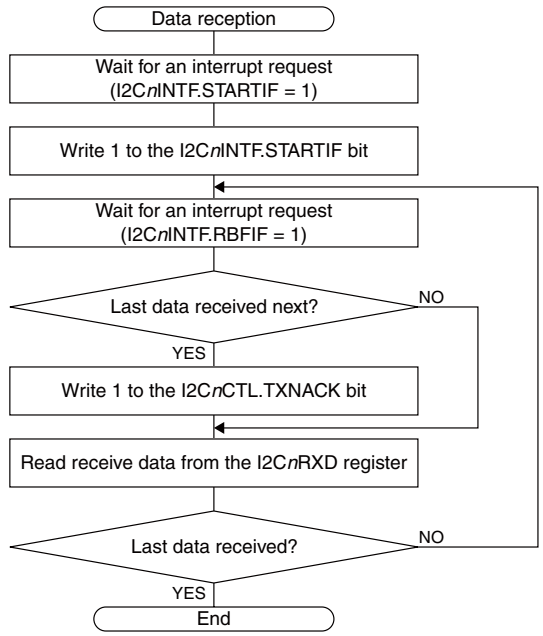


Figure 12.4.6.2 Slave Mode Data Reception Flowchart

12.4.7 Slave Operations in 10-bit Address Mode

The I2C Ch.*n* functions as a slave device in 10-bit address mode when the I2C*n*CTL.MST bit = 0 and the I2C*n*MOD.OADR10 bit = 1.
The following shows the address receiving operations in 10-bit address mode. Figure 12.4.7.1 shows an operation example. See Figure 12.4.4.1 for the 10-bit address configuration.

10-bit address receiving operations

After a START condition is issued, the master sends the first address that includes the two high-order slave address bits and the R/W bit (= 0). If the received two high-order slave address bits are matched with the I2C*n*OADR.OADR[9:8] bits, the I2C Ch.*n* returns an ACK. At this time, other slaves may returns an ACK as the two high-order bits may be matched.
Then the master sends the eight low-order slave address bits as the second address. If this address is matched with the I2C*n*OADR.OADR[7:0] bits, the I2C Ch.*n* returns an ACK and starts data receiving operations.
If the master issues a request to the slave to send data (data reception in the master), the master generates a repeated START condition and sends the first address with the R/W bit set to 1. This reception switches the I2C Ch.*n* to data sending mode.

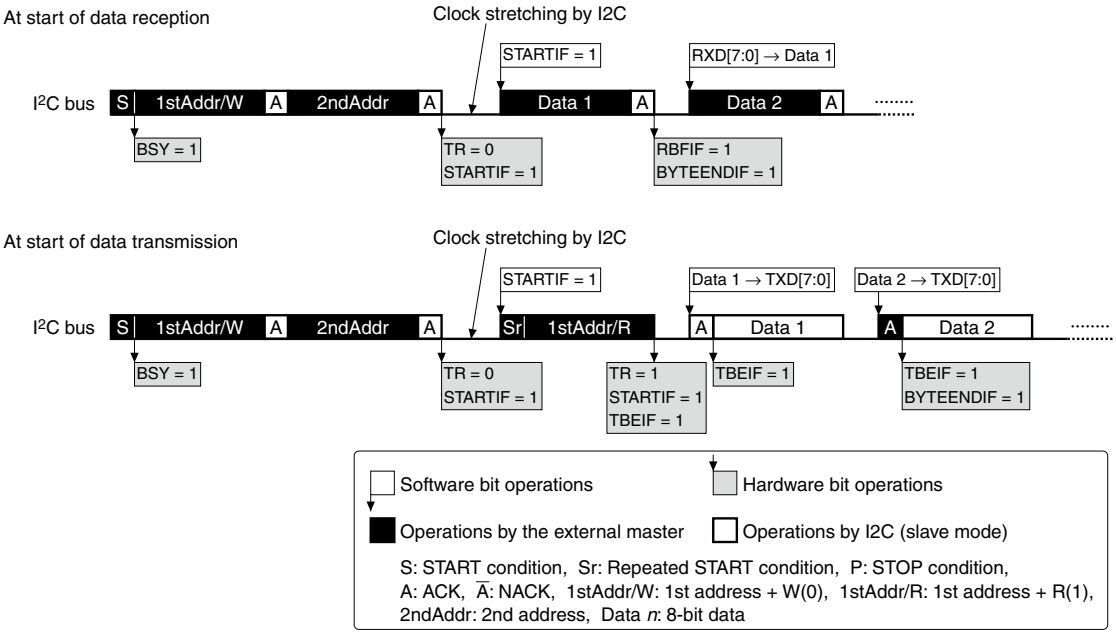


Figure 12.4.7.1 Example of Data Transfer Starting Operations in 10-bit Address Mode (Slave Mode)

12.4.8 Automatic Bus Clearing Operation

The I2C Ch.*n* set into master mode checks the SDA state immediately before generating a START condition. If SDA is set to a low level at this time, the I2C Ch.*n* automatically executes bus clearing operations that output up to ten clocks from the SCL*n* pin with SDA left free state.
When SDA goes high from low within nine clocks, the I2C Ch.*n* issues a START condition and starts normal operations. If SDA does not change from low when the I2C Ch.*n* outputs the ninth clock, it is regarded as an automatic bus clearing failure. In this case, the I2C Ch.*n* clears the I2C*n*CTL.TXSTART bit to 0 and sets both the I2C*n*INTF.ERRIF and I2C*n*INTF.STARTIF bits to 1.

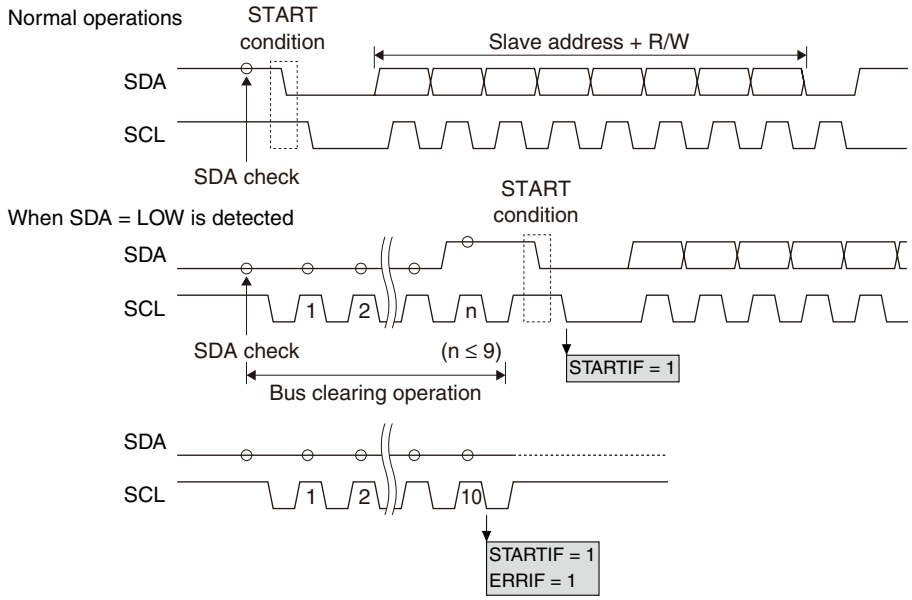


Figure 12.4.8.1 Automatic Bus Clearing Operation

12.4.9 Error Detection

The I2C includes a hardware error detection function.
Furthermore, the I2C*n*INTF.SDALOW and I2C*n*INTF.SCLLOW bits are provided to allow software to check whether the SDA and SCL lines are fixed at low. If unintended low level is detected on SDA or SCL, a software recovery processing, such as I2C Ch.*n* software reset, can be performed.

The table below lists the hardware error detection conditions and the notification method.

Table 12.4.9.1 Hardware Error Detection Function

No.	Error detecting period/timing	I²C bus line monitored and error condition	Notification method
1	While the I2C Ch.*n* controls SDA to high for sending address, data, or a NACK	SDA = low	I2C*n*INTF.ERRIF = 1
2	<Master mode only> When 1 is written to the I2C*n*CTL.TXSTART bit while the I2C*n*INTF.BSY bit = 0	SCL = low	I2C*n*INTF.ERRIF = 1 I2C*n*CTL.TXSTART = 0 I2C*n*INTF.STARTIF = 1
3	<Master mode only> When 1 is written to the I2C*n*CTL.TXSTOP bit while the I2C*n*INTF.BSY bit = 0	SCL = low	I2C*n*INTF.ERRIF = 1 I2C*n*CTL.TXSTOP = 0 I2C*n*INTF.STOPIF = 1
4	<Master mode only> When 1 is written to the I2C*n*CTL.TXSTART bit while the I2C*n*INTF.BSY bit = 0 (Refer to "Automatic Bus Clearing Operation.")	SDA Automatic bus clearing failure	I2C*n*INTF.ERRIF = 1 I2C*n*CTL.TXSTART = 0 I2C*n*INTF.STARTIF = 1

12.5 Interrupts

The I2C has a function to generate the interrupts shown in Table 12.5.1.

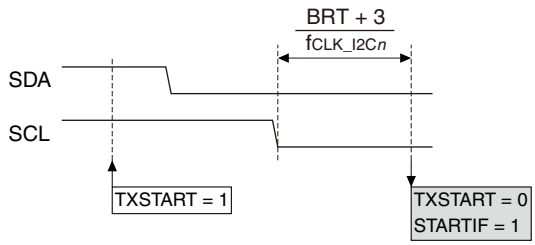
Table 12.5.1 I2C Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
End of data transfer	I2C*n*INTF.BYTEENDIF	When eight-bit data transfer and the following ACK/NACK transfer are completed	Writing 1, software reset
General call address reception	I2C*n*INTF.GCIF	Slave mode only: When the general call address is received	Writing 1, software reset
NACK reception	I2C*n*INTF.NACKIF	When a NACK is received	Writing 1, software reset
STOP condition	I2C*n*INTF. STOPIF	Master mode: When a STOP condition is generated and the bus free time (t_{BUF}) between STOP and START conditions has elapsed Slave mode: When a STOP condition is detected while the I2C Ch.*n* is selected as the slave currently accessed	Writing 1, software reset
START condition	I2C*n*INTF. STARTIF	Master mode: When a START condition is issued Slave mode: When an address match is detected (including general call)	Writing 1, software reset
Error detection	I2C*n*INTF. ERRIF	Refer to "Error Detection."	Writing 1, software reset
Receive buffer full	I2C*n*INTF. RBFIF	When received data is loaded to the receive data buffer	Reading received data (to empty the receive data buffer), software reset
Transmit buffer empty	I2C*n*INTF. TBEIF	Master mode: When a START condition is issued or when an ACK is received from the slave Slave mode: When transmit data written to the transmit data buffer is transferred to the shift register or when an address match is detected with R/W bit set to 1	Writing transmit data

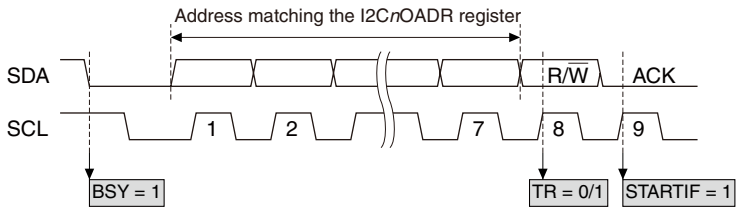
The I2C provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

(1) START condition interrupt

Master mode

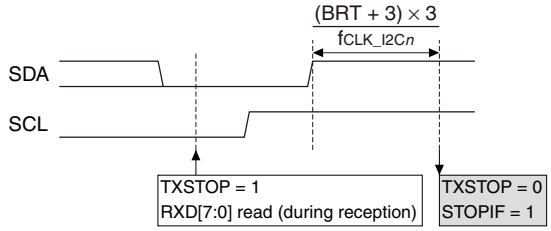


Slave mode

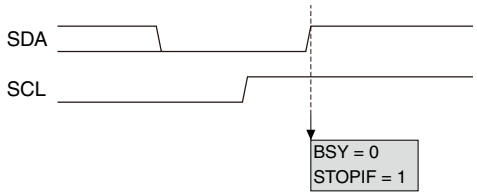


(2) STOP condition interrupt

Master mode



Slave mode



(f_{CLK_I2Cn}: I2C operating clock frequency [Hz], BRT: I2C*n*BR.BRT[6:0] bits setting value (1 to 127))

Figure 12.5.1 START/STOP Condition Interrupt Timings

12.6 Control Registers

I2C Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*CLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the I2C operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits select the division ratio of the I2C operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of the I2C.

Table 12.6.1 Clock Source and Division Ratio Settings

I2C*n*CLK.CLKDIV[1:0] bits	I2C*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x3	1/8	1/1	1/8	1/1
0x2	1/4		1/4	
0x1	1/2		1/2	
0x0	1/1		1/1	

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The I2C*n*CLK register settings can be altered only when the I2C*n*CTL.MODEN bit = 0.

I2C Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*MOD	15–8	–	0x00	–	R	–
	7–3	–	0x00	–	R	
	2	OADR10	0	H0	R/W	
	1	GCEN	0	H0	R/W	
	0	–	0	–	R	

Bits 15–3 Reserved

Bit 2 OADR10

This bit sets the number of own address bits for slave mode.
1 (R/W): 10-bit address
0 (R/W): 7-bit address

Bit 1 GCEN

This bit sets whether to respond to master general calls in slave mode or not.
1 (R/W): Respond to general calls.
0 (R/W): Do not respond to general calls.

Bit 0 Reserved

Note: The I2C*n*MOD register settings can be altered only when the I2C*n*CTL.MODEN bit = 0.

I2C Ch.*n* Baud-Rate Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*BR	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6–0	BRT[6:0]	0x7f	H0	R/W	

Bits 15–7 Reserved

Bits 6–0 BRT[6:0]

These bits set the I2C Ch.*n* transfer rate for master mode. For more information, refer to "Baud Rate Generator."

Notes: • The I2C*n*BR register settings can be altered only when the I2C*n*CTL.MODEN bit = 0.

• Be sure to avoid setting the I2C*n*BR register to 0.

I2C Ch.*n* Own Address Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*OADR	15–10	–	0x00	–	R	–
	9–0	OADR[9:0]	0x000	H0	R/W	

Bits 15–10 Reserved

Bits 9–0 OADR[9:0]

These bits set the own address for slave mode.
The I2C*n*OADR.OADR[9:0] bits are effective in 10-bit address mode (I2C*n*MOD.OADR10 bit = 1), or the I2C*n*OADR.OADR[6:0] bits are effective in 7-bit address mode (I2C*n*MOD.OADR10 bit = 0).

Note: The I2C*n*OADR register settings can be altered only when the I2C*n*CTL.MODEN bit = 0.

I2C Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*CTL	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5	MST	0	H0	R/W	
	4	TXNACK	0	H0/S0	R/W	
	3	TXSTOP	0	H0/S0	R/W	
	2	TXSTART	0	H0/S0	R/W	
	1	SFTRST	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–6 Reserved

Bit 5 MST

This bit selects the I2C Ch.*n* operating mode.

1 (R/W): Master mode

0 (R/W): Slave mode

Bit 4 TXNACK

This bit issues a request for sending a NACK at the next responding.

1 (W): Issue a NACK.

0 (W): Ineffective

1 (R): On standby or during sending a NACK

0 (R): NACK has been sent.

This bit is automatically cleared after a NACK has been sent.

Bit 3 TXSTOP

This bit issues a STOP condition in master mode. This bit is ineffective in slave mode.

1 (W): Issue a STOP condition.

0 (W): Ineffective

1 (R): On standby or during generating a STOP condition

0 (R): STOP condition has been generated.

This bit is automatically cleared when the bus free time (tBUF defined in the I²C Specifications) has elapsed after the STOP condition has been generated.

Bit 2 TXSTART

This bit issues a START condition in master mode. This bit is ineffective in slave mode.

1 (W): Issue a START condition.

0 (W): Ineffective

1 (R): On standby or during generating a START condition

0 (R): START condition has been generated.

This bit is automatically cleared when a START condition has been generated.

Bit 1 SFTRST

This bit issues software reset to the I2C.

1 (W): Issue software reset

0 (W): Ineffective

1 (R): Software reset is executing.

0 (R): Software reset has finished. (During normal operation)

Setting this bit resets the I2C transmit/receive control circuit and interrupt flags. This bit is automatically cleared after the reset processing has finished.

Bit 0 MODEN

This bit enables the I2C operations.

1 (R/W): Enable I2C operations (The operating clock is supplied.)

0 (R/W): Disable I2C operations (The operating clock is stopped.)

Note: If the I2C*n*CTL.MODEN bit is altered from 1 to 0 during sending/receiving data, the data being sent/received cannot be guaranteed. When setting the I2C*n*CTL.MODEN bit to 1 again after that, be sure to write 1 to the I2C*n*CTL.SFTRST bit as well.

I2C Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*TXD	15–8	–	0x00	–	R	–
	7–0	TXD[7:0]	0x00	H0	R/W	

Bits 15–8 Reserved

Bits 7–0 TXD[7:0]

Data can be written to the transmit data buffer through these bits. Make sure the I2C*n*INTF.TBEIF bit is set to 1 before writing data.

Note: Be sure to avoid writing to the I2C*n*TXD register when the I2C*n*INTF.TBEIF bit = 0, otherwise transmit data cannot be guaranteed.

I2C Ch.*n* Receive Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*RXD	15–8	–	0x00	–	R	–
	7–0	RXD[7:0]	0x00	H0	R	

Bits 15–8 Reserved

Bits 7–0 RXD[7:0]

The receive data buffer can be read through these bits.

I2C Ch.*n* Status and Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*INTF	15–13	–	0x0	–	R	–
	12	SDALOW	0	H0	R	
	11	SCLLOW	0	H0	R	
	10	BSY	0	H0/S0	R	
	9	TR	0	H0	R	
	8	–	0	–	R	
	7	BYTEENDIF	0	H0/S0	R/W	Cleared by writing 1.
	6	GCIF	0	H0/S0	R/W	
	5	NACKIF	0	H0/S0	R/W	
	4	STOPIF	0	H0/S0	R/W	
	3	STARTIF	0	H0/S0	R/W	
	2	ERRIF	0	H0/S0	R/W	
	1	RBFIF	0	H0/S0	R	Cleared by reading the I2C*n*RXD register.
	0	TBEIF	0	H0/S0	R	Cleared by writing to the I2C*n*TXD register.

Bits 15–13 Reserved

Bit 12 SDALOW

This bit indicates that SDA is set to low level.

1 (R): SDA = Low level

0 (R): SDA = High level

Bit 11 SCLLOW

This bit indicates that SCL is set to low level.

1 (R): SCL = Low level

0 (R): SCL = High level

Bit 10 BSY

This bit indicates that the I²C bus is placed into busy status.

1 (R): I²C bus busy

0 (R): I²C bus free

Bit 9 TR

This bit indicates whether the I2C is set in transmission mode or not.

1 (R): Transmission mode

0 (R): Reception mode

Bit 8 Reserved

Bit 7 BYTEENDIF
Bit 6 GCIF
Bit 5 NACKIF
Bit 4 STOPIF
Bit 3 STARTIF
Bit 2 ERRIF
Bit 1 RBFIF
Bit 0 TBEIF

These bits indicate the I2C interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag

0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:

I2C*n*INTF.BYTEENDIF bit: End of transfer interrupt
I2C*n*INTF.GCIF bit: General call address reception interrupt
I2C*n*INTF.NACKIF bit: NACK reception interrupt
I2C*n*INTF.STOPIF bit: STOP condition interrupt
I2C*n*INTF.STARTIF bit: START condition interrupt
I2C*n*INTF.ERRIF bit: Error detection interrupt
I2C*n*INTF.RBFIF bit: Receive buffer full interrupt
I2C*n*INTF.TBEIF bit: Transmit buffer empty interrupt

I2C Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C*n*INTE	15–8	–	0x00	–	R	–
	7	BYTEENDIE	0	H0	R/W	
	6	GCIE	0	H0	R/W	
	5	NACKIE	0	H0	R/W	
	4	STOPIE	0	H0	R/W	
	3	STARTIE	0	H0	R/W	
	2	ERRIE	0	H0	R/W	
	1	RBFIE	0	H0	R/W	
	0	TBEIE	0	H0	R/W	

Bits 15–8 Reserved

Bit 7 BYTEENDIE
Bit 6 GCIE
Bit 5 NACKIE
Bit 4 STOPIE
Bit 3 STARTIE
Bit 2 ERRIE
Bit 1 RBFIE
Bit 0 TBEIE

These bits enable I2C interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
I2C*n*INTE.BYTEENDIE bit: End of transfer interrupt
I2C*n*INTE.GCIE bit: General call address reception interrupt
I2C*n*INTE.NACKIE bit: NACK reception interrupt
I2C*n*INTE.STOPIE bit: STOP condition interrupt
I2C*n*INTE.STARTIE bit: START condition interrupt
I2C*n*INTE.ERRIE bit: Error detection interrupt
I2C*n*INTE.RBFIE bit: Receive buffer full interrupt
I2C*n*INTE.TBEIE bit: Transmit buffer empty interrupt

13 Clock Timer (CT)

13.1 Overview

CT is a clock timer that counts a 256 Hz clock. The main features of CT are outlined below.

- Consists of an 8-bit binary counter that counts a 256 Hz clock.
- The counter data (128 Hz to 1 Hz) can be read via software.
- 32 Hz, 8 Hz, 2 Hz, and 1 Hz interrupts can be generated.

Figure 13.1.1 shows the configuration of CT.

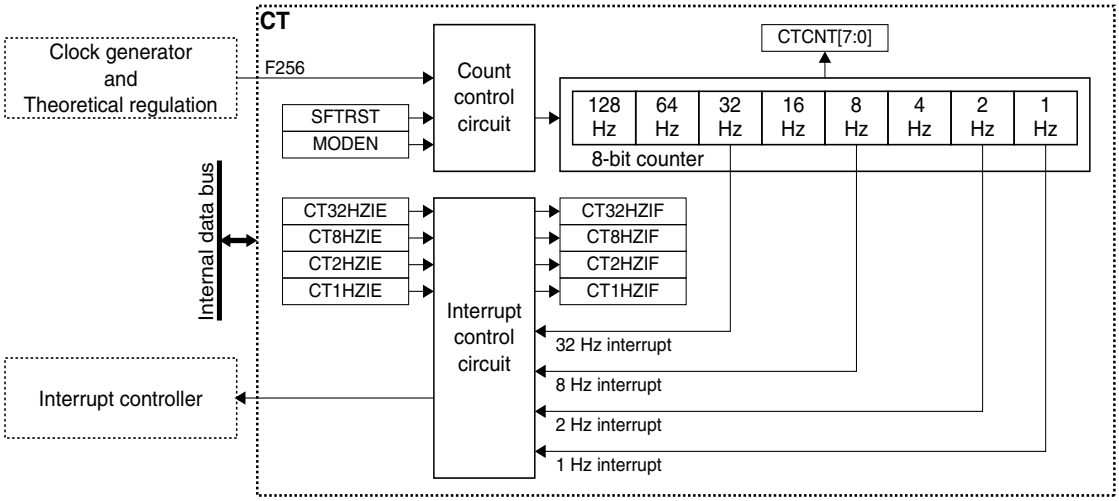


Figure 13.1.1 CT Configuration

13.2 Clock Settings

CT uses F256 (256 Hz regulated clock), which is generated by the clock generator from OSC1 as the clock source, as its operating clock. CT is operable when OSC1 is enabled.
When using CT during SLEEP mode, the clock must be configured so that it will keep supplying by writing 0 to the CLGOSC.OSC1SLPC bit.

13.3 Operations

Follow the sequences shown below to start counting of CT and to read the counter.

Count start

1. Write 1 to the CTCTL.SFTRST bit to reset CT.
2. Write 1 to the CT interrupt flags in the CTINTF register to clear them.
3. Write 1 to the interrupt enable bits in the CTINTE register to enable CT interrupts.
4. Write 1 to the CTCTL.MODEN bit to start CT count up operation.

Note: The timer switches to run/stop status synchronized with the count clock falling edge after data is written to the CTCTL.MODEN bit. When 0 is written to the CTCTL.MODEN bit, the timer stops after counting an additional "+1." 1 is retained for the CTCTL.MODEN bit reading until the timer actually stops. Figure 13.3.1 shows the run/stop control timing chart.

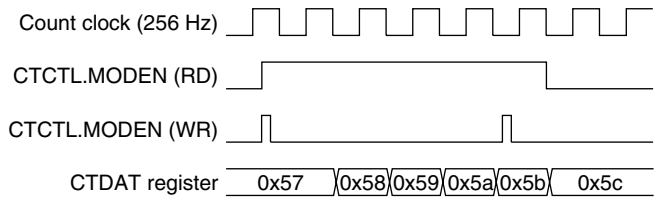


Figure 13.3.1 Run/Stop Control Timing Chart

Counter read

1. Read the count value from the CTDAT.CTCNT[7:0] bits.
2. Read again.
   i. If the two read values are the same, assume that the count values are read correctly.
   ii. If different values are read, perform reading once more and compare the read value with the previous one.

13.4 Interrupts

CT has a function to generate the interrupts shown in Table 13.4.1.

Table 13.4.1 CT Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
32 Hz	CTINTF.CT32HZIF	Signal falling edge (See Figure 13.4.1.)	Writing 1
8 Hz	CTINTF.CT8HZIF		Writing 1
2 Hz	CTINTF.CT2HZIF		Writing 1
1 Hz	CTINTF.CT1HZIF		Writing 1

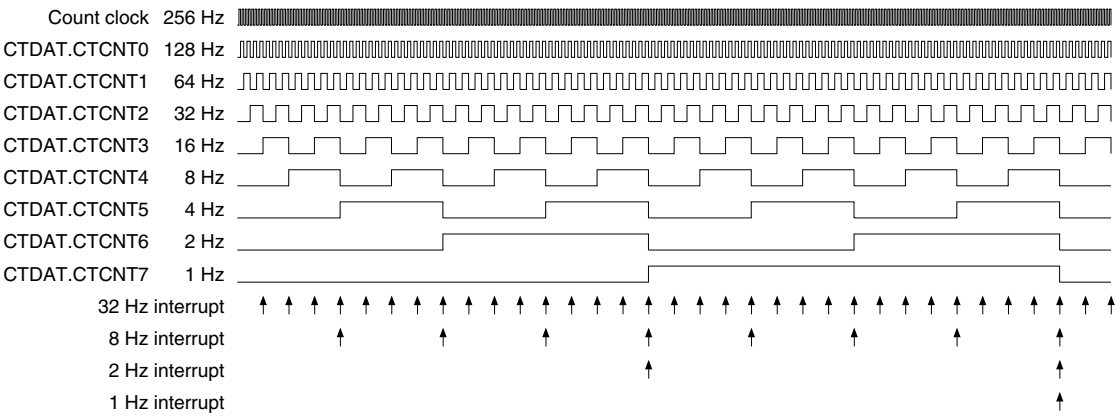


Figure 13.4.1 CT Interrupt Timings

CT provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the “Interrupt Controller” chapter.

13.5 Control Registers

CT Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CTCTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	SFTRST	0	H0	W	
	0	MODEN	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 SFTRST

This bit clears the counter to 0x00.

1 (W): Reset

0 (W): Ineffective

0 (R): Always 0 when being read

When the timer in running status is reset, it continues counting from count 0x00.

Bit 0 MODEN

This bit starts/stops counting of CT.

1 (R/W): Running/start control

0 (R/W): Idle/stop control

When CT stops counting by writing 0 to this bit, the counter retains the value when it stopped. Writing 1 to this bit again resumes counting from the value retained.

CT Counter Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CTDAT	15–8	–	0x00	–	R	–
	7–0	CTCNT[7:0]	0x00	H0/S0	R	

Bits 15–8 Reserved

Bits 7–0 CTCNT[7:0]

The counter data can be read through these bits.

The bits correspond to various frequencies, as follows:

D7: 1 Hz

D6: 2 Hz

D5: 4 Hz

D4: 8 Hz

D3: 16 Hz

D2: 32 Hz

D1: 64 Hz

D0: 128 Hz

Note: The correct counter value may not be read out (reading is unstable) if the register is read while counting is underway. Read the counter register twice in succession and treat the value as valid if the values read are identical.

CT Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CTINTF	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3	CT32HZIF	0	H0	R/W	Cleared by writing 1.
	2	CT8HZIF	0	H0	R/W	
	1	CT2HZIF	0	H0	R/W	
	0	CT1HZIF	0	H0	R/W	

Bits 15–4 Reserved

Bit 3 CT32HZIF
Bit 2 CT8HZIF
Bit 1 CT2HZIF
Bit 0 CT1HZIF

These bits indicate the CT interrupt cause occurrence status.
1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:
CTINTF.CT32HZIF bit: 32 Hz interrupt
CTINTF.CT8HZIF bit: 8 Hz interrupt
CTINTF.CT2HZIF bit: 2 Hz interrupt
CTINTF.CT1HZIF bit: 1 Hz interrupt

CT Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CTINTE	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3	CT32HZIE	0	H0	R/W	
	2	CT8HZIE	0	H0	R/W	
	1	CT2HZIE	0	H0	R/W	
	0	CT1HZIE	0	H0	R/W	

Bits 15–4 Reserved

Bit 3 CT32HZIE
Bit 2 CT8HZIE
Bit 1 CT2HZIE
Bit 0 CT1HZIE

These bits enable CT interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
CTINTE.CT32HZIE bit: 32 Hz interrupt
CTINTE.CT8HZIE bit: 8 Hz interrupt
CTINTE.CT2HZIE bit: 2 Hz interrupt
CTINTE.CT1HZIE bit: 1 Hz interrupt

14 Real-Time Clock (RTC)

14.1 Overview

RTC is a real-time clock that counts hours, minutes, and seconds. The main features of RTC are outlined below.

- Contains time counters (seconds, minutes, and hours).
- Either binary or BCD data can be read from and written to the counters.
- Capable of controlling the starting and stopping of the time-of-day clock.
- 24-hour or 12-hour mode is selectable.
- Can generate periodic interrupts (32 Hz, 8 Hz, 4 Hz, 1 Hz, 10 seconds, 1 minute, 10 minutes, 1 hour, half-day, and 1 day).

Figure 14.1.1 shows the configuration of RTC.

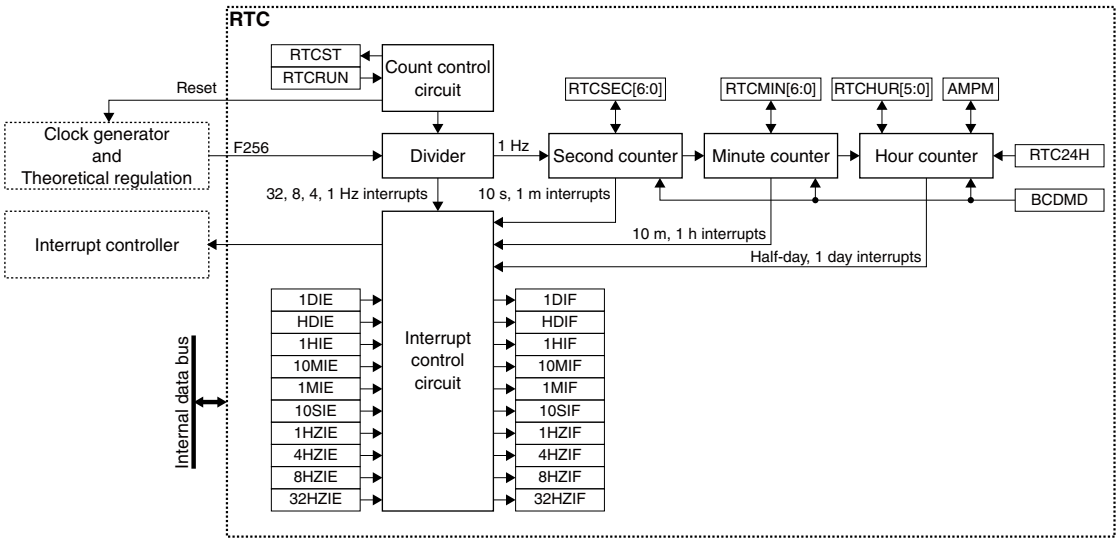


Figure 14.1.1 RTC Configuration

14.2 Clock Settings

RTC uses F256 (256 Hz regulated clock), which is generated by the clock generator from OSC1 as the clock source, as its operating clock. RTC is operable when OSC1 is enabled.
When using RTC during SLEEP mode, the clock must be configured so that it will keep supplying by writing 0 to the CLGOSC.OSC1SLPC bit.

14.3 RTC Counters

The RTC contains three counters: second counter, minute counter and hour counter. The count values can be read out from the respective registers. Each counter can also be set to any desired time by writing data to the register.

Binary mode and BCD mode

The second/minute/hour counters can be configured to binary mode (RTCCTL.BCDMD bit = 0) or BCD mode (RTCCTL.BCDMD bit = 1).

Second counter

This 7-bit counter counts from 0 to 59 seconds synchronously with the 1 Hz signal derived from the divider. The counter is cleared to 0 when it reaches 60 seconds and outputs a carry over of 1 to the minute counter. The counter data can be read/written using the RTCMIN.RTCSEC[6:0] bits.

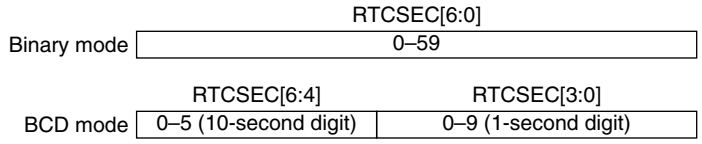


Figure 14.3.1 Second Counter

Minute counter

This 7-bit counter counts from 0 to 59 minutes with 1 carried over from the second counter. The counter is cleared to 0 when it reaches 60 minutes and outputs a carry over of 1 to the hour counter. The counter data can be read/written using the RTCMIN.RTCMIN[6:0] bits.

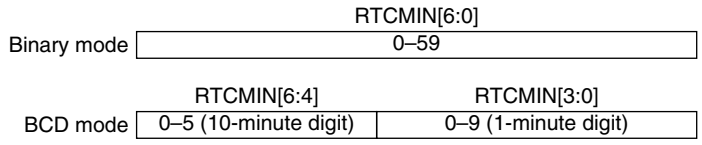


Figure 14.3.2 Minute Counter

Hour counter

This 6-bit counter counts from 0 to 23 o'clock (24H mode, RTCCTL.RTC24H bit = 0) or from 1 to 12 (12H mode, RTCCTL.RTC24H bit = 1) with 1 carried over from the minute counter. The counter data can be read/written using the RTCHUR.RTCHUR[5:0] bits.

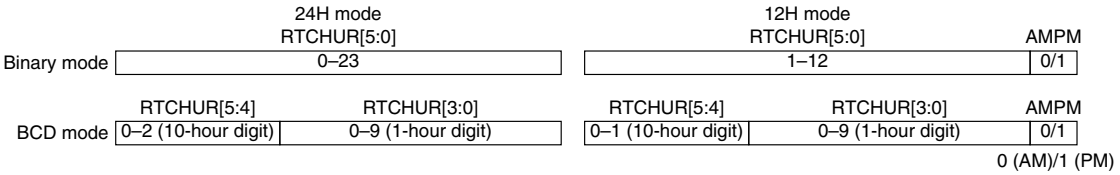


Figure 14.3.3 Hour Counter

Table 14.3.1 Hour Counter Values

Time	24H mode		12H mode		
	RTCHUR[5:0] (binary)	RTCHUR[5:0] (BCD)	RTCHUR[5:0] (binary)	RTCHUR[5:0] (BCD)	AMPM
0 o'clock (12am)	0x0	0x00	0xc	0x12	0
1 o'clock (1am)	0x1	0x01	0x1	0x01	0
2 o'clock (2am)	0x2	0x02	0x2	0x02	0
3 o'clock (3am)	0x3	0x03	0x3	0x03	0
4 o'clock (4am)	0x4	0x04	0x4	0x04	0
5 o'clock (5am)	0x5	0x05	0x5	0x05	0
6 o'clock (6am)	0x6	0x06	0x6	0x06	0
7 o'clock (7am)	0x7	0x07	0x7	0x07	0
8 o'clock (8am)	0x8	0x08	0x8	0x08	0
9 o'clock (9am)	0x9	0x09	0x9	0x09	0
10 o'clock (10am)	0xa	0x10	0xa	0x10	0
11 o'clock (11am)	0xb	0x11	0xb	0x11	0
12 o'clock (12pm)	0xc	0x12	0xc	0x12	1
13 o'clock (1pm)	0xd	0x13	0x1	0x01	1
14 o'clock (2pm)	0xe	0x14	0x2	0x02	1
15 o'clock (3pm)	0xf	0x15	0x3	0x03	1
16 o'clock (4pm)	0x10	0x16	0x4	0x04	1
17 o'clock (5pm)	0x11	0x17	0x5	0x05	1
18 o'clock (6pm)	0x12	0x18	0x6	0x06	1
19 o'clock (7pm)	0x13	0x19	0x7	0x07	1
20 o'clock (8pm)	0x14	0x20	0x8	0x08	1
21 o'clock (9pm)	0x15	0x21	0x9	0x09	1
22 o'clock (10pm)	0x16	0x22	0xa	0x10	1
23 o'clock (11pm)	0x17	0x23	0xb	0x11	1

14.4 Operations

14.4.1 Time Setting

Follow the sequence shown below to set time to RTC.

1. Set the following RTCCTL register bits:
   - Write 0 to the RTCCTL.RTCRUN bit. (Stop RTC)
   - RTCCTL.BCDMD bit (Select binary/BCD mode)
   - RTCCTL.RTC24H bit (Select 24H/12H mode)
2. Check to see if the RTCCTL.RTCST bit = 0 (RTC is idle). If the RTCCTL.RTCST bit = 1 (RTC is operating), wait until it is set to 0.
3. Write a time to the control bits listed below.
   - RTCMIN.RTCSEC[6:0] bit (Second)
   - RTCMIN.RTCMIN[6:0] bit (Minute)
   - RTCHUR.RTCHUR[5:0] bit (Hour)
   - RTCHUR.AMPM bit (AM/PM) (effective when RTCCTL.RTC24H bit = 1)
4. Write 1 to the interrupt flags in the RTCINTF register. (Clear interrupt flags)
5. Write 1 to the interrupt enable bits in the RTCINTE register. (Enable interrupts)
6. Write 1 to the RTCCTL.RTCRUN bit. (Start RTC)

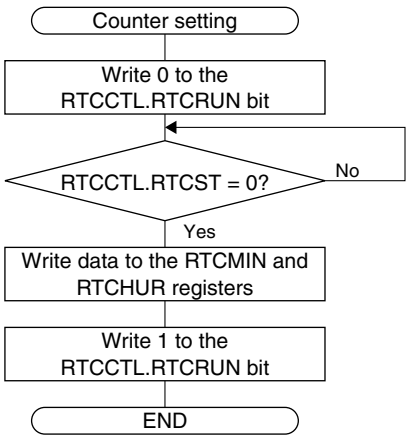


Figure 14.4.1.1 Counter Setting Procedure

Notes:
- An initial reset does not initialize the second/minute/hour counter values. Be sure to initialize the counters via software.
- Do not set the counters while the RTC is running (RTCCTL.RTCST bit = 1), as proper settings to the counters cannot be guaranteed.
- Counter values to be set must be within the effective range according to binary/BCD mode. The counter will be undefined if a value out of the range is written.
- Depending on the value set, an interrupt may occur immediately after starting the RTC.
- The divider that generates F256 (256 Hz regulated clock) is reset when the RTC starts running (when 1 is written to the RTCCTL.RTCRUN bit). This affects the count operations of other timers that use F256, as new F256 cycle begins from that point.
- After an initial reset, the RTCCTL.RTCRUN bit is set to 0 and the RTC idles. The OSC1 oscillator circuit is also idle. Therefore, resetting the IC suspends the RTC operation for the period shown below.

  RTC idle time = [#RESET = low period] + [Boot clock oscillation stabilization time] + [Time until OSC1 is started] + [OSC1 oscillation stabilization time] + [Time until RTC is restarted]

14.4.2 Time Read

Read procedure 1

1. Read the RTCMIN and RTCHUR registers.
2. Read again.
   i. If the two read values are the same, assume that the count values are read correctly.
   ii. If different values are read, perform reading once more and compare the read value with the previous one.

Read procedure 2

1. Wait for a 1 Hz interrupt (or 10 seconds to one day interrupt) to occur.
2. Read the RTCMIN and RTCHUR registers within one second from Step 1.

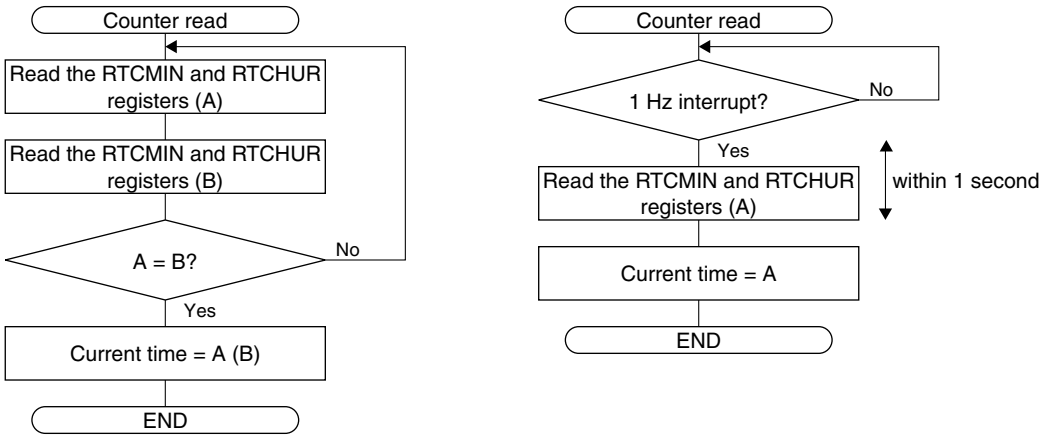


Figure 14.4.2.1 Counter Read Procedure

14.5 Interrupts

RTC has a function to generate the interrupts shown in Table 14.5.1.

Table 14.5.1 RTC Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
One day	RTCINTF.1DIF	Hour counter = 23→0 (24H mode) Hour counter = PM11→AM12 (12H mode)	Writing 1
Half-day	RTCINTF.HDIF	Hour counter = 11→12, 23→0 (24H mode) Hour counter = AM11→PM12, PM11→AM12 (12H mode)	Writing 1
1 hour	RTCINTF.1HIF	Minute counter = 59→0	Writing 1
10 minutes	RTCINTF.10MIF	Minute counter = 9→10, 19→20, 29→30, 39→40, 49→50, 59→0	Writing 1
1 minute	RTCINTF.1MIF	Second counter = 59→0	Writing 1
10 seconds	RTCINTF.10SIF	Second counter = 9→10, 19→20, 29→30, 39→40, 49→50, 59→0	Writing 1
1 Hz	RTCINTF.1HZIF	Divider 1 Hz signal cycle	Writing 1
4 Hz	RTCINTF.4HZIF	Divider 4 Hz signal cycle	Writing 1
8 Hz	RTCINTF.8HZIF	Divider 8 Hz signal cycle	Writing 1
32 Hz	RTCINTF.32HZIF	Divider 32 Hz signal cycle	Writing 1

The RTC provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the “Interrupt Controller” chapter.

14.6 Control Registers

RTC Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCCTL	15–9	–	0x00	–	R	–
	8	RTCST	0	H0	R	
	7–6	–	0x0	–	R	
	5	BCDMD	0	H0	R/W	
	4	RTC24H	0	H0	R/W	
	3–1	–	0x0	–	R	
	0	RTCRUN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 RTCST

This bit indicates the RTC operating status.
1 (R): Running
0 (R): Stop

The RTCCTL.RTCST bit goes 1 when the RTC starts counting by writing 1 to the RTCCTL.RTCRUN bit and it reverts to 0 when the count operation is actually stopped after 0 is written to the RTCCTL. RTCRUN bit. When setting counter values, write 0 to the RTCCTL.RTCRUN bit and make sure that the RTCCTL.RTCST bit is cleared to 0 before writing data.

Bits 7–6 Reserved

Bit 5 BCDMD

This bit sets the second, minute, and hour counters into binary or BCD mode.
1 (R/W): BCD mode
0 (R/W): Binary mode

See "RTC Counters" for the configuration of the counter in each mode.

Bit 4 RTC24H

This bit sets the hour counter to 24H mode or 12H mode.
1 (R/W): 12H mode
0 (R/W): 24H mode

This selection changes the count range of the hour counter. Note, however, that the counter value is not updated automatically, therefore, it must be programmed again.

Note: Be sure to avoid writing to this bit when the RTCCTL.RTCRUN bit = 1.

Bits 3–1 Reserved

Bit 0 RTCRUN

This bit starts/stops the RTC.
1 (R/W): Start
0 (R/W): Stop

When the RTC stops counting by writing 0 to this bit, the counter retains the value when it was stopped. Writing 1 to this bit again resumes counting from the value retained.
Writing 1 to the RTCCTL.RTCRUN bit resets the OSC1A divider in the clock generator.

RTC Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCINTE	15–10	–	0x00	–	R	–
	9	1DIE	0	H0	R/W	
	8	HDIE	0	H0	R/W	
	7	1HIE	0	H0	R/W	
	6	10MIE	0	H0	R/W	
	5	1MIE	0	H0	R/W	
	4	10SIE	0	H0	R/W	
	3	1HZIE	0	H0	R/W	
	2	4HZIE	0	H0	R/W	
	1	8HZIE	0	H0	R/W	
	0	32HZIE	0	H0	R/W	

Bits 15–10 Reserved

Bit 9 1DIE
Bit 8 HDIE
Bit 7 1HIE
Bit 6 10MIE
Bit 5 1MIE
Bit 4 10SIE
Bit 3 1HZIE
Bit 2 4HZIE
Bit 1 8HZIE
Bit 0 32HZIE

These bits enable RTC interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
RTCINTE.1DIE bit: One day interrupt
RTCINTE.HDIE bit: Half-day interrupt
RTCINTE.1HIE bit: 1 hour interrupt
RTCINTE.10MIE bit: 10 minutes interrupt
RTCINTE.1MIE bit: 1 minute interrupt
RTCINTE.10SIE bit: 10 seconds interrupt
RTCINTE.1HZIE bit: 1 Hz interrupt
RTCINTE.4HZIE bit: 4 Hz interrupt
RTCINTE.8HZIE bit: 8 Hz interrupt
RTCINTE.32HZIE bit: 32 Hz interrupt

RTC Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCINTF	15–10	–	0x00	–	R	–
	9	1DIF	0	H0	R/W	Cleared by writing 1.
	8	HDIF	0	H0	R/W	
	7	1HIF	0	H0	R/W	
	6	10MIF	0	H0	R/W	
	5	1MIF	0	H0	R/W	
	4	10SIF	0	H0	R/W	
	3	1HZIF	0	H0	R/W	
	2	4HZIF	0	H0	R/W	
	1	8HZIF	0	H0	R/W	
	0	32HZIF	0	H0	R/W	

Bits 15–10 Reserved

Bit 9 **1DIF**
Bit 8 **HDIF**
Bit 7 **1HIF**
Bit 6 **10MIF**
Bit 5 **1MIF**
Bit 4 **10SIF**
Bit 3 **1HZIF**
Bit 2 **4HZIF**
Bit 1 **8HZIF**
Bit 0 **32HZIF**

These bits indicate the RTC interrupt cause occurrence status.
1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:
RTCINTF.1DIF bit: One day interrupt
RTCINTF.HDIF bit: Half-day interrupt
RTCINTF.1HIF bit: 1 hour interrupt
RTCINTF.10MIF bit: 10 minutes interrupt
RTCINTF.1MIF bit: 1 minute interrupt
RTCINTF.10SIF bit: 10 seconds interrupt
RTCINTF.1HZIF bit: 1 Hz interrupt
RTCINTF.4HZIF bit: 4 Hz interrupt
RTCINTF.8HZIF bit: 8 Hz interrupt
RTCINTF.32HZIF bit: 32 Hz interrupt

RTC Minute/Second Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCMIN	15	–	0	–	R	–
	14–8	RTCMIN[6:0]	x	–	R/W	
	7	–	0	–	R	
	6–0	RTCSEC[6:0]	x	–	R/W	

Note: Be sure to avoid writing to this register while the RTCCTL.RTCST bit = 1.

Bit 15 **Reserved**

Bits 14–8 **RTCMIN[6:0]**

These bits are used to read and write data from/to the minute counter.
For the configuration of the minute counter, see "RTC Counters." For the counter read and write procedures, see "Operations."

Bit 7 **Reserved**

Bits 6–0 **RTCSEC[6:0]**

These bits are used to read and write data from/to the second counter.
For the configuration of the second counter, see "RTC Counters." For the counter read and write procedures, see "Operations."

RTC Hour Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCHUR	15–8	–	0x00	–	R	–
	7	AMPM	x	–	R/W	
	6	–	0	–	R	
	5–0	RTCHUR[5:0]	x	–	R/W	

Note: Be sure to avoid writing to this register while the RTCCTL.RTCST bit = 1.

Bits 15–8 Reserved

Bit 7 AMPM

This bit is used to set and read A.M. or P.M. data in 12H mode (RTCCTL.RTC24H bit = 1).
1 (R/W): P.M.
0 (R/W): A.M.

In 24H mode (RTCCTL.RTC24H bit = 0), this bit is fixed at 0. In this case, do not write 1 to the RTCHUR.AMPM bit.

Note: The RTCHUR.AMPM bit will be fixed at 0 immediately after the RTCCTL.RTC24H bit is changed from 12-hour mode to 24-hour mode.

Bit 6 Reserved

Bits 5–0 RTCHUR[5:0]

These bits are used to read and write data from/to the hour counter.
For the configuration of the hour counter, see "RTC Counters." For the counter read and write procedures, see "Operations."

15 Theoretical Regulation (TR)

15.1 Overview

TR is a theoretical regulation function that theoretically corrects time clock errors due to deviation in oscillation frequencies. The main features of TR are outlined below.

- Adjusts the OSC1A clock (32.768 kHz Typ.).
  (Note that other oscillation clocks cannot be adjusted.)
- Adjustable range: -31/32,768 to +32/32,768 seconds in a correction operation
- Allows software to execute theoretical regulation at any time.

Figure 15.1.1 shows the configuration of TR.

Peripheral circuits that can use the regulated clock (F256) in this IC
- Real-time clock
- Clock timer
- 16-bit PWM timer

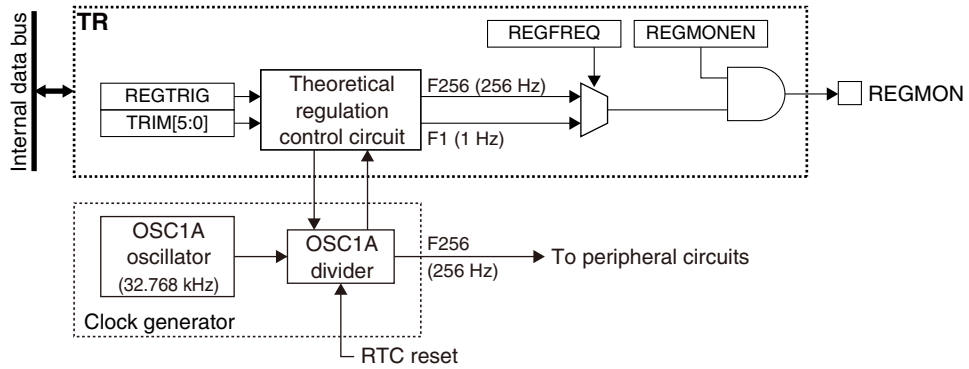


Figure 15.1.1 TR Configuration

15.2 Output Pin

Table 15.2.1 shows the TR output pin.

Table 15.2.1 TR Output Pin

Pin name	I/O*	Initial status*	Function
REGMON	O	O (L)	Theoretical regulation clock monitor output

* Indicates the status when the pin is configured for TR.

If the port is shared with the REGMON pin and other functions, the REGMON function must be assigned to the port before starting the monitor output. For more information, refer to the "I/O Ports" chapter.

15.3 Operations

15.3.1 Executing Theoretical Regulation

Follow the sequences shown below to execute theoretical regulation.

1. Set the correction value to the TRCTL.TRIM[5:0] bits.
2. Generate interrupts in the theoretical regulation execution cycles using a timer to perform theoretical regulation periodically.
3. Write 1 to the TRCTL.REGTRIG bit (using the interrupt handler in Step 2). (Execute theoretical regulation)

Correction value

The correction value (-31/32,768 to +32/32,768 seconds) for theoretical regulation is specified using the TRCTL.TRIM[5:0] bits.

Table 15.3.1.1 Correction Value Setting Example

TRCTL.TRIM[5:0] bits	Amount of correction/ one adjustment (n/32,768 seconds)	Rate * [seconds/day]	TRCTL.TRIM[5:0] bits	Amount of correction/ one adjustment (n/32,768 seconds)	Rate * [seconds/day]
0x20	-31	+8.174	0x00	+1	-0.264
0x21	-30	+7.910	0x01	+2	-0.527
0x22	-29	+7.646	0x02	+3	-0.791
0x23	-28	+7.383	0x03	+4	-1.055
0x24	-27	+7.119	0x04	+5	-1.318
0x25	-26	+6.855	0x05	+6	-1.582
0x26	-25	+6.592	0x06	+7	-1.846
0x27	-24	+6.328	0x07	+8	-2.109
0x28	-23	+6.064	0x08	+9	-2.373
0x29	-22	+5.801	0x09	+10	-2.637
0x2a	-21	+5.537	0x0a	+11	-2.900
0x2b	-20	+5.273	0x0b	+12	-3.164
0x2c	-19	+5.010	0x0c	+13	-3.428
0x2d	-18	+4.746	0x0d	+14	-3.691
0x2e	-17	+4.482	0x0e	+15	-3.955
0x2f	-16	+4.219	0x0f	+16	-4.219
0x30	-15	+3.955	0x10	+17	-4.482
0x31	-14	+3.691	0x11	+18	-4.746
0x32	-13	+3.428	0x12	+19	-5.010
0x33	-12	+3.164	0x13	+20	-5.273
0x34	-11	+2.900	0x14	+21	-5.537
0x35	-10	+2.637	0x15	+22	-5.801
0x36	-9	+2.373	0x16	+23	-6.064
0x37	-8	+2.109	0x17	+24	-6.328
0x38	-7	+1.846	0x18	+25	-6.592
0x39	-6	+1.582	0x19	+26	-6.855
0x3a	-5	+1.318	0x1a	+27	-7.119
0x3b	-4	+1.055	0x1b	+28	-7.383
0x3c	-3	+0.791	0x1c	+29	-7.646
0x3d	-2	+0.527	0x1d	+30	-7.910
0x3e	-1	+0.264	0x1e	+31	-8.174
0x3f	0	0	0x1f	+32	-8.438

* Rates when theoretical regulation is executed in 10-second cycles
(= -(n/32,768) × 6 [times/minute] × 60 [minutes] × 24 [hours])

The correction value (0x00 to 0x3f) should be written to the Flash memory or read the switch settings via the I/O ports and set it to the TRCTL.TRIM[5:0] bits.

Theoretical regulation operations

Writing 1 to the TRCTL.REGTRIG bit extends or reduces the cycle time of the 256 Hz clock output by the OSC1A divider for the regulation value specified by the TRCTL.TRIM[5:0] bits. Theoretical regulation is performed only once by writing 1 to the TRCTL.REGTRIG bit. To perform theoretical regulation periodically, use a timer interrupt handler to write 1 to the TRCTL.REGTRIG bit.
The corrected 256 Hz clock (F256) will be supplied to the peripheral circuits that use the regulated clock.

Note: Use an interrupt from a timer that runs with the regulated clock (F256) to execute theoretical regulation. An interrupt from the timer that runs all the time should be used to reduce current consumption.

15.3.2 Regulated Clock External Monitor

Either the 256 Hz (F256) or 1 Hz (F1) corrected clock can be output from the REGMON pin for monitoring. The control procedure is shown below.

Monitoring 256 Hz clock (F256)

1. Activate at least one peripheral circuit that uses F256 (the operating clock must be configured to F256).
2. Set the TRCTL.REGFREQ bit to 0. (Select 256 Hz as the output frequency)
3. Set the TRCTL.REGMONEN bit to 1. (Start clock monitor output)

Monitoring 1 Hz clock (F1)

1. Activate the real-time clock.
2. Set the TRCTL.REGFREQ bit to 1. (Select 1 Hz as the output frequency)
3. Set the TRCTL.REGMONEN bit to 1. (Start clock monitor output)

The clock is output from the REGMON pin by setting the TRCTL.REGMONEN bit to 1. Setting the TRCTL.REGMONEN bit to 0 stops the clock output and the REGMON pin goes low (Vss) level.

15.4 Control Register

Theoretical Regulation Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TRCTL	15–10	–	0x00	–	R	–
	9	REGFREQ	0	H0	R/W	
	8	REGMONEN	0	H0	R/W	
	7	REGTRIG	0	H0	W	Always read as 0.
	6	–	0	–	R	–
	5–0	TRIM[5:0]	0x00	H0	R/W	

Bits 15–10 Reserved

Bit 9 REGFREQ

This bit selects the frequency of the regulated clock to be output from the REGMON pin for monitoring.

1 (R/W): 1 Hz (F1)
0 (R/W): 256 Hz (F256)

Bit 8 REGMONEN

This bit controls the regulated clock monitor output.

1 (R/W): Enable
0 (R/W): Disable

Note: Before the monitor output can be started, a peripheral circuit that uses the regulated clock must be activated (see “Regulated Clock External Monitor”).

Bit 7 REGTRIG

This bit executes theoretical regulation.

1 (W): Trigger (Theoretical regulation is executed only once.)
0 (W): Ineffective
0 (R): Always 0 when being read

Note: A maximum 16.6 ms of delay occurs before theoretical regulation actually starts after writing to the TRCTL.REGTRIG bit. Writing 1 to the TRCTL.REGTRIG bit in this period is ineffective, so to write 1 to the TRCTL.REGTRIG bit successively, an interval at least 16.6 ms is necessary between writings.

Bit 6 Reserved

Bits 5–0 TRIM[5:0]

These bits specify the correction value (-31/32,768 to +32/32,768 seconds) for theoretical regulation.

16 16-bit PWM Timers (T16A3)

16.1 Overview

T16A3 is a 16-bit PWM timer that consists of counter blocks and comparator/capture blocks. The features of T16A3 are listed below.

- Counter block
  - 16-bit up counter
  - A clock source and a clock division ratio for generating the count clock are selectable in each channel.
  - One-shot mode (counting for one cycle configured) and repeat mode (counting continuously until stopped via software)
  - Half-clock mode (Counting both clock rising and falling edges achieves high PWM waveform accuracy.)
  - Event counter function using an external clock
- Comparator/capture block
  - The comparator compares the counter value with two specified values to generate interrupt signals and a PWM waveform. (Can be used as an interval timer, PWM waveform generator, and external event counter.)
  - The capture unit captures counter values using external trigger signals and generates interrupts. (Can be used to measure external event periods.)
  - Multi-comparator/capture mode (One counter can be used with two or more comparator/capture blocks connected.)

Figure 16.1.1 shows the T16A3 configuration.

Channel configuration in this IC

• Number of channels:	2 channels (Ch.0 and Ch.1)	
• Event counter function:	Ch.0	EXCL0 pin input
	Ch.1	EXCL1 pin input
• Timer generating signal output:	Ch.0	Two outputs with the TOUTA0 and TOUTB0 pins
	Ch.1	Two outputs with the TOUTA1 and TOUTB1 pins
• Capture signal input:	Ch.0	Two inputs with CAPA0 and CAPB0 pins
	Ch.1	Two inputs with CAPA1 and CAPB1 pins
• Half-clock mode:	Ch.0	Available
	Ch.1	Available
• Multi-comparator/capture mode:	Ch.0	Available
	Ch.1	Available

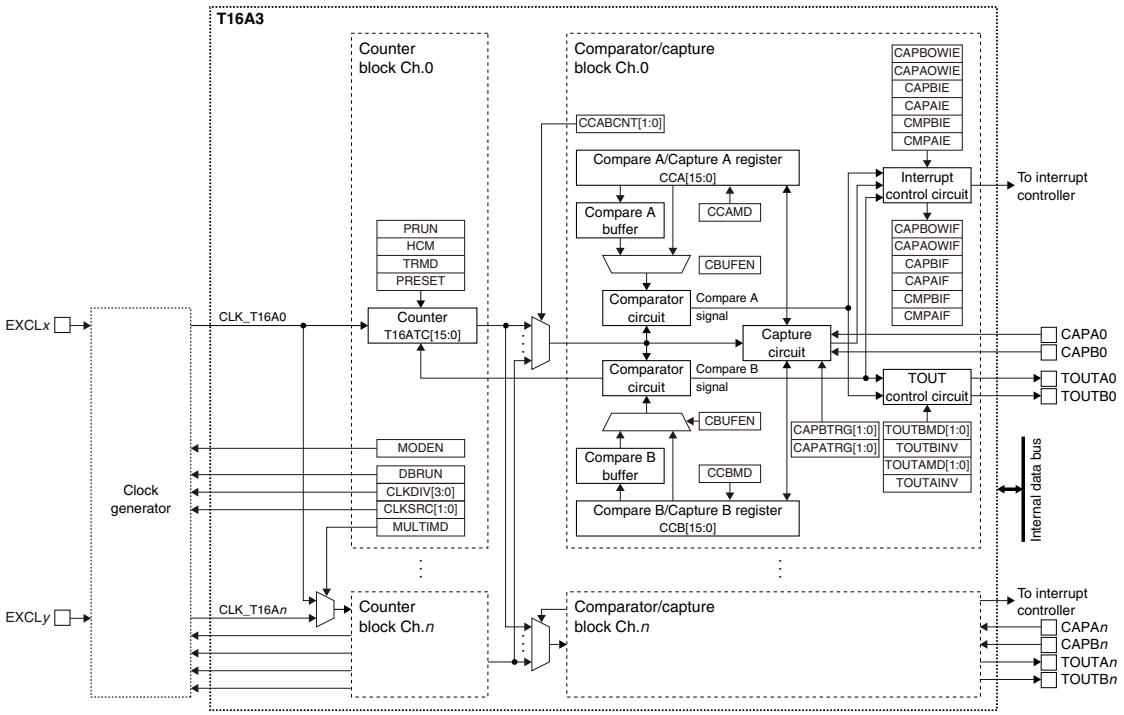


Figure 16.1.1 T16A3 Configuration

16.2 Input/Output Pins

Table 16.2.1 lists the T16A3 pins.

Table 16.2.1 List of T16A3 Pins

Pin name	I/O*	Initial status*	Function
EXCL*m*	I	I (Hi-Z)	External clock input
TOUTA*n*/CAPA*n*, TOUTB*n*/CAPB*n*	O or I	O (Low)	TOUTA/B signal output (in comparator mode) or capture A/B trigger signal input (in capture mode)

* Indicates the status when the pin is configured for T16A3.

If the port is shared with the T16A3 pin and other functions, the T16A3 input/output function must be assigned to the port before activating T16A3. For more information, refer to the "I/O Ports" chapter.

16.3 Clock Settings

16.3.1 T16A3 Operating Clock

When using T16A3 Ch.*n*, the T16A3 Ch.*n* operating clock CLK_T16A*n* must be supplied to T16A3 Ch.*n* from the clock generator. The CLK_T16A*n* supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).

   When an external clock is used, select the EXCL*m* pin function (refer to the "I/O Ports" chapter).

2. Set the following T16A*n*CLK register bits:
   - T16A*n*CLK.CLKSRC[1:0] bits (Clock source selection)
   - T16A*n*CLK.CLKDIV[3:0] bits (Clock division ratio selection = Clock frequency setting)

16.3.2 Clock Supply in SLEEP Mode

When using T16A3 during SLEEP mode, the T16A3 operating clock CLK_T16A*n* must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_T16A*n* clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_T16A*n* clock source is 1, the CLK_T16A*n* clock source is deactivated during SLEEP mode and T16A3 stops with the register settings and counter value maintained at those before entering SLEEP mode. After the CPU returns to normal mode, CLK_T16A*n* is supplied and the T16A3 operation resumes.

16.3.3 Clock Supply in DEBUG Mode

The CLK_T16A*n* supply during DEBUG mode should be controlled using the T16A*n*CLK.DBRUN bit.
The CLK_T16A*n* supply to T16A3 Ch.*n* is suspended when the CPU enters DEBUG mode if the T16A*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_T16A*n* supply resumes. Although T16A3 Ch.*n* stops operating when the CLK_T16A*n* supply is suspended, the counter and registers retain the status before DEBUG mode was entered. If the T16A*n*CLK.DBRUN bit = 1, the CLK_T16A*n* supply is not suspended and T16A3 Ch.*n* will keep operating in DEBUG mode.

16.3.4 Event Counter Clock

The channel that supports the event counter function uses the EXCL*m* pin, not EXOSC, as the clock input pin when external clock is selected using the T16A*n*CLK.CLKSRC[1:0] bits.
The counter counts up at the rising edge of the EXCL*m* input signal.

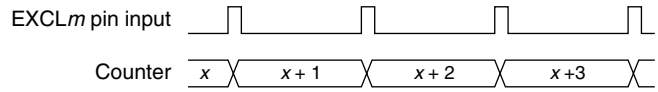


Figure 16.3.4.1 Count Up Timing

16.4 Operations

16.4.1 Initialization

T16A3 Ch.*n* should be initialized and started counting with the procedure shown below. Perform initial settings for comparator mode when using T16A3 as an interval timer, PWM waveform generator, or external event counter. Perform initial settings for capture mode when using T16A3 to measure external event periods.

Initial settings for comparator mode

1. Configure the T16A3 Ch.*n* operating clock.
2. Set the T16A0CLK.MULTIMD bit. (Select multi-comparator/capture or normal channel mode)
3. Set the following T16A*n*CCCTL register bits:
   - Set the T16A*n*CCCTL.CCAMD bit to 0. * (Set the T16A*n*CCA register to comparator mode)
   - Set the T16A*n*CCCTL.CCBMD bit to 0. * (Set the T16A*n*CCB register to comparator mode)

   * One of the T16A*n*CCA or T16A*n*CCB register can be set to capture mode.

   Set the following bits when the TOUTA output is used.
   - T16A*n*CCCTL.TOUTAINV bit (Select TOUT signal polarity)
   - T16A*n*CCCTL.TOUTAMD[1:0] bits (Select TOUT signal generation mode)

   Set the following bits when the TOUTB output is used.
   - T16A*n*CCCTL.TOUTBINV bit (Select TOUT signal polarity)
   - T16A*n*CCCTL.TOUTBMD[1:0] bits (Select TOUT signal generation mode)
4. Set the T16A*n*CCA and T16A*n*CCB registers. (Set compare data)
5. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the T16A*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the T16A*n*INTE register to 1. (Enable interrupts)

6. Set the following T16A*n*CTL register bits:
   - T16A*n*CTL.HCM bit (Select half-clock or normal clock mode)
   - T16A*n*CTL.CCABCNT[1:0] bits (Select counter channel)
   - T16A*n*CTL.CBUFEN bit (Enable/disable compare buffer)
   - T16A*n*CTL.TRMD bit (Select one-shot or repeat mode)
   - Set the T16A*n*CTL.PRESET bit to 1. (Reset counter)
   - Set the T16A*n*CTL.MODEN bit to 1. (Enable count operations)
   - Set the T16A*n*CTL.PRUN bit to 1. (Start counting)

Initial settings for capture mode

1. Configure the T16A3 Ch.*n* operating clock.
2. Set the T16A0CLK.MULTIMD bit. (Select multi-comparator/capture or normal channel mode)
3. Set the following T16A*n*CCCTL register bits:
   - Set the T16A*n*CCCTL.CCAMD bit to 1. * (Set the T16A*n*CCA register to capture mode)
   - Set the T16A*n*CCCTL.CCBMD bit to 1. * (Set the T16A*n*CCB register to capture mode)
   - T16A*n*CCCTL.CAPATRG[1:0] bits (Select capture A trigger edge)
   - T16A*n*CCCTL.CAPBTRG[1:0] bits (Select capture B trigger edge)

   * One of the T16A*n*CCA or T16A*n*CCB register can be set to comparator mode.
4. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the T16A*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the T16A*n*INTE register to 1. (Enable interrupts)
5. Set the following T16A*n*CTL register bits:
   - T16A*n*CTL.CCABCNT[1:0] bits (Select counter channel)
   - T16A*n*CTL.TRMD bit (Select one-shot or repeat mode)
   - Set the T16A*n*CTL.PRESET bit to 1. (Reset counter)
   - Set the T16A*n*CTL.MODEN bit to 1. (Enable count operations)
   - Set the T16A*n*CTL.PRUN bit to 1. (Start counting)

16.4.2 Counter Block Operations

The counter in each counter block channel is a 16-bit up counter that counts the selected operating clock (count clock).

Counter reset

Setting the T16A*n*CTL.PRESET bit to 1 clears the counter to 0. In comparator mode, the counter is also cleared to 0 by the compare B signal (generated when the counter value reaches the T16A*n*CCB register value).

Counting start

To start counting, set the T16A*n*CTL.MODEN bit to 1 (start clock supply) and the T16A*n*CTL.PRUN bit to 1 (start counting). The counting stop control depends on the count mode set.

Count mode (repeat mode and one-shot mode)

Each counter features two count modes: repeat mode and one-shot mode. The count mode can be selected using the T16A*n*CTL.TRMD bit.

Repeat mode (T16A*n*CTL.TRMD bit = 0)

This mode enables the counter to run continuously. Once the count starts, the counter continues running until 0 is written to the T16A*n*CTL.PRUN bit. The counter continues counting even if the counter returns to 0 due to a counter overflow or by the compare B signal. Select this mode to generate periodic interrupts at desired intervals, to measure pulse width or external event intervals, or to generate a timer output waveform.

One-shot mode (T16A*n*CTL.TRMD bit = 1)

This mode enables the counter to run for specified periods. The T16A*n*CTL.PRUN bit is cleared to 0 and the counter stops automatically as soon as the compare B signal is generated. Select this mode to stop the counter after an interrupt has occurred once, such as for checking a specific lapse of time.

Counter value read

The counter value can be read out from the T16A*n*TC.T16ATC[15:0] bits. However, since T16A3 operates on CLK_T16A*n*, one of the operations shown below is required to read correctly by the CPU.

- Read the counter value twice or more and check to see if the same value is read.
- Stop the timer and then read the counter value.

16.4.3 Comparator/Capture Block Operations

The comparator/capture block functions as a comparator to compare the counter value with the register value set or a capture unit to capture counter values using the external trigger signals.

Comparator/capture block operating mode (comparator mode and capture mode)

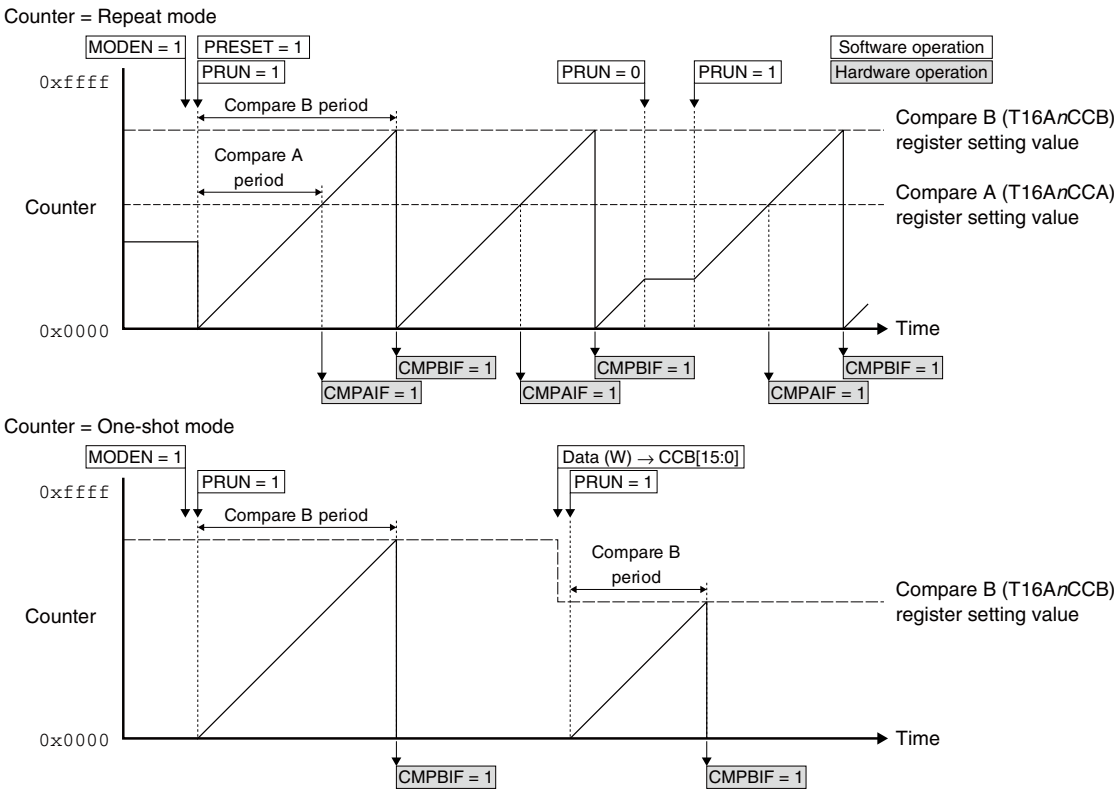
The T16A*n*CCA and T16A*n*CCB registers that are embedded in the comparator/capture block can be set to comparator mode or capture mode, individually.

Comparator mode (T16A*n*CCCTL.CCAMD/CCBMD bit = 0)

The comparator mode compares the counter value and the value set via software. It generates an interrupt and toggles the timer output signal level when the values are matched. The T16A*n*CCA and/or T16A*n*CCB registers function as the compare A and/or compare B registers that are used for setting compare values in this mode. Furthermore, the TOUTA*n*/CAPA*n* pin and the TOUTB*n*/CAPB*n* pin are configured to the TOUTA*n* pin and the TOUTB*n* pin, respectively.

When the counter reaches the value set in the T16A*n*CCA register during counting, the comparator asserts the compare A signal and issues a compare A interrupt request.

When the counter reaches the value set in the T16A*n*CCB register, the comparator asserts the compare B signal and issues a compare B interrupt request. At the same time, the counter is cleared to 0.



(Note that the T16A*n*INTF.CMPBIF/CMPAIF bit clearing operations via software are omitted from the figure.)

Figure 16.4.3.1 Operations in Comparator Mode

The time from counter = 0x0000 to occurrence of a compare A interrupt (compare A period) and the time to occurrence of a compare B interrupt (compare B period) can be calculated as follows:

$$\text{Compare A period} = \frac{(CCA + 1)}{ct\_clk}\ [s] \qquad \text{Compare B period} = \frac{(CCB + 1)}{ct\_clk}\ [s] \qquad \text{(Eq. 16.1)}$$

Where

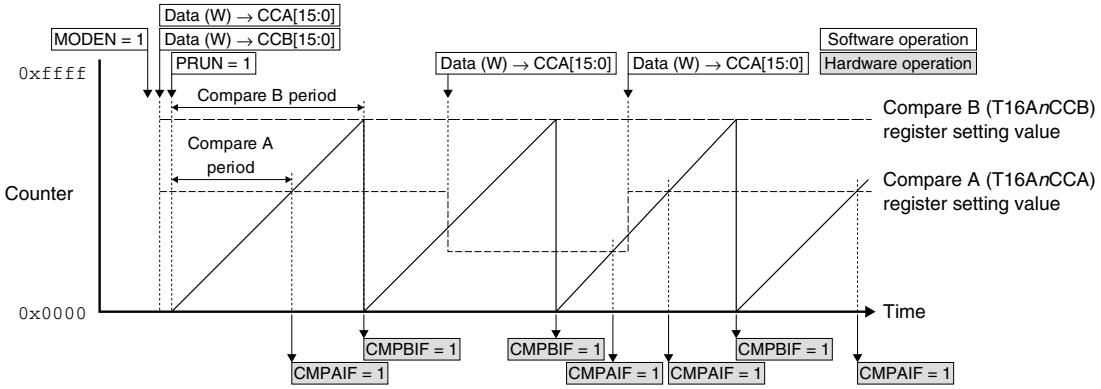
CCA: T16A*n*CCA register setting value (0 to 65,535)
CCB: T16A*n*CCB register setting value (0 to 65,535)
ct_clk: Count clock frequency [Hz]

The compare A and compare B signals are also used to generate a timer output waveform (TOUT). Refer to "TOUT Output Control" for more information.

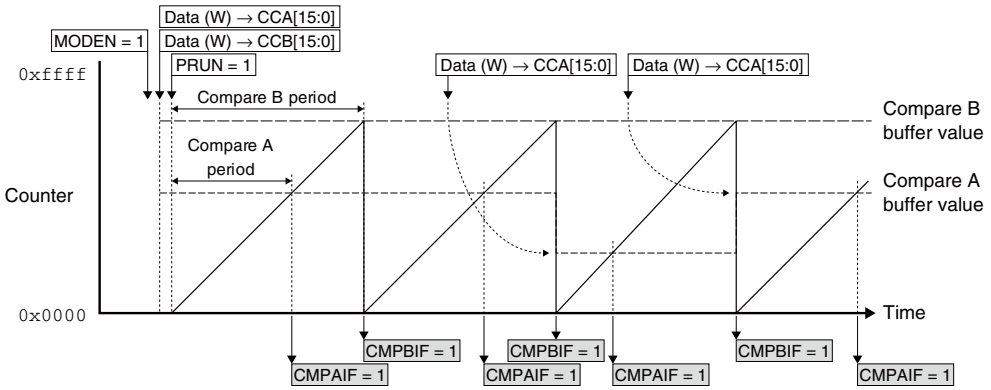
Compare buffers

The comparator provides the compare A and compare B buffers for comparison with the counter instead of the T16A*n*CCA and T16A*n*CCB registers. These compare buffers are used to synchronize the comparison data update timings with the counter operations. Setting the T16A*n*CTL.CBUFEN bit to 1 enables the compare buffers. The compare A and B signals will be generated by comparing the counter values with the compare A and B buffer values instead of the T16A*n*CCA and T16A*n*CCB register values. The T16A*n*CCA and T16A*n*CCB register values written via software are loaded to the compare A and compare B buffers when the compare B signal is generated (when the T16A*n*CTL.CBUFEN bit is set to 0, the T16A*n*CCA and T16A*n*CCB register values are loaded to the compare buffers simultaneously with writing).

When the compare buffers are not used (T16A*n*CTL.CBUFEN bit = 0)



When the compare buffers are used (T16A*n*CTL.CBUFEN bit = 1)



(Note that the T16A*n*INTF.CMPBIF/CMPAIF bit clearing operations via software are omitted from the figure.)

Figure 16.4.3.2 Compare Buffer Operations (Counter = Repeat Mode)

Capture mode (T16A*n*CCCTL.CCAMD/CCBMD bit = 1)

The capture mode captures the counter value when an external event such as a key entry occurs (at the specified edge of the CAPA*n*/CAPB*n* external input signal). In this mode, the T16A*n*CCA and/or T16A*n*CCB registers function as the capture A and/or capture B registers. Furthermore, the TOUTA*n*/CAPA*n* pin and the TOUTB*n*/CAPB*n* pin are configured to the CAPA*n* pin and the CAPB*n* pin, respectively.

The trigger edge of the signal can be selected using the T16A*n*CCCTL.CAPATRG[1:0] bits for capture A and the T16A*n*CCCTL.CAPBTRG[1:0] bits for capture B.

When a specified trigger edge is input during counting, the current counter value is loaded to the T16A*n*CCA or T16A*n*CCB register. At the same time the T16A*n*INTF.CAPAIF or CAPBIF bit is set. The interrupt occurred by this bit can be used to read the captured data from the T16A*n*CCA or T16A*n*CCB register. For example, external event cycles and pulse widths can be measured from the difference between two captured counter values read.

If the captured data stored in the T16A*n*CCA or T16A*n*CCB register is overwritten by the next trigger when the T16A*n*INTF.CAPAIF or CAPBIF bit has already been set, an overwrite error occurs (the T16A*n*INTF.CAPAOWIF or CAPBOWIF bit is set).

Notes:
- The correct captured data may not be obtained if the captured data is read at the same time the next value is being captured. Read the T16A*n*CCA or T16A*n*CCB register twice to check if the read data is correct as necessary.
- To capture counter data properly, both the high and low period of the CAPA*n*/CAPB*n* trigger signal must be longer than the source clock cycle time.

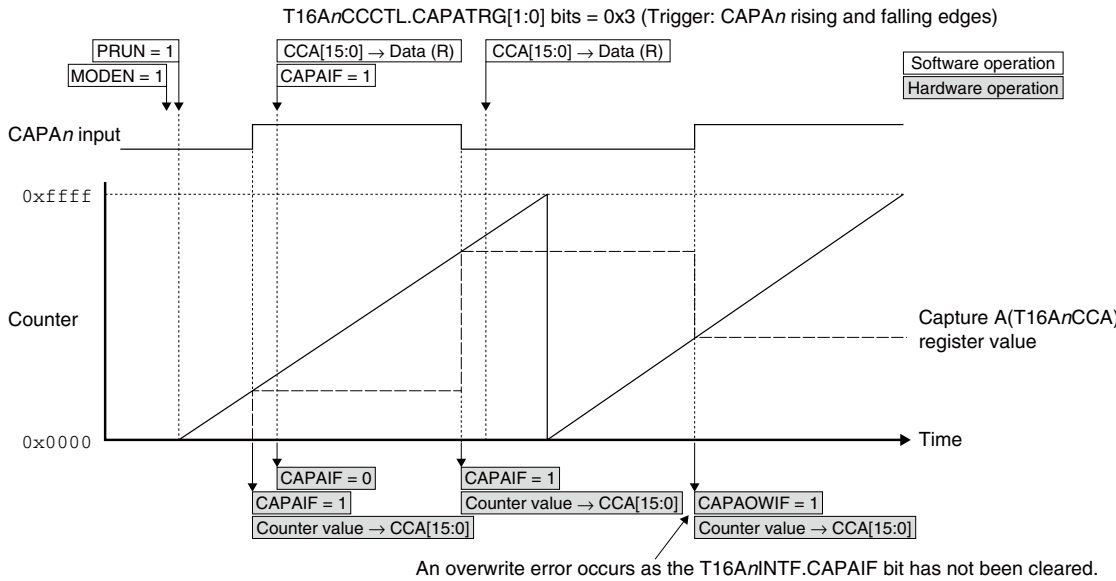


Figure 16.4.3.3 Operations in Capture Mode (Counter = Repeat Mode Only Supported)

Multi-comparator/capture mode and the counter channel used

Normally, a counter block is connected to the comparator/capture block in the same channel and a different clock can be used in each channel.

On the other hand, when T16A3 is placed into multi-comparator/capture mode (T16A0CLK.MULTIMD bit = 1), a counter channel to be connected to each comparator/capture channel can be selected using the T16A*n*CTL.CCABCNT[1:0] bits. This enables a counter block channel to connect to two or more comparator/capture block channels. Note, however, that all channels must use the count clock configured in the counter block Ch.0, and a different count clock cannot be used in each channel.

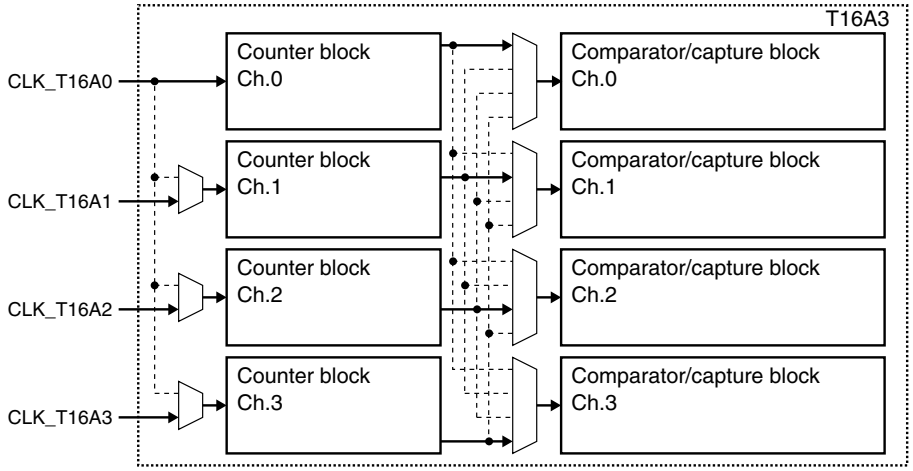


Figure 16.4.3.4 Timer Configuration in Normal Channel Mode (e.g., 4 channels)

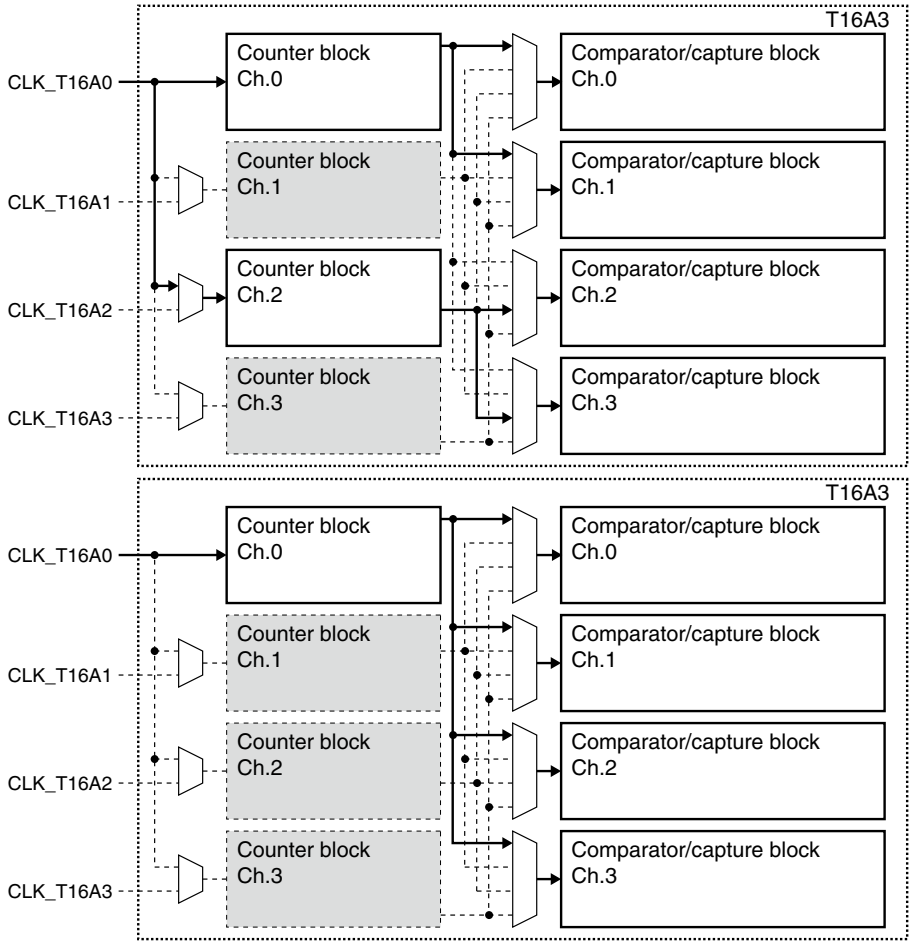


Figure 16.4.3.5 Timer Configuration in Multi-Comparator/Capture Mode (e.g., 4 channels)

16.4.4 TOUT Output Control

The comparator/capture block set in comparator mode can generate TOUT signals using the compare A and compare B signals and can output it to outside the IC. Figure 16.4.4.1 shows the TOUT output circuit (one channel).

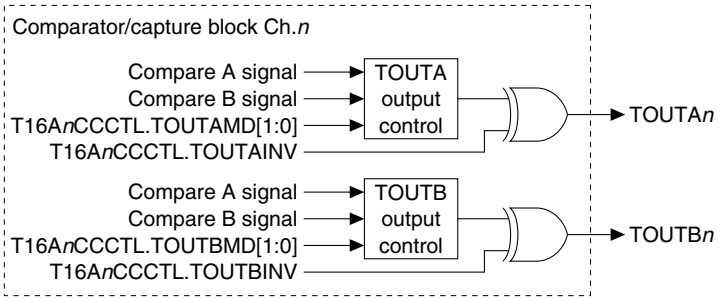


Figure 16.4.4.1 TOUT Output Circuit

Each timer channel includes two TOUT output circuits and their signal generation and output can be controlled individually. Although the output circuit and register names use letters 'A' and 'B' to distinguish two systems, it does not mean that they correspond to compare A and B.

TOUT generation mode

The T16A*n*CCCTL.TOUTAMD[1:0] bits (for system A) or the T16A*n*CCCTL.TOUTBMD[1:0] bits (for system B) are used to set how the TOUT signal waveform is changed by the compare A and compare B signals. These bits are also used to turn the TOUT outputs on and off.

TOUT signal polarity

The TOUT signal polarity can be set using the T16A*n*CCCTL.TOUTAINV bit (for system A) or the T16A*n*CCCTL.TOUTBINV bit (for system B).

Figure 16.4.4.2 shows the TOUT output waveform.

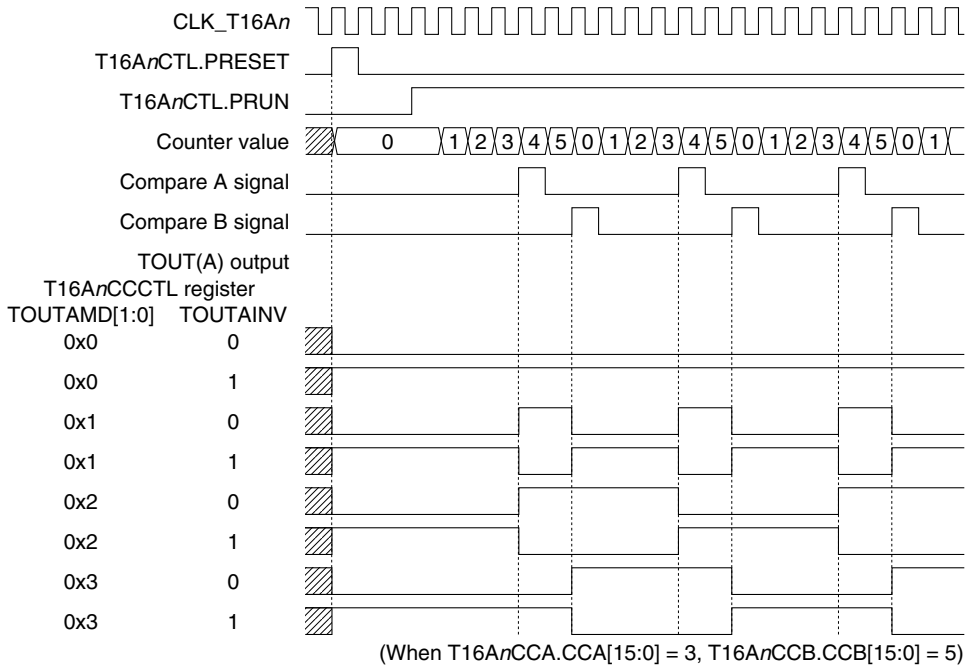
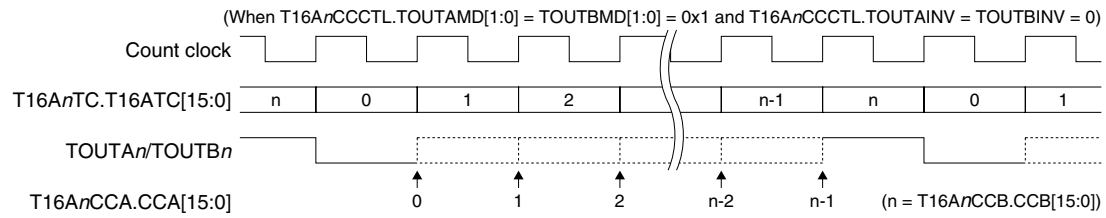


Figure 16.4.4.2 TOUT Output Waveform

PWM waveform output and normal clock/half-clock mode

By setting the T16A*n*CCCTL.TOUTAMD[1:0] and/or TOUTBMD[1:0] bits to 0x1, T16A3 generates a PWM waveform of which the cycle is determined by the compare B signal and the duty ratio is determined by the compare A signal. T16A3 supports half-clock mode to improve the accuracy of the PWM output waveform duty ratio. In half-clock mode (T16A*n*CTL.HCM bit = 1), T16A3 uses the dual-edge counter, which counts at the rising and falling edges of the count clock, to compare with the compare A register. This makes it possible to control the duty ratio with double accuracy as compared to normal clock mode (T16A*n*CTL.HCM bit = 0).

Notes:
- Be sure to avoid placing T16A3 to half-clock mode under a condition shown below.
  (1) When T16A3 is placed into capture mode
  (2) When the T16A*n*CCCTL.TOUTAMD[1:0] or TOUTBMD[1:0] bits are set to 0x2 or 0x3
- The dual-edge counter value cannot be read.
- Do not use the compare A interrupt in half-clock mode.
- In half-clock mode, the T16A*n*CCB register setting value must be less than [T16A*n*CCA setting value / 2 + 0x8000].
- The compare B value (T16A*n*CCB register value) will be compared with the T16A*n*TC counter value even if T16A3 Ch.*n* is set to half-clock mode.



Example: T16A*n*CTL.HCM = 0, T16A*n*CCA.CCA[15:0] = 1, and T16A*n*CCB.CCB[15:0] = 5

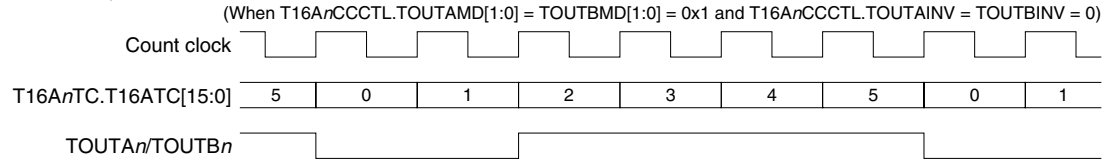
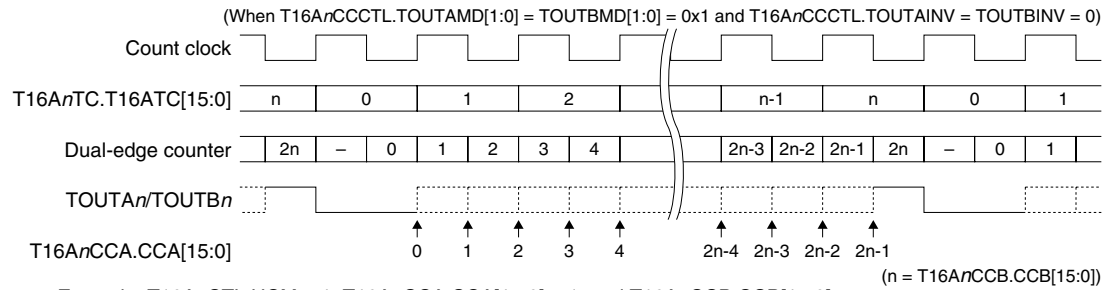


Figure 16.4.4.3 PWM Waveform Output Timings in Normal Clock Mode



Example: T16A*n*CTL.HCM = 1, T16A*n*CCA.CCA[15:0] = 1, and T16A*n*CCB.CCB[15:0] = 5

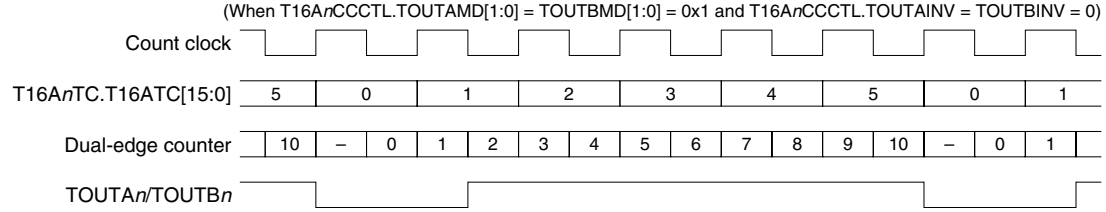


Figure 16.4.4.4 PWM Waveform Output Timings in Half-Clock Mode

16.5 Interrupt

Each T16A3 channel has a function to generate the interrupt shown in Table 16.5.1.

Table 16.5.1 T16A3 Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Capture B overwrite	T16A*n*INTF.CAPBOWIF	When the T16A*n*CCB register is overwritten in capture mode with the T16A*n*INTF.CAPBIF bit set to 1	Writing 1
Capture A overwrite	T16A*n*INTF.CAPAOWIF	When the T16A*n*CCA register is overwritten in capture mode with the T16A*n*INTF.CAPAIF bit set to 1	Writing 1
Capture B	T16A*n*INTF.CAPBIF	When the counter value is loaded to the T16A*n*CCB register by a trigger input to CAPB*n* in capture mode	Writing 1
Capture A	T16A*n*INTF.CAPAIF	When the counter value is loaded to the T16A*n*CCA register by a trigger input to CAPA*n* in capture mode	Writing 1
Compare B	T16A*n*INTF.CMPBIF	When the counter value reaches the T16A*n*CCB register (or compare B buffer) value in comparator mode	Writing 1
Compare A	T16A*n*INTF.CMPAIF	When the counter value reaches the T16A*n*CCA register (or compare A buffer) value in comparator mode	Writing 1

T16A3 provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

16.6 Control Registers

T16A3 Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*CLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–4	CLKDIV[3:0]	0x0	H0	R/W	
	3	–	0	–	R	
	2	MULTIMD	0	H0	R/W	T16A3 Ch.0
		–	0	–	R	T16A3 Ch.1–*n*
	1–0	CLKSRC[1:0]	0x0	H0	R/W	–

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the T16A3 Ch.*n* operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–4 CLKDIV[3:0]

These bits select the division ratio of the T16A3 Ch.*n* operating clock (counter clock).

Bit 3 Reserved

Bit 2 MULTIMD (T16A3 Ch.0)

This bit sets T16A3 to multi-comparator/capture mode.

1 (R/W): Multi-comparator/capture mode

0 (R/W): Normal channel mode

For detailed information, refer to "Comparator/Capture Block Operations, Multi-comparator/capture mode and the counter channel used."

Bit 2 Reserved (T16A3 Ch.1–*n*)

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of T16A3 Ch.*n*.

Table 16.6.1 Clock Source and Division Ratio Settings

T16A*n*CLK.CLKDIV[3:0] bits	T16A*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC/EXCL*m*
0xf	1/32,768	1/1	1/32,768	1/1
0xe	1/16,384		1/16,384	
0xd	1/8,192		1/8,192	
0xc	1/4,096		1/4,096	
0xb	1/2,048		1/2,048	
0xa	1/1,024		1/1,024	
0x9	1/512	F256*	1/512	
0x8	1/256	1/256	1/256	
0x7	1/128	1/128	1/128	
0x6	1/64	1/64	1/64	
0x5	1/32	1/32	1/32	
0x4	1/16	1/16	1/16	
0x3	1/8	1/8	1/8	
0x2	1/4	1/4	1/4	
0x1	1/2	1/2	1/2	
0x0	1/1	1/1	1/1	

* Regulated 256 Hz clock

(Note 1) The oscillator circuits/external input that are not supported in this IC cannot be selected as the clock source.
(Note 2) When the T16A*n*CLK.CLKSRC[1:0] bits are set to 0x3, EXCL*m* is selected for the channel with an event counter function or EXOSC is selected for other channels.

T16A3 Counter Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*CTL	15–9	–	0x00	–	R	–
	8	PRUN	0	H0	R/W	
	7	–	X	–	R	Read value is undefined.
	6	HCM	0	H0	R/W	–
	5–4	CCABCNT[1:0]	0x0	H0	R/W	
	3	CBUFEN	0	H0	R/W	
	2	TRMD	0	H0	R/W	
	1	PRESET	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 PRUN

This bit starts/stops counting.
1 (W): Start counting
0 (W): Stop counting
1 (R): Counting
0 (R): Idle

By writing 1 to this bit, the counter block starts count operations. However, the T16A*n*CTL.MODEN bit must be set to 1 in conjunction with this bit or it must be set in advance. While the timer is running, writing 0 to this bit stops count operations. When the counter stops by the compare B signal in one-shot mode, this bit is automatically cleared to 0.

Bit 7 Reserved

Bit 6 HCM

This bit sets T16A3 to half-clock mode.
1 (R/W): Enable (half-clock mode)
0 (R/W): Disable (normal clock mode)

For detailed information, refer to "TOUT Output Control, PWM waveform output and normal clock/half-clock mode."

Bits 5–4 CCABCNT[1:0]

These bits select a counter to be connected to the comparator/capture block of each channel in multi-comparator/capture mode (T16A0CLK.MULTIMD bit = 1).

Table 16.6.2 Counter Selection

T16A*n*CTL.CCABCNT[1:0] bits	Counter channel
0x3	Counter block Ch.3
0x2	Counter block Ch.2
0x1	Counter block Ch.1
0x0	Counter block Ch.0

When using T16A3 in normal channel mode (T16A0CLK.MULTIMD bit = 0), be sure to connect the counter of the same channel to each comparator/capture block.

Bit 3 CBUFEN

This bit enables or disables the compare buffers.

1 (R/W): Enable

0 (R/W): Disable

For detailed information, refer to "Comparator/Capture Block Operations, Compare buffers."

Note: Make sure the counter is halted (T16A*n*CTL.PRUN bit = 0) before setting the T16A*n*CTL. CBUFEN bit.

Bit 2 TRMD

This bit selects the count mode.

1 (R/W): One-shot mode

0 (R/W): Repeat mode

For detailed information, refer to "Counter Block Operations, Count mode (repeat mode and one-shot mode)."

Bit 1 PRESET

This bit resets the counter.

1 (W): Reset

0 (W): Ineffective

1 (R): Resetting in progress

0 (R): Resetting finished or normal operation

Writing 1 to this bit clears the counter to 0.

Bit 0 MODEN

This bit enables the T16A3 Ch.*n* operations.

1 (R/W): Enable (Start supplying operating clock)

0 (R/W): Disable (Stop supplying operating clock)

T16A3 Counter Ch.*n* Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*TC	15–0	T16ATC[15:0]	0x0000	H0	R	–

Bits 15–0 T16ATC[15:0]

The current counter value can be read out through these bits.

T16A3 Comparator/Capture Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*CCCTL	15–14	CAPBTRG[1:0]	0x0	H0	R/W	–
	13–12	TOUTBMD[1:0]	0x0	H0	R/W	
	11–10	–	0x0	–	R	
	9	TOUTBINV	0	H0	R/W	
	8	CCBMD	0	H0	R/W	
	7–6	CAPATRG[1:0]	0x0	H0	R/W	
	5–4	TOUTAMD[1:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1	TOUTAINV	0	H0	R/W	
	0	CCAMD	0	H0	R/W	

Bits 15–14 CAPBTRG[1:0]

These bits select the trigger edge(s) of the external signal (CAPB*n*) at which the counter value is captured in the T16A*n*CCB register.

Table 16.6.3 Capture B Trigger Edge Selection

T16A*n*CCCTL.CAPBTRG[1:0] bits	Trigger edge
0x3	Falling edge and rising edge
0x2	Falling edge
0x1	Rising edge
0x0	Not triggered

The T16A*n*CCCTL.CAPBTRG[1:0] bits are control bits for capture mode and are ineffective in comparator mode.

Bits 13–12 TOUTBMD[1:0]

These bits configure how the TOUTB signal waveform (TOUTB*n* output) is changed by the compare A and compare B signals. These bits are also used to turn the TOUTB output on and off.

Table 16.6.4 TOUTB Signal Generation Mode

T16A*n*CCCTL. TOUTBMD[1:0] bits	When compare A occurs	When compare B occurs
0x3	No change	Toggle
0x2	Toggle	No change
0x1	Rise	Fall
0x0	Disable output	

The T16A*n*CCCTL.TOUTBMD[1:0] bits are control bits for comparator mode and are ineffective in capture mode.

Bits 11–10 Reserved

Bit 9 TOUTBINV

This bit selects the TOUTB signal (TOUTB*n* output) polarity.
1 (R/W): Inverted (active low)
0 (R/W): Normal (active high)

The T16A*n*CCCTL.TOUTBINV bit is a control bit for comparator mode and are ineffective in capture mode.

Bit 8 CCBMD

This bit selects the T16A*n*CCB register operating mode.
1 (R/W): Capture mode (T16A*n*CCB register = capture B register)
0 (R/W): Comparator mode (T16A*n*CCB register = compare B register)

Bits 7–6 CAPATRG[1:0]

These bits select the trigger edge(s) of the external signal (CAPA*n*) at which the counter value is captured in the T16A*n*CCA register.

Table 16.6.5 Capture A Trigger Edge Selection

T16A*n*CCCTL.CAPATRG[1:0] bits	Trigger edge
0x3	Falling edge and rising edge
0x2	Falling edge
0x1	Rising edge
0x0	Not triggered

The T16A*n*CCCTL.CAPATRG[1:0] bits are control bits for capture mode and are ineffective in comparator mode.

Bits 5–4 TOUTAMD[1:0]

These bits configure how the TOUTA signal waveform (TOUTA*n* output) is changed by the compare A and compare B signals. These bits are also used to turn the TOUTA output on and off.

Table 16.6.6 TOUTA Signal Generation Mode

T16A*n*CCCTL. TOUTAMD[1:0] bits	When compare A occurs	When compare B occurs
0x3	No change	Toggle
0x2	Toggle	No change
0x1	Rise	Fall
0x0	Disable output	

The T16A*n*CCCTL.TOUTAMD[1:0] bits are control bits for comparator mode and are ineffective in capture mode.

Bits 3–2 Reserved

Bit 1 TOUTAINV

This bit selects the TOUTA signal (TOUTA*n* output) polarity.
1 (R/W): Inverted (active low)
0 (R/W): Normal (active high)

The T16A*n*CCCTL.TOUTAINV bit is a control bit for comparator mode and are ineffective in capture mode.

Bit 0 CCAMD

This bit selects the T16A*n*CCA register operating mode.
1 (R/W): Capture mode (T16A*n*CCA register = capture A register)
0 (R/W): Comparator mode (T16A*n*CCA register = compare A register)

T16A3 Comparator/Capture Ch.*n* A Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*CCA	15–0	CCA[15:0]	0x0000	H0	R/W	–

Bits 15–0 CCA[15:0]

In comparator mode (T16A*n*CCCTL.CCAMD bit = 0), this register is configured as the compare A register and used to set compare A data that is compared with the counter value.
In capture mode (T16A*n*CCCTL.CCAMD bit = 1), this register is configured as the capture A register and the counter value captured by the external trigger signal (CAPA*n*) is loaded.

Note: When writing data to the T16A*n*CCA register successively, data should be written at intervals of one or more T16A3 count clock cycles.

T16A3 Comparator/Capture Ch.*n* B Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*CCB	15–0	CCB[15:0]	0x0000	H0	R/W	–

Bits 15–0 CCB[15:0]

In comparator mode (T16A*n*CCCTL.CCBMD bit = 0), this register is configured as the compare B register and used to set compare B data that is compared with the counter value.

In capture mode (T16A*n*CCCTL.CCBMD bit = 1), this register is configured as the capture B register and the counter value captured by the external trigger signal (CAPB*n*) is loaded.

Note: When writing data to the T16A*n*CCB register successively, data should be written at intervals of one or more T16A3 count clock cycles.

T16A3 Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*INTF	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5	CAPBOWIF	0	H0	R/W	Cleared by writing 1.
	4	CAPAOWIF	0	H0	R/W	
	3	CAPBIF	0	H0	R/W	
	2	CAPAIF	0	H0	R/W	
	1	CMPBIF	0	H0	R/W	
	0	CMPAIF	0	H0	R/W	

Bits 15–6 Reserved

Bit 5 CAPBOWIF
Bit 4 CAPAOWIF
Bit 3 CAPBIF
Bit 2 CAPAIF
Bit 1 CMPBIF
Bit 0 CMPAIF

These bits indicate the T16A3 Ch.*n* interrupt cause occurrence status.

1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:

T16A*n*INTF.CAPBOWIF bit: Capture B overwrite interrupt
T16A*n*INTF.CAPAOWIF bit: Capture A overwrite interrupt
T16A*n*INTF.CAPBIF bit: Capture B interrupt
T16A*n*INTF.CAPAIF bit: Capture A interrupt
T16A*n*INTF.CMPBIF bit: Compare B interrupt
T16A*n*INTF.CMPAIF bit: Compare A interrupt

T16A3 Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16A*n*INTE	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5	CAPBOWIE	0	H0	R/W	
	4	CAPAOWIE	0	H0	R/W	
	3	CAPBIE	0	H0	R/W	
	2	CAPAIE	0	H0	R/W	
	1	CMPBIE	0	H0	R/W	
	0	CMPAIE	0	H0	R/W	

Bits 15–6 Reserved

Bit 5 CAPBOWIE
Bit 4 CAPAOWIE
Bit 3 CAPBIE
Bit 2 CAPAIE
Bit 1 CMPBIE
Bit 0 CMPAIE

These bits enable T16A3 Ch.*n* interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
T16A*n*INTE.CAPBOWIE bit: Capture B overwrite interrupt
T16A*n*INTE.CAPAOWIE bit: Capture A overwrite interrupt
T16A*n*INTE.CAPBIE bit: Capture B interrupt
T16A*n*INTE.CAPAIE bit: Capture A interrupt
T16A*n*INTE.CMPBIE bit: Compare B interrupt
T16A*n*INTE.CMPAIE bit: Compare A interrupt

Note: To prevent generating unnecessary interrupts, clear the corresponding interrupt flag before enabling interrupts.

17 Parallel Interface (PIO)

17.1 Overview

PIO is an 8-bit parallel interface that allows connection of an SRAM type device. The features of PIO are listed below.

- 8-bit data bus
- 8-bit address bus
- Bus control output signals: read, write, and chip enable
- Provides GPIO mode to use the address bus pins as general-purpose output ports and the data bus pins as general-purpose input ports.
- Provides PIO pins with pull-up control.

Figure 17.1.1 shows the PIO configuration.

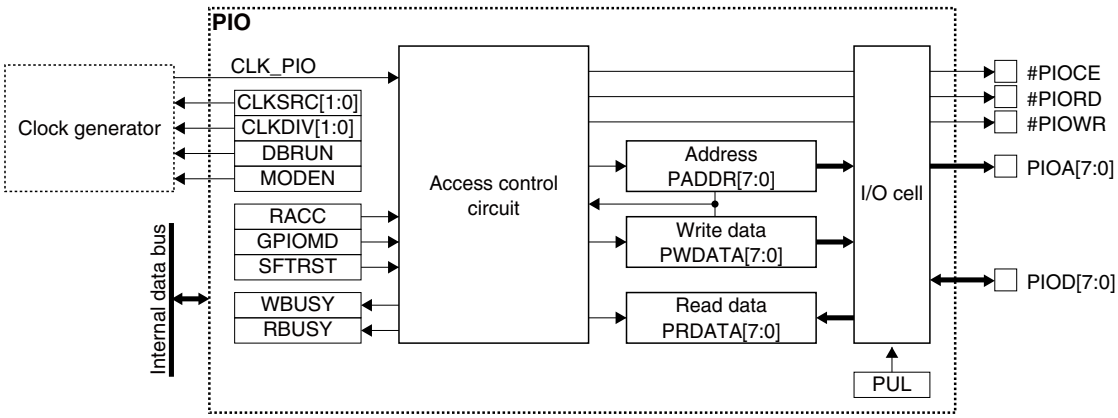


Figure 17.1.1 PIO Configuration

17.2 Input/Output Pins and External Connections

17.2.1 List of Input/Output Pins

Table 17.2.1.1 lists the PIO pins.

Table 17.2.1.1 List of PIO Pins

Pin name	I/O*	Initial status*	Function
PIOA[7:0]	O	O (Low)	PIO address output
PIOD[7:0]	I/O	I (Hi-Z)	PIO data input/output
#PIOCE	O	O (High)	PIO chip enable signal output
#PIORD	O	O (High)	PIO read signal output
#PIOWR	O	O (High)	PIO write signal output

* Indicates the status when the pin is configured for PIO.

If the port is shared with the PIO pin and other functions, the PIO input/output function must be assigned to the port before activating PIO. For more information, refer to the "I/O Ports" chapter.

17.2.2 External Connections

Figure 17.2.2.1 shows a connection diagram between PIO in this IC and an external (SRAM type) device.

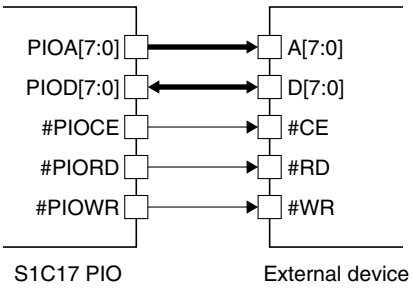


Figure 17.2.2.1 Connections between PIO and an External Device

17.2.3 Pin Pull-Up Function

PIO includes a pull-up resistor for the PIOD[7:0] pins. Setting the PIOMOD.PUL bit to 1 enables the resistor to pull up the PIOD[7:0] pins.

17.3 Clock Settings

17.3.1 PIO Operating Clock

When using PIO, the PIO operating clock CLK_PIO must be supplied to PIO from the clock generator. The CLK_PIO supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Set the following PIOCLK register bits:
   - PIOCLK.CLKSRC[1:0] bits (Clock source selection)
   - PIOCLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)

17.3.2 Clock Supply in SLEEP Mode

When using PIO during SLEEP mode, the PIO operating clock CLK_PIO must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_PIO clock source.

17.3.3 Clock Supply in DEBUG Mode

The CLK_PIO supply during DEBUG mode should be controlled using the PIOCLK.DBRUN bit.
The CLK_PIO supply to PIO is suspended when the CPU enters DEBUG mode if the PIOCLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_PIO supply resumes. Although PIO stops operating when the CLK_PIO supply is suspended, the input/output pins and registers retain the status before DEBUG mode was entered. If the PIOCLK.DBRUN bit = 1, the CLK_PIO supply is not suspended and PIO will keep operating in DEBUG mode.

17.4 Operations

17.4.1 Initialization

PIO should be initialized with the procedure shown below.

1. Assign the PIO input/output function to the ports. (Refer to the "I/O Ports" chapter.)
2. Set the PIOCLK.CLKSRC[1:0] and PIOCLK.CLKDIV[1:0] bits. (Configure operating clock)

3. Configure the following PIOMOD register bits:
   - PIOMOD.PUL bit (Enable/disable pin pull-up)
   - PIOMOD.GPIOMD bit (Select SRAM mode or GPIO mode)
4. Set the following PIOCTL register bits:
   - Set the PIOCTL.SFTRST bit to 1. (Execute software reset)
   - Set the PIOCTL.MODEN bit to 1. (Enable PIO operations)

17.4.2 Operations in SRAM Mode

The PIO, which is placed into SRAM mode (PIOMOD.GPIOMD bit = 0), functions as an SRAM type parallel interface.

Data write procedure

1. Write an address to be accessed and output data to the PIOWRDAT register.
   - PIOWRDAT.PADDR[7:0] bits (Address)
   - PIOWRDAT.PWDATA[7:0] bits (Output data)
2. When outputting data successively, wait until the PIOSTAT.WBUSY bit goes to 0.
3. Repeat Steps 1 and 2 until the end of data output.

Data write operations

Writing an address/data to the PIOWRDAT register generates a trigger signal and it is sampled within one CLK_PIO clock cycle. This starts a data write cycle.
After one clock cycle from the trigger sampling, PIO asserts the #PIOCE signal and outputs the address and data from the PIOA[7:0] pins and the PIOD[7:0] pins, respectively. The PIOSTAT.WBUSY bit is also set to 1 (write cycle busy status).
PIO asserts the #PIOWR signal while the access cycle (one CLK_PIO clock cycle) after the setup cycle (one CLK_PIO clock cycle).
PIO negates the #PIOCE signal and switches the PIOD[7:0] pin direction to input after the hold cycle (one CLK_PIO clock cycle).
Finally, PIO clears the PIOSTAT.WBUSY bit to 0 (write cycle completed/idle status).

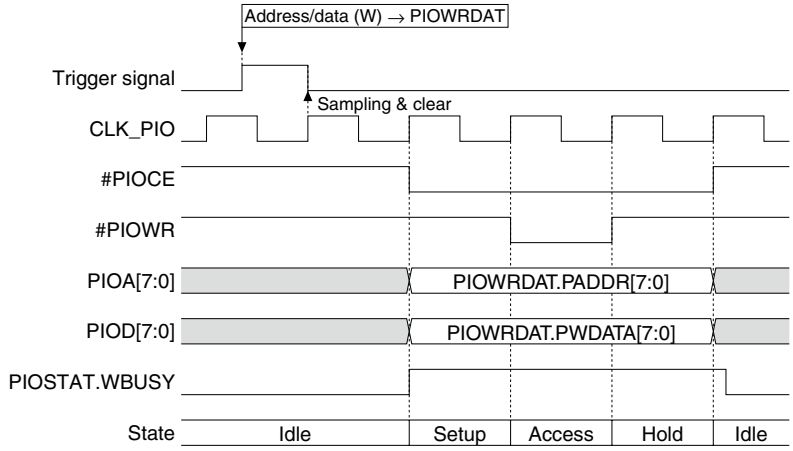


Figure 17.4.2.1 Data Write Timing

Data read procedure

1. Write an address to be accessed to the PIOWRDAT.PADDR[7:0] bits.
2. Write 1 to the PIOCTL.RACC bit. (Data read trigger)
3. Wait until the PIOSTAT.RBUSY bit goes to 0.
4. Read the input data from the PIORDDAT.PRDATA[7:0] bits.
5. Repeat Steps 1 to 4 until the end of data input.

Data read operations

Writing 1 to the PIOCTL.RACC bit generates a trigger signal and it is sampled within one CLK_PIO clock cycle. This starts a data read cycle.

After one clock cycle from the trigger sampling, PIO asserts the #PIOCE signal and outputs the address from the PIOA[7:0] pins. The PIOSTAT.RBUSY bit is also set to 1 (read cycle busy status).

PIO asserts the #PIORD signal while the access cycle (one CLK_PIO clock cycle) to input data from the PIOD[7:0] pins to after the setup cycle (one CLK_PIO clock cycle). The input data is loaded to the PIORDDAT.PRDATA[7:0] bits.

PIO negates the #PIOCE signal after the hold cycle (one CLK_PIO clock cycle).

Finally, PIO clears the PIOSTAT.RBUSY bit to 0 (read cycle completed/idle status).

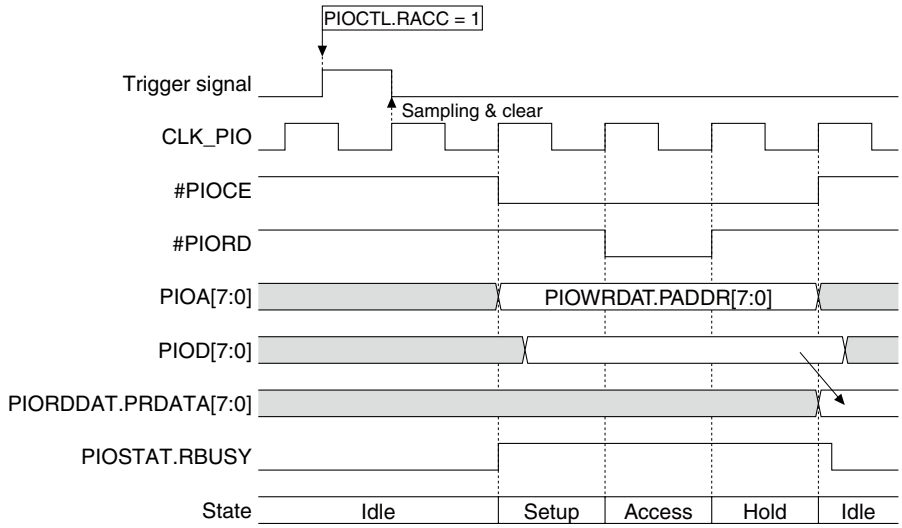


Figure 17.4.2.2 Data Read Timing

17.4.3 Operations in GPIO Mode

In GPIO mode (PIOMOD.GPIOMD bit = 1), the address bus pins and data bus pins are configured as eight bits of general-purpose output ports and eight bits of general-purpose input ports, respectively. The bus control signals are always fixed at inactive state and the PIOWRDAT.PWDATA[7:0] bits become ineffective.

Port output

Write data to be output from the PIOA[7:0] pins to the PIOWRDAT.PADDR[7:0] bits.

The PIOA[7:0] pin outputs change immediately after writing.

Port input

The PIOD[7:0] pin status can be read through the PIORDDAT.PRDATA[7:0] bits.

The PIOD[7:0] pin status is sampled in the CLK_PIO clock cycles and loaded to the PIORDDAT.PRDATA[7:0] bits. Therefore, maximum one CLK_PIO cycle of delay occurs until the input transition is reflected to the PIORDDAT.PRDATA[7:0] bits.

17.5 Control Registers

PIO Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PIOCLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the PIO operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits select the division ratio of the PIO operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of PIO.

Table 17.5.1 Clock Source and Division Ratio Settings

PIOCLK. CLKDIV[1:0] bits	PIOCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x3	1/8	1/8	1/8	1/1
0x2	1/4	1/4	1/4	
0x1	1/2	1/2	1/2	
0x0	1/1	1/1	1/1	

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The PIOCLK register settings can be altered only when the PIOCTL.MODEN bit = 0.

PIO Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PIOMOD	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	PUL	0	H0	R/W	
	0	GPIOMD	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 PUL

This bit enables pull-up of the PIO pins.

1 (R/W): Enable pull-up

0 (R/W): Disable pull-up

Bit 0 GPIOMD

This bit sets PIO to GPIO mode.

1 (R/W): GPIO mode

0 (R/W): SRAM mode

PIO Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PIOCTL	15–9	–	0x00	–	R	–
	8	RACC	0	H0	W	Always read as 0.
	7–2	–	0x00	–	R	–
	1	SFTRST	0	H0	W	Always read as 0.
	0	MODEN	0	H0	R/W	–

Bits 15–9 Reserved

Bit 8 RACC

This bit starts a data read cycle.

1 (W): Data read trigger
0 (W): Ineffective
0 (R): Always 0 when being read

Bits 7–2 Reserved

Bit 1 SFTRST

This bit issues software reset to PIO.

1 (W): Issue software reset
0 (W): Ineffective
0 (R): Always 0 when being read

Setting this bit resets the PIO access control circuit. If 1 is written while a bus cycle is being executed, the bus cycle is terminated at that point.

Bit 0 MODEN

This bit enables the PIO operations.

1 (R/W): Enable PIO operations (The operating clock is supplied.)
0 (R/W): Disable PIO operations (The operating clock is stopped.)

Note: If the PIOCTL.MODEN bit is altered from 1 to 0 while a bus cycle is being executed, the input/output data cannot be guaranteed. When setting the PIOCTL.MODEN bit to 1 again after that, be sure to write 1 to the PIOCTL.SFTRST bit as well.

PIO Address/Write Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PIOWRDAT	15–8	PADDR[7:0]	0x00	H0	R/W	–
	7–0	PWDATA[7:0]	0x00	H0	R/W	

Bits 15–8 PADDR[7:0]

These bits are used to set an address to be accessed.

Bits 7–0 PWDATA[7:0]

These bits are used to set output data.

Writing to the PIOWRDAT register starts a data write cycle.

Notes: • If writing 1 to the PIOCTL.RACC bit and writing to the PIOWRDAT register are performed at a time (read and write triggers are issued at the same time), PIO executes the read cycle.

• Do not write an address/data to the PIOWRDAT register during write cycle (when the PIOSTAT.WBUSY bit = 1) and read cycle (when the PIOSTAT.RBUSY bit = 1).

PIO Read Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PIORDDAT	15–8	–	0x00	–	R	–
	7–0	PRDATA[7:0]	0x00	H0	R	

Bits 15–8 Reserved

Bits 7–0 PRDATA[7:0]

Input data can be read through these bits. Before reading this register, issue a read trigger (PIOCTL. RACC bit = 1) and wait until the read cycle has finished (PIOSTAT.RBUSY bit changes 1 to 0).

PIO Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PIOSTAT	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	WBUSY	0	H0/S0	R	
	0	RBUSY	0	H0/S0	R	

Bits 15–2 Reserved

Bit 1 WBUSY

This bit indicates the write cycle operating status.

1 (R): Write cycle is being executed.

0 (R): Idle

Bit 0 RBUSY

This bit indicates the read cycle operating status.

1 (R): Read cycle is being executed.

0 (R): Idle

18 EPD Timing Controller (EPD Tcon)

18.1 Overview

EPD Tcon is an EPD timing controller that controls an active matrix EPD panel. The features of EPD Tcon are listed below.

- Supports active matrix EPD panels.
- Controls the EPD panel signals via the parallel interface.
- Reads display data stored in the display RAM (RAM2) and sends it to the EPD panel via SPI Ch.1. (The display RAM can be accessed from both S1C17 and EPD Tcon.)

Figure 18.1.1 shows the EPD control system configuration.

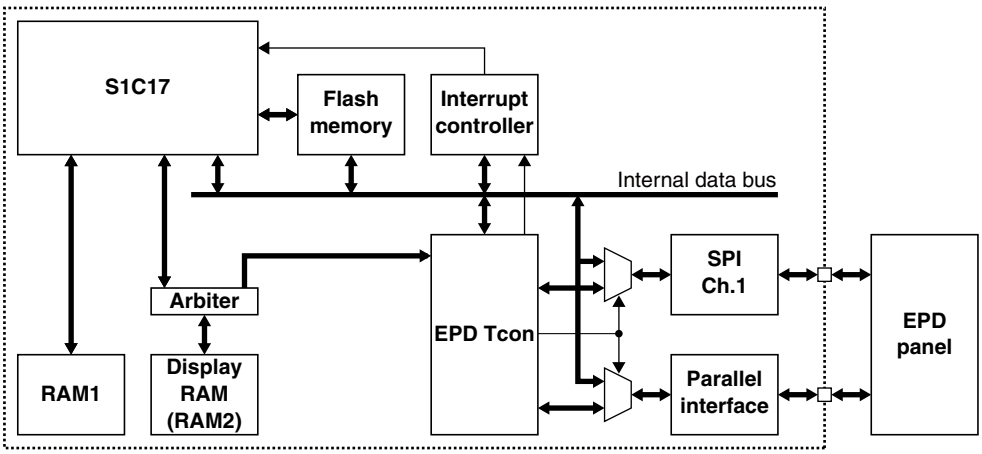


Figure 18.1.1 EPD Control System Configuration

A dedicated API library is provided for EPD Tcon and all the EPD Tcon functions can be controlled using the API. For the data format in the display RAM, various settings, and operations, refer to the descriptions of the EPD Tcon API library (EPD Timing Controller S1C17F13 Manual (separately attached sheet)).

Note: EPD Tcon occupies SPI Ch.1 or the parallel interface while it is running. This peripheral circuit cannot be accessed from the S1C17.

18.2 Interrupt

EPD Tcon has a function to generate the interrupt shown in Table 18.2.1.

Table 18.2.1 EPD Tcon Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Display refresh completion	EPDINTF.ENDIF	When an EPD panel display update processing has completed	Writing 1

The EPD Tcon provides an interrupt enable bit corresponding to the interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the “Interrupt Controller” chapter.

18.3 Control Registers

EPD Tcon Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
EPDCTL	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	MODEN	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 MODEN

This bit enables the EPD Tcon operations.

1 (R/W): Enable EPD Tcon operations (The operating clock is supplied.)

0 (R/W): Disable EPD Tcon operations (The operating clock is stopped.)

EPD Tcon Interrupt Flag and Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
EPDINTF	15–9	–	0x00	–	R	–
	8	BUSY	1	H0	R	
	7–1	–	0x00	–	R	
	0	ENDIF	0	H0	R/W	Cleared by writing 1.

Bits 15–9 Reserved

Bit 8 BUSY

This bit indicates the EPD Tcon operating status.

1 (R): Operating (During display data transfer)

0 (R): Idle

Bits 7–1 Reserved

Bit 0 ENDIF

This bit indicates the EPD Tcon interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag

0 (W): Ineffective

EPD Tcon Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
EPDINTE	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	ENDIE	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 ENDIE

This bit enables EPD Tcon interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

19 R/F Converter (RFC)

19.1 Overview

The RFC is a CR oscillation type A/D converter (R/F converter).
The features of the RFC are listed below.

- Converts the sensor resistance into a digital value by performing CR oscillation and counting the oscillation clock.
- Achieves high-precision measurement system with low errors by oscillating the reference resistor and the sensor in the same conditions to obtain the difference between them.
- Includes a 24-bit measurement counter to count the oscillation clocks.
- Includes a 24-bit time base counter to count the internal clock for equalizing the measurement time between the reference resistor and the sensor.
- Supports DC bias resistive sensors and AC bias resistive sensors.
  (A thermometer/hygrometer can be easily implemented by connecting a thermistor or a humidity sensor and a few passive elements (resistor and capacitor).)
- Allows measurement (counting) by inputting external clocks.
- Provides an output and continuous oscillation function for monitoring the oscillation frequency.
- Can generate reference oscillation completion, sensor (A and B) oscillation completion, measurement counter overflow error, and time base counter overflow error interrupts.

Figure 19.1.1 shows the RFC configuration.

Channel configuration in this IC
- 2 channels (Ch.0 and Ch.1)

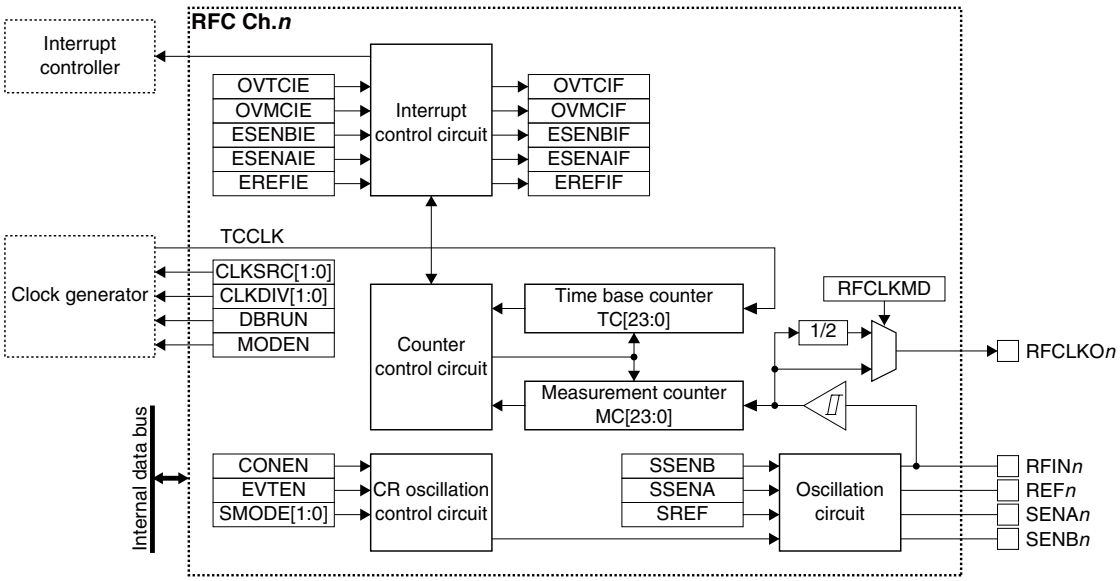


Figure 19.1.1 RFC Configuration

19.2 Input/Output Pins and External Connections

19.2.1 List of Input/Output Pins

Table 19.2.1.1 lists the RFC pins.

Table 19.2.1.1 List of RFC Pins

Pin name	I/O*	Initial status*	Function
SENB*n*	A	Hi-Z	Sensor B oscillation control pin
SENA*n*	A	Hi-Z	Sensor A oscillation control pin
REF*n*	A	Hi-Z	Reference oscillation control pin
RFIN*n*	A	VSS	RFCLK input or oscillation control pin
RFCLKO*n*	O	Hi-Z	RFCLK monitoring output pin RFCLK is output to monitor the oscillation frequency.

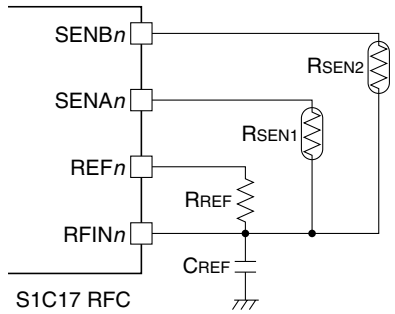
* Indicates the status when the pin is configured for the RFC.

If the port is shared with the RFC pin and other functions, the RFC input/output function must be assigned to the port before activating the RFC. For more information, refer to the "I/O Ports" chapter.

Note: The RFIN*n* pin goes to VSS level when the port is switched. Be aware that large current may flow if the pin is biased by an external circuit.

19.2.2 External Connections

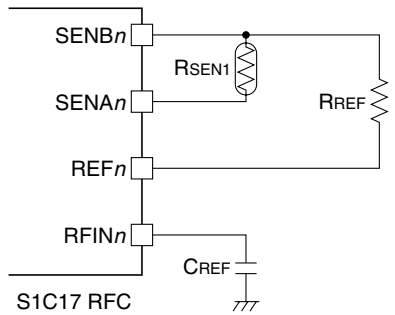
The figures below show connection examples between the RFC and external sensors. For the oscillation mode and external clock input mode, refer to "Operating Mode."



RREF: Reference resistor
RSEN1: Resistive sensor (DC bias)
RSEN2: Resistive sensor (DC bias)
CREF: Oscillation capacitor

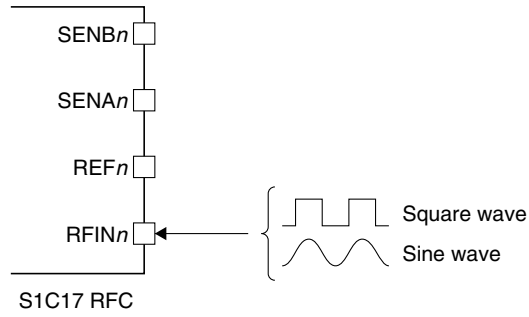
* Leave the unused pin (SENA*n* or SENB*n*) open if one resistive sensor only is used.

Figure 19.2.2.1 Connection Example in Resistive Sensor DC Oscillation Mode



RREF: Reference resistor
RSEN1: Resistive sensor (AC bias)
CREF: Oscillation capacitor

Figure 19.2.2.2 Connection Example in Resistive Sensor AC Oscillation Mode



∗ Leave the unused pins open.

Figure 19.2.2.3 External Clock Input in External Clock Input Mode

19.3 Clock Settings

19.3.1 RFC Operating Clock

When using the RFC, the RFC operating clock TCCLK must be supplied to the RFC from the clock generator. The TCCLK supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Set the following RFC*n*CLK register bits:
   - RFC*n*CLK.CLKSRC[1:0] bits (Clock source selection)
   - RFC*n*CLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)

The time base counter performs counting with TCCLK set here. Selecting a higher clock results in higher conversion accuracy, note, however, that the frequency should be determined so that the time base counter will not overflow during reference oscillation.

19.3.2 Clock Supply in SLEEP Mode

When using RFC during SLEEP mode, the RFC operating clock TCCLK must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the TCCLK clock source.

19.3.3 Clock Supply in DEBUG Mode

The TCCLK supply during DEBUG mode should be controlled using the RFC*n*CLK.DBRUN bit.
The TCCLK supply to the RFC is suspended when the CPU enters DEBUG mode if the RFC*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the TCCLK supply resumes. Although the RFC stops operating when the TCCLK supply is suspended, the output pin and registers retain the status before DEBUG mode was entered. If the RFC*n*CLK.DBRUN bit = 1, the TCCLK supply is not suspended and the RFC will keep operating in DEBUG mode.

19.4 Operations

19.4.1 Initialization

The RFC should be initialized with the procedure shown below.

1. Configure the RFC*n*CLK.CLKSRC[1:0] and RFC*n*CLK.CLKDIV[1:0] bits. (Configure operating clock)
2. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the RFC*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the RFC*n*INTE register to 1. (Enable interrupts)
3. Assign the RFC input/output function to the ports. (Refer to the "I/O Ports" chapter.)

4. Configure the following RFC*n*CTL register bits:
   - RFC*n*CTL.EVTEN bit (Enable/disable external clock input mode)
   - RFC*n*CTL.SMODE[1:0] bits (Select oscillation mode)
   - Set the RFC*n*CTL.MODEN bit to 1. (Enable RFC operations)

19.4.2 Operating Modes

The RFC has two oscillation modes that use the RFC internal oscillation circuit and an external clock input mode for measurements using an external input clock. The channels may be configured to a different mode from others.

Oscillation mode

The oscillation mode is selected using the RFC*n*CTL.SMODE[1:0] bits.

DC oscillation mode for resistive sensor measurements

This mode performs measurements by DC driving the reference resistor and the resistive sensor to oscillate. Set the RFC into this mode when a DC bias resistive sensor is connected. This mode allows connection of two resistive sensors to a channel.

AC oscillation mode for resistive sensor measurements

This mode performs measurements by AC driving the reference resistor and the resistive sensor to oscillate. Set the RFC into this mode when an AC bias resistive sensor is connected. One resistive sensor only can be connected to a channel.

External clock input mode (event counter mode)

This mode enables input of external clock/pulses to perform counting similar to the internal oscillation clock. A sine wave may be input as well as a square wave (for the threshold value of the Schmitt input, refer to "R/F Converter Characteristics, High level Schmitt input threshold voltage V_{T+} and Low level Schmitt input threshold voltage V_{T-}" in the "Electrical Characteristics" chapter). This function is enabled by setting the RFC*n*CTL.EVTEN bit to 1. The measurement procedure is the same as when the internal oscillation circuit is used.

19.4.3 RFC Counters

The RFC incorporates two counters shown below.

Measurement counter (MC)

The measurement counter is a 24-bit presettable up counter. Counting the reference oscillation clock and the sensor oscillation clock for the same duration of time using this counter minimizes errors caused by voltage, and unevenness of IC quality, as well as external parts and on-board parasitic elements. The counter values should be corrected via software after the reference and sensor oscillations are completed according to the sensor characteristics to determine the value being currently detected by the sensor.

Time base counter (TC)

The time base counter is a 24-bit presettable up/down counter. The time base counter counts up with TCCLK during reference oscillation to measure the reference oscillation time. During sensor oscillation, it counts down from the reference oscillation time and stops the sensor oscillation when it reaches 0x000000. This means that the sensor oscillation time becomes equal to the reference oscillation time. The value counted during reference oscillation should be saved in the memory. It can be reused at subsequent sensor oscillations omitting reference oscillations.

Counter initial value

To obtain the difference between the reference oscillation and sensor oscillation clock count values from the measurement counter simply, appropriate initial values must be set to the measurement counter before starting reference oscillation.

Connecting the reference element and sensor with the same resistance will result in <Initial value: n> = <Counter value at the end of sensor oscillation: m> (if error = 0). Setting a large <Initial value: n> increases the resolution of measurement. However, the measurement counter may overflow during sensor oscillation when the sensor value decreases below the reference element value (the measurement will be canceled). The initial value for the measurement counter should be determined taking the range of sensor value into consideration.
The time base counter should be set to 0x000000 before starting reference oscillation.

Counter value read

The measurement and time base counters operate on RFCCLK and TCCLK, respectively. Therefore, to read correctly by the CPU while the counter is running, read the counter value twice or more and check to see if the same value is read.

19.4.4 Converting Operations and Control Procedure

An R/F conversion procedure and the RFC operations are shown below. Although the following descriptions assume that the internal oscillation circuit is used, external clock input mode can be controlled with the same procedure.

R/F control procedure

1. Set the initial value (0x000000 - n) to the RFC*n*MCH and RFC*n*MCL registers (measurement counter).
2. Clear the RFC*n*TCH and RFC*n*TCL registers (time base counter) to 0x000000.
3. Clear both the RFC*n*INTF.EREFIF and RFC*n*INTF.OVTCIF bits by writing 1.
4. Set the RFC*n*TRG.SREF bit to 1 to start reference oscillation.
5. Wait for an RFC interrupt.
   i. If the RFC*n*INTF.EREFIF bit = 1 (reference oscillation completion), clear the RFC*n*INTF.EREFIF bit and then go to Step 6.
   ii. If the RFC*n*INTF.OVTCIF bit = 1 (time base counter overflow error), clear the RFC*n*INTF.OVTCIF bit and terminate measurement as an error or retry after altering the measurement counter initial value.
6. Clear the RFC*n*INTF.ESENAIF, RFC*n*INTF.ESENBIF, and RFC*n*INTF.OVMCIF bits by writing 1.
7. Set the RFC*n*TRG.SSENA bit (sensor A) or the RFC*n*TRG.SSENB bit (sensor B) corresponding to the sensor to be measured to 1 to start sensor oscillation (use the RFC*n*TRG.SSENA bit in AC oscillation mode).
8. Wait for an RFC interrupt.
   i. If the RFC*n*INTF.ESENAIF bit = 1 (sensor A oscillation completion) or the RFC*n*INTF.ESENBIF bit = 1 (sensor B oscillation completion), clear the RFC*n*INTF.ESENAIF or RFC*n*INTF.ESENBIF bit and then go to Step 9.
   ii. If the RFC*n*INTF.OVMCIF bit = 1 (measurement counter overflow error), clear the RFC*n*INTF.OVMCIF bit and terminate measurement as an error or retry after altering the measurement counter initial value.
9. Read the RFC*n*MCH and RFC*n*MCL registers (measurement counter) and correct the results depending on the sensor to obtain the detected value.

R/F Converting Operations

Reference oscillation

When the RFC*n*TRG.SREF bit is set to 1 in Step 4 of the conversion procedure above, the RFC Ch.*n* starts CR oscillation using the reference resistor. The measurement counter starts counting up using the CR oscillation clock from the initial value that has been set. The time base counter starts counting up using TCCLK from 0x000000.

When the measurement counter or the time base counter overflows (0xffffff → 0x000000), the RFC*n*TRG.SREF bit is cleared to 0 and the reference oscillation stops automatically.

The measurement counter overflow sets the RFC*n*INTF.EREFIF bit to 1 indicating that the reference oscillation has been terminated normally. If the RFC*n*INTE.EREFIE bit = 1, a reference oscillation completion interrupt request occurs at this point.

The time base counter overflow sets the RFC*n*INTF.OVTCIF bit to 1 indicating that the reference oscillation has been terminated abnormally. If the RFC*n*INTE.OVTCIE bit = 1, a time base counter overflow error interrupt request occurs at this point.

Sensor oscillation

When the RFC*n*TRG.SSENA bit (sensor A) or the RFC*n*TRG.SSENB bit (sensor B) is set to 1 in Step 7 of the conversion procedure above, the RFC Ch.*n* starts CR oscillation using the sensor. The measurement counter starts counting up using the CR oscillation clock from 0x000000. The time base counter starts counting down using TCCLK from the value at the end of reference oscillation.

When the time base counter reaches 0x000000 or the measurement counter overflows (0xffffff → 0x000000), the RFC*n*TRG.SSENA bit or the RFC*n*TRG.SSENB bit that started oscillation is cleared to 0 and the sensor oscillation stops automatically.

The time base counter reaching 0x000000 sets the RFC*n*INTF.ESENAIF bit (sensor A) or the RFC*n*INTF.ESENBIF bit (sensor B) to 1 indicating that the sensor oscillation has been terminated normally. If the RFC*n*INTE.ESENAIE bit = 1 or the RFC*n*INTE.ESENBIE bit = 1, a sensor A or sensor B oscillation completion interrupt request occurs at this point.

The measurement counter overflow sets the RFC*n*INTF.OVMCIF to 1 indicating that the sensor oscillation has been terminated abnormally. If the RFC*n*INTE.OVMCIE bit = 1, a measurement counter overflow error interrupt request occurs at this point.

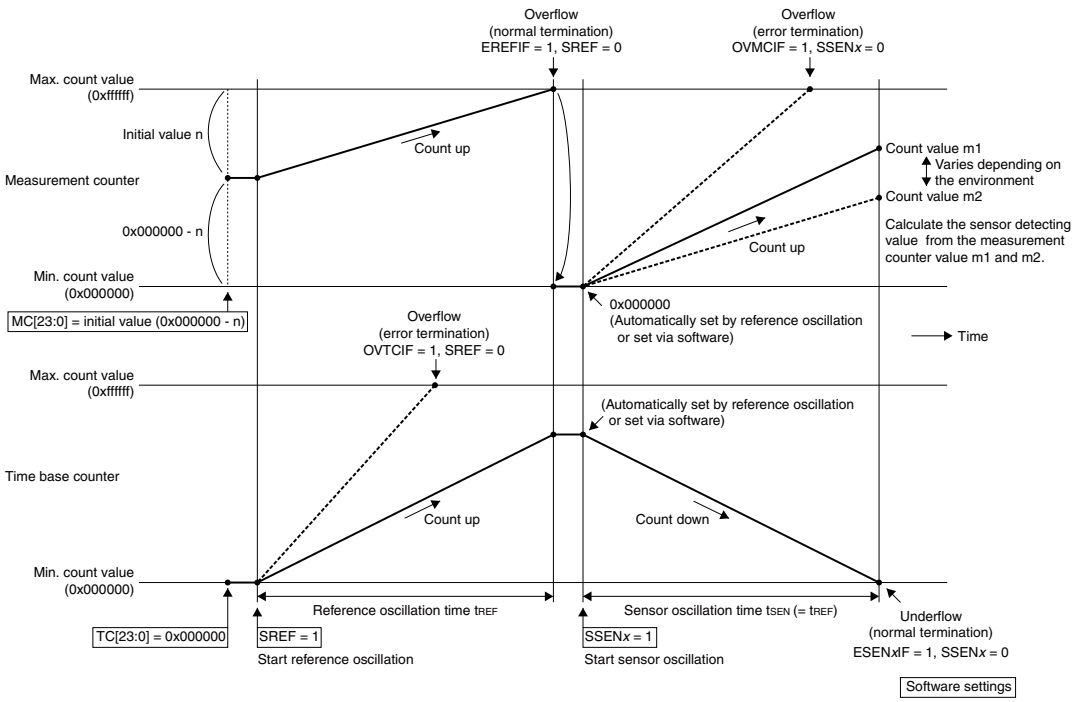


Figure 19.4.4.1 Counter Operations During Reference/Sensor Oscillation

Forced Termination

To abort reference oscillation or sensor oscillation, write 0 to the RFC*n*TRG.SREF bit (reference oscillation), the RFC*n*TRG.SSENA bit (sensor A oscillation), or the RFC*n*TRG.SSENB bit (sensor B oscillation) used to start the oscillation. The counters maintain the value at the point they stopped, note, however, that the conversion results cannot be guaranteed if the oscillation is resumed. When resuming oscillation, execute from counter initialization again.

Conversion Error

Performing reference oscillation and sensor oscillation with the same resistor and capacitor results n ≈ m. The difference between n and m is a conversion error. Table 19.4.4.1 lists the error factors. (n: measurement counter initial value, m: measurement counter value at the end of sensor oscillation)

Table 19.4.4.1 Error Factors

Error factor	Influence
External part tolerances	Large
Power supply voltage fluctuations	Large
Parasitic capacitance and resistance of the board	Middle
Temperature	Small
Unevenness of IC quality	Small

19.4.5 CR Oscillation Frequency Monitoring Function

The CR oscillation clock (RFCLK) generated during converting operation can be output from the RFCLKO*n* pin for monitoring. By setting the RFC*n*CTL.CONEN bit to 1, the RFC Ch.*n* enters continuous oscillation mode that disables oscillation stop conditions to continue oscillating operations. In this case, set the the RFC*n*TRG.SREF bit (reference oscillation), the RFC*n*TRG.SSENA bit (sensor A oscillation), or the RFC*n*TRG.SSENB bit (sensor B oscillation) to 1 to start oscillation. Set the bit to 0 to stop oscillation. Using this function helps easily measure the CR oscillation clock frequency. Furthermore, setting the RFC*n*CTL.RFCLKMD bit to 1 changes the output clock to the divided-by-two RFCLK clock.

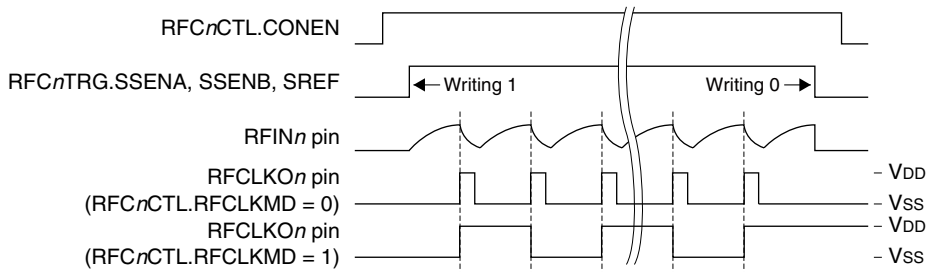


Figure 19.4.5.1 CR Oscillation Clock (RFCLK) Waveform

19.5 Interrupts

The RFC has a function to generate the interrupts shown in Table 19.5.1.

Table 19.5.1 RFC Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Reference oscillation completion	RFC*n*INTF.EREFIF	When reference oscillation has been completed normally due to a measurement counter overflow	Writing 1
Sensor A oscillation completion	RFC*n*INTF.ESENAIF	When sensor A oscillation has been completed normally due to the time base counter reaching 0x000000	Writing 1
Sensor B oscillation completion	RFC*n*INTF.ESENBIF	When sensor B oscillation has been completed normally due to the time base counter reaching 0x000000	Writing 1
Measurement counter overflow error	RFC*n*INTF.OVMCIF	When sensor oscillation has been terminated abnormally due to a measurement counter overflow	Writing 1
Time base counter overflow error	RFC*n*INTF.OVTCIF	When reference oscillation has been terminated abnormally due to a time base counter overflow	Writing 1

The RFC provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the “Interrupt Controller” chapter.

19.6 Control Registers

RFC Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*CLK	15–9	–	0x00	–	R	–
	8	DBRUN	1	H0	R/W	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the RFC operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits select the division ratio of the RFC operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of the RFC.

Table 19.6.1 Clock Source and Division Ratio Settings

RFC*n*CLK. CLKDIV[1:0] bits	RFC*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x3	1/8			1/1
0x2	1/4			
0x1	1/2			
0x0	1/1			

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The RFC*n*CLK register settings can be altered only when the RFC*n*CTL.MODEN bit = 0.

RFC Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*CTL	15–9	–	0x00	–	R	–
	8	RFCLKMD	0	H0	R/W	
	7	CONEN	0	H0	R/W	
	6	EVTEN	0	H0	R/W	
	5–4	SMODE[1:0]	0x0	H0	R/W	
	3–1	–	0x0	–	R	
	0	MODEN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 RFCLKMD

This bit sets the RFCLKO*n* pin to output the divided-by-two oscillation clock.

1 (R/W): Divided-by-two clock output

0 (R/W): Oscillation clock output

For more information, refer to “CR Oscillation Frequency Monitoring Function.”

Bit 7 CONEN

This bit disables the automatic CR oscillation stop function to enable continuous oscillation function.
1 (R/W): Enable continuous oscillation
0 (R/W): Disable continuous oscillation

For more information, refer to "CR Oscillation Frequency Monitoring Function."

Bit 6 EVTEN

This bit enables external clock input mode (event counter mode).
1 (R/W): External clock input mode
0 (R/W): Normal mode

For more information, refer to "Operating Modes."

Note: Do not input an external clock before the RFC*n*CTL.EVTEN bit is set to 1. The RFIN*n* pin is pulled down to Vss level when the port function is switched for the R/F converter.

Bits 5–4 SMODE[1:0]

These bits configure the oscillation mode. For more information, refer to "Operating Modes."

Table 19.6.2 Oscillation Mode Selection

RFC*n*CTL.SMODE[1:0] bits	Oscillation mode
0x3, 0x2	Reserved
0x1	AC oscillation mode for resistive sensor measurements
0x0	DC oscillation mode for resistive sensor measurements

Bits 3–1 Reserved

Bit 0 MODEN

This bit enables the RFC operations.
1 (R/W): Enable RFC operations (The operating clock is supplied.)
0 (R/W): Disable RFC operations (The operating clock is stopped.)

Note: If the RFC*n*CTL.MODEN bit is altered from 1 to 0 during R/F conversion, the counter value being converted cannot be guaranteed. R/F conversion cannot be resumed.

RFC Ch.*n* Oscillation Trigger Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*TRG	15–8	–	0x00	–	R	–
	7–3	–	0x00	–	R	
	2	SSENB	0	H0	R/W	
	1	SSENA	0	H0	R/W	
	0	SREF	0	H0	R/W	

Bits 15–3 Reserved

Bit 2 SSENB

This bit controls CR oscillation for sensor B. This bit also indicates the CR oscillation status.
1 (W): Start oscillation
0 (W): Stop oscillation
1 (R): Being oscillated
0 (R): Stopped

Note: Writing 1 to the RFC*n*TRG.SSENB bit does not start oscillation when the RFC*n*CTL.SMODE[1:0] bits = 0x1 (AC oscillation mode for resistive sensor measurements).

Bit 1 SSENA

This bit controls CR oscillation for sensor A. This bit also indicates the CR oscillation status.
1 (W): Start oscillation
0 (W): Stop oscillation
1 (R): Being oscillated
0 (R): Stopped

Bit 0 SREF

This bit controls CR oscillation for the reference resistor. This bit also indicates the CR oscillation status.

1 (W): Start oscillation
0 (W): Stop oscillation
1 (R): Being oscillated
0 (R): Stopped

Notes: • Settings in this register are all ineffective when the RFC*n*CTL.MODEN bit = 0 (RFC operation disabled).

• When writing 1 to the RFC*n*TRG.SREF bit, the RFC*n*TRG.SSENA bit, or the RFC*n*TRG.SSENB bit to start oscillation, be sure to avoid having more than one bit set to 1.

• Be sure to clear the interrupt flags (RFC*n*INTF.EREFIF bit, RFC*n*INTF.ESENAIF bit, RFC*n*INTF.ESENBIF bit, RFC*n*INTF.OVMCIF bit, and RFC*n*INTF.OVTCIF bit) before starting oscillation using this register.

RFC Ch.*n* Measurement Counter Low and High Registers

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*MCL	15–0	MC[15:0]	0x0000	H0	R/W	–
RFC*n*MCH	15–8	–	0x00	–	R	–
	7–0	MC[23:16]	0x00	H0	R/W	

Or

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*MCL	31–24	–	0x00	–	R	–
RFC*n*MCH	23–0	MC[23:0]	0x000000	H0	R/W	

Bits 31–24 Reserved

Bits 23–0 MC[23:0]

Measurement counter data can be read and written through these bits.

Note: The measurement counter must be set from the low-order value (RFC*n*MCL.MC[15:0] bits) first when data is set using a 16-bit access instruction. The counter may not be set to the correct value if the high-order value (RFC*n*MCH.MC[23:16] bits) is written first.

RFC Ch.*n* Time Base Counter Low and High Registers

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*TCL	15–0	TC[15:0]	0x0000	H0	R/W	–
RFC*n*TCH	15–8	–	0x00	–	R	–
	7–0	TC[23:16]	0x00	H0	R/W	

Or

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*TCL	31–24	–	0x00	–	R	–
RFC*n*TCH	23–0	TC[23:0]	0x000000	H0	R/W	

Bits 31–24 Reserved

Bits 23–0 TC[23:0]

Time base counter data can be read and written through these bits.

Note: The time base counter must be set from the low-order value (RFC*n*TCL.TC[15:0] bits) first when data is set using a 16-bit access instruction. The counter may not be set to the correct value if the high-order value (RFC*n*TCH.TC[23:16] bits) is written first.

RFC Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*INTF	15–8	–	0x00	–	R	–
	7–5	–	0x0	–	R	
	4	OVTCIF	0	H0	R/W	Cleared by writing 1.
	3	OVMCIF	0	H0	R/W	
	2	ESENBIF	0	H0	R/W	
	1	ESENAIF	0	H0	R/W	
	0	EREFIF	0	H0	R/W	

Bits 15–5 Reserved

Bit 4 OVTCIF
Bit 3 OVMCIF
Bit 2 ESENBIF
Bit 1 ESENAIF
Bit 0 EREFIF

These bits indicate the RFC interrupt cause occurrence status.
1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Clear flag
0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:
RFC*n*INTF.OVTCIF bit: Time base counter overflow error interrupt
RFC*n*INTF.OVMCIF bit: Measurement counter overflow error interrupt
RFC*n*INTF.ESENBIF bit: Sensor B oscillation completion interrupt
RFC*n*INTF.ESENAIF bit: Sensor A oscillation completion interrupt
RFC*n*INTF.EREFIF bit: Reference oscillation completion interrupt

RFC Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC*n*INTE	15–8	–	0x00	–	R	–
	7–5	–	0x0	–	R	
	4	OVTCIE	0	H0	R/W	
	3	OVMCIE	0	H0	R/W	
	2	ESENBIE	0	H0	R/W	
	1	ESENAIE	0	H0	R/W	
	0	EREFIE	0	H0	R/W	

Bits 15–5 Reserved

Bit 4 OVTCIE
Bit 3 OVMCIE
Bit 2 ESENBIE
Bit 1 ESENAIE
Bit 0 EREFIE

These bits enable RFC interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
RFC*n*INTE.OVTCIE bit: Time base counter overflow error interrupt
RFC*n*INTE.OVMCIE bit: Measurement counter overflow error interrupt
RFC*n*INTE.ESENBIE bit: Sensor B oscillation completion interrupt
RFC*n*INTE.ESENAIE bit: Sensor A oscillation completion interrupt
RFC*n*INTE.EREFIE bit: Reference oscillation completion interrupt

20 Temperature Detection Circuit (TEM)

20.1 Overview

TEM is a temperature detection circuit including a temperature sensor. The features of TEM are listed below.

- Temperature sensor detectable temperature range: -20 °C to 70 °C
- Sensor output voltage to digital value (8 bits) conversion circuit
- Conversion time (comparison time) adjustment function
  * TEM compares the sensor detection voltage with the reference voltage.
- Can generate conversion completion interrupts.

Figure 20.1.1 shows TEM configuration.

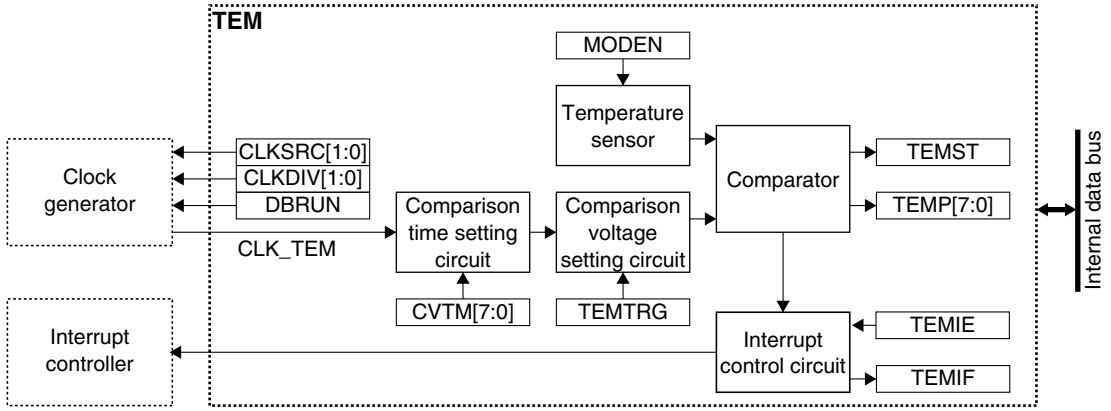


Figure 20.1.1 TEM Configuration

20.2 Clock Settings

20.2.1 TEM Operating Clock

When using TEM, the TEM operating clock CLK_TEM must be supplied to TEM from the clock generator. The CLK_TEM supply should be controlled as in the procedure shown below.

1. Enable the clock source in the clock generator if it is stopped (refer to "Clock Generator" in the "Power Supply, Reset, and Clocks" chapter).
2. Set the following TEMCLK register bits:
   - TEMCLK.CLKSRC[1:0] bits (Clock source selection)
   - TEMCLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)

20.2.2 Clock Supply in SLEEP Mode

When using TEM during SLEEP mode, the TEM operating clock CLK_TEM must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_TEM clock source.

20.2.3 Clock Supply in DEBUG Mode

The CLK_TEM supply during DEBUG mode should be controlled using the TEMCLK.DBRUN bit.
The CLK_TEM supply to TEM is suspended when the CPU enters DEBUG mode if the TEMCLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_TEM supply resumes. Although TEM stops operating when the CLK_TEM supply is suspended, the registers retain the status before DEBUG mode was entered. If the TEMCLK.DBRUN bit = 1, the CLK_TEM supply is not suspended and TEM will keep operating in DEBUG mode.

20.3 Operations

20.3.1 Initialization

TEM should be initialized with the procedure shown below.

1. Set the TEMCLK.CLKSRC[1:0] and TEMCLK.CLKDIV[1:0] bits. (Configure operating clock)
2. Set the TEMTMG.CVTM[7:0] bits. (Set comparison time)
3. Set the following bits when using the interrupt:
   - Write 1 to the TEMINTF.TEMIF bit. (Clear interrupt flag)
   - Set the TEMINTE.TEMIE bit to 1. (Enable interrupt)

20.3.2 Comparison Time Setting

Set the time for comparing each comparison voltage by the comparator to the TEMTMG.CVTM[7:0] bits.
Be sure to set a 150 µs or more comparison time including clock frequency dispersion.

$$\text{Comparison time} = \frac{\text{CVTM} + 1}{\text{CLK\_TEM}} \geq 150\ [\mu s] \qquad \text{(Eq. 20.1)}$$

Where
CVTM: TEMTMG.CVTM[7:0] bit setting value (0 to 255)
CLK_TEM: CLK_TEM frequency [Hz]

20.3.3 Temperature Detection

A temperature detection control procedure and operations of TEM are shown below.

Control procedure

1. Set the TEMCTL.MODEN bit to 1. (Enable TEM/Activate temperature sensor)
2. Wait a time interval at least 10 ms required for the temperature sensor to stabilize the operation.
3. Write 1 to the TEMCTL.TEMTRG bit. (Start temperature conversion)
4. After a TEM interrupt has occurred, read the conversion results from the TEMRSLT.TEMP[7:0] bits.

Temperature detecting operations

Writing 1 to the TEMCTL.MODEN bit turns the temperature sensor on to start temperature detection.
The temperature conversion operation starts by writing 1 to the TEMCTL.TEMTRG bit after taking the temperature sensor stabilization wait time.
The comparator converts the temperature sensor output voltage (analog value that varies according to temperature) into an 8-bit digital value and stores the results to the TEMRSLT.TEMP[7:0] bits. The following shows the conversion time:

Conversion time = Comparison time (Eq. 20.1) × 8 (Eq. 20.2)

Example: Conversion time = 1,200 µs when Comparison time = 150 µs.

The TEMINTF.TEMST bit is set to 1 during converting operation and reverts to 0 after the conversion has finished. Note, however, that a maximum one CLK_TEM cycle of delay occurs until the TEMINTF.TEMST bit is set to 1 after 1 is written to the TEMCTL.TEMTRG bit.
When the conversion operation has finished, the interrupt flag (TEMINTF.TEMIF bit) is set to 1.

The temperature conversion being executed can be terminated by writing 0 to the TEMCTL.TEMTRG bit. The conversion results become invalid in this case.

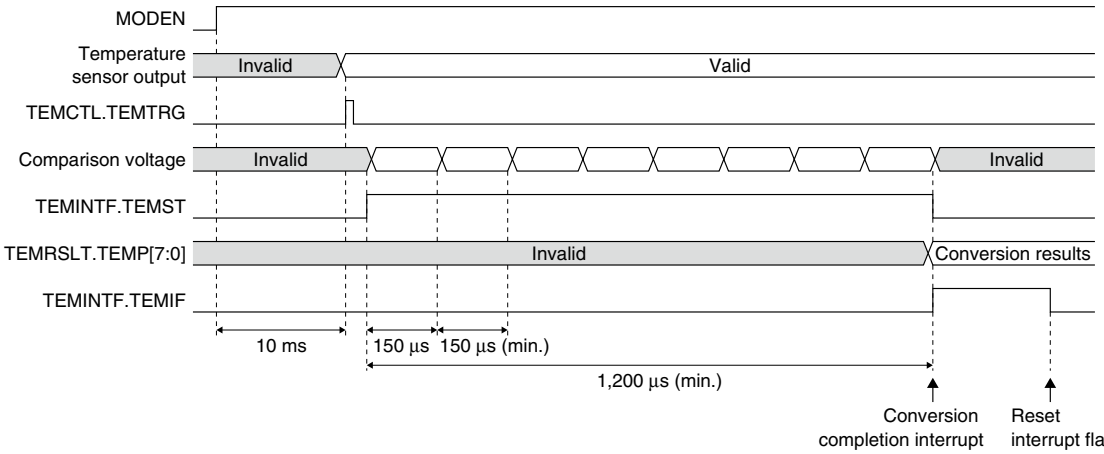


Figure 20.3.3.1 Temperature Conversion Operation

Correspondence between detection results and temperature

The table below lists the temperature (within the detection range) corresponding to the 8-bit value read from the TEMRSLT.TEMP[7:0] bits.

Table 20.3.3.1 Detected Temperature

Out of the guaranteed range		Guaranteed range		Out of the guaranteed range	
TEMP[7:0]	Temperature (°C)	TEMP[7:0]	Temperature (°C)	TEMP[7:0]	Temperature (°C)
		0xb5	0.1	0x80	50.5
		0xb4	1.1	0x7f	51.5
		0xb3	2.0	0x7e	52.4
		0xb2	3.0	0x7d	53.3
		0xb1	3.9	0x7c	54.3
		0xb0	4.9	0x7b	55.2
		0xaf	5.9	0x7a	56.2
		0xae	6.8	0x79	57.1
		0xad	7.8	0x78	58.0
		0xac	8.7	0x77	59.0
		0xab	9.7	0x76	59.9
		0xaa	10.7	0x75	60.8
		0xa9	11.6	0x74	61.8
		0xa8	12.6	0x73	62.7
		0xa7	13.5	0x72	63.6
		0xa6	14.5	0x71	64.6
0xff–0xca	Invalid	0xa5	15.5	0x70	65.5
		0xa4	16.4	0x6f	66.4
		0xa3	17.4	0x6e	67.4
		0xa2	18.3	0x6d	68.3
		0xa1	19.3	0x6c	69.2
		0xa0	20.2	(0x6b)	(70.1)
		0x9f	21.2		
		0x9e	22.1		
		0x9d	23.1		
		0x9c	24.0		
		0x9b	25.0		
		0x9a	26.0		
		0x99	26.9		
		0x98	27.9		
		0x97	28.8		
		0x96	29.8		
(0xca)	(-20.3)	0x95	30.7		
0xc9	-19.3	0x94	31.7		
0xc8	-18.3	0x93	32.6		
0xc7	-17.3	0x92	33.5		
0xc6	-16.4	0x91	34.5		
0xc5	-15.4	0x90	35.4	0x6b–0x0	Invalid
0xc4	-14.4	0x8f	36.4		
0xc3	-13.5	0x8e	37.3		
0xc2	-12.5	0x8d	38.3		
0xc1	-11.5	0x8c	39.2		
0xc0	-10.5	0x8b	40.2		
0xbf	-9.6	0x8a	41.1		
0xbe	-8.6	0x89	42.1		
0xbd	-7.6	0x88	43.0		
0xbc	-6.7	0x87	43.9		
0xbb	-5.7	0x86	44.9		
0xba	-4.7	0x85	45.8		
0xb9	-3.8	0x84	46.8		
0xb8	-2.8	0x83	47.7		
0xb7	-1.8	0x82	48.7		
0xb6	-0.9	0x81	49.6		

Error within the guaranteed temperature range (0°C to 50°C): ±5°C

The effective range of the converted values is 0x6c to 0xc9. If the read value of the TEMRSLT.TEMP[7:0] bits is out of the range, it should be handled as a conversion error.

Note: The detection results are the temperature inside the device detected by the sensor embedded in the device.

20.4 Interrupt

TEM has a function to generate the interrupt shown in Table 20.4.1.

Table 20.4.1 TEM Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Conversion completion	TEMINTF.TEMIF	When the temperature conversion operation has completed	TEMRSLT register read

The TEM provides an interrupt enable bit corresponding to the interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

20.5 Control Registers

TEM Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TEMCLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x0	H0	R/W	
	3–2	–	0x0	–	R	
	1–0	CLKSRC[1:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the TEM operating clock is supplied in DEBUG mode or not.

1 (R/W): Clock supplied in DEBUG mode

0 (R/W): No clock supplied in DEBUG mode

Bits 7–6 Reserved

Bits 5–4 CLKDIV[1:0]

These bits select the division ratio of the TEM operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of TEM.

Table 20.5.1 Clock Source and Division Ratio Settings

TEMCLK. CLKDIV[1:0] bits	TEMCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	OSC3B	OSC1	OSC3A	EXOSC
0x3	1/8	1/1	1/8	1/1
0x2	1/4		1/4	
0x1	1/2		1/2	
0x0	1/1		1/1	

(Note) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The TEMCLK register settings can be altered only when the TEMCTL.MODEN bit = 0.

TEM Timing Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TEMTMG	15–8	–	0x00	–	R	–
	7–0	CVTM[7:0]	0x00	H0	R/W	

Bits 15–8 Reserved

Bits 7–0 CVTM[7:0]

These bits set the time for comparing the sensor output with the comparison voltage using the comparator (see Eq. 20.1).

TEM Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TEMCTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	TEMTRG	0	H0	W	Always read as 0.
	0	MODEN	0	H0	R/W	–

Bits 15–2 Reserved

Bit 1 TEMTRG

This bit starts temperature conversion.

1 (W): Start conversion

0 (W): Stop conversion

0 (R): Always 0 when being read

Bit 0 MODEN

This bit enables the TEM operations.

1 (R/W): Enable TEM operations (The operating clock is supplied.)

0 (R/W): Disable TEM operations (The operating clock is stopped.)

TEM Conversion Result Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TEMRSLT	15–8	–	0x00	–	R	–
	7–0	TEMP[7:0]	0x00	H0	R	

Bits 15–8 Reserved

Bits 7–0 TEMP[7:0]

The temperature conversion results are stored in these bits.

Invalid data is read before the conversion has completed.

For correspondence between the read value and temperature, see Table 20.4.3.1.

TEM Interrupt Flag and Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TEMINTF	15–8	–	0x00	–	R	–
	7–5	–	0x0	–	R	
	4	TEMST	0	H0	R	
	3–1	–	0x0	–	R	
	0	TEMIF	0	H0	R/W	Cleared by reading the TEMRSLT register.

Bits 15–5 Reserved

Bit 4 TEMST

This bit indicates the temperature conversion operating status.

1 (R): Converting

0 (R): Idle

Bits 3–1 Reserved

Bit 0 TEMIF

This bit indicates the TEM interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag

0 (W): Ineffective

TEM Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TEMINTE	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	TEMIE	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 TEMIE

This bit enables TEM interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

21 Multiplier/Divider (COPRO)

21.1 Overview

COPRO is the coprocessor that provides multiplier/divider functions. The features of COPRO are listed below.

- Multiplication: Supports signed/unsigned multiplications.
  (16 bits × 16 bits = 32 bits)
  Can be executed in 1 cycle.
- Multiplication and accumulation (MAC): Supports signed MAC operations with overflow detection function.
  (16 bits × 16 bits + 32 bits = 32 bits)
  Can be executed in 1 cycle.
- Division: Supports signed/unsigned divisions.
  (16 bits ÷ 16 bits = 16 bits with 16-bit residue)
  Can be executed in 17 to 20 cycles.

Figure 21.1.1 shows the COPRO configuration.

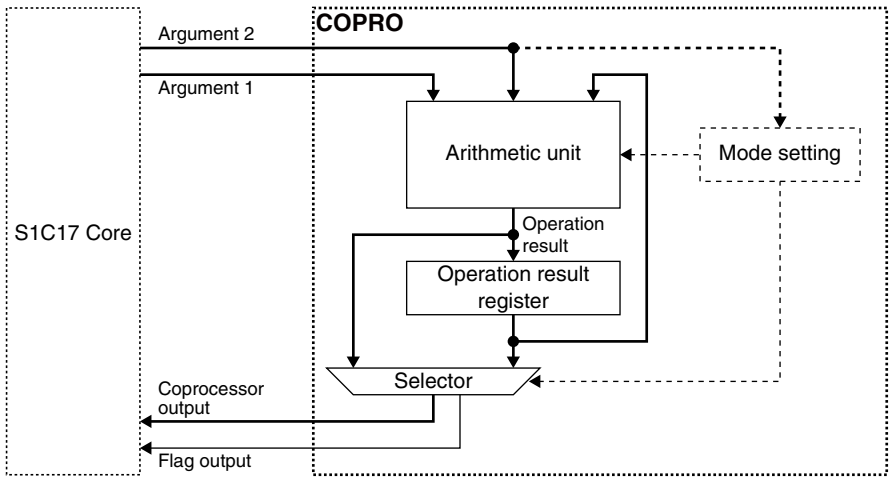


Figure 21.1.1 COPRO Configuration

21.2 Operation Mode and Output Mode

COPRO operates according to the operation mode specified by the application program. As listed in Table 21.2.1, COPRO supports nine operations.
The multiplication, division and MAC results are 32-bit data, therefore, the S1C17 Core cannot read them in one access cycle. The output mode is provided to specify the high-order 16 bits or low-order 16 bits of the operation results to be read from COPRO.
The operation and output modes can be specified with a 7-bit data by writing it to the mode setting register in COPRO. Use a "`ld.cw`" instruction for this writing.

```
ld.cw   %rd,%rs      %rs[6:0] is written to the mode setting register. (%rd: not used)
ld.cw   %rd,imm7     imm7[6:0] is written to the mode setting register. (%rd: not used)
```

6 4	3 0
Output mode setting value	Operation mode setting value

Figure 21.2.1 Mode Setting Register

Table 21.2.1 Mode Settings

Setting value (D[6:4])	Output mode	Setting value (D[3:0])	Operation mode
0x0	**16 low-order bits output mode** The low-order 16 bits of operation results can be read as the coprocessor output.	0x0	**Initialize mode 0** Clears the operation result register to 0x0.
0x1	**16 high-order bits output mode** The high-order 16 bits of operation results can be read as the coprocessor output.	0x1	**Initialize mode 1** Loads the 16-bit augend into the low-order 16 bits of the operation result register.
0x2–0x7	Reserved	0x2	**Initialize mode 2** Loads the 32-bit augend into the operation result register.
		0x3	**Operation result read mode** Outputs the data in the operation result register without computation.
		0x4	**Unsigned multiplication mode** Performs unsigned multiplication.
		0x5	**Signed multiplication mode** Performs signed multiplication.
		0x6	Reserved
		0x7	**Signed MAC mode** Performs signed MAC operation.
		0x8	**Unsigned division mode** Performs unsigned division.
		0x9	**Signed division mode** Performs signed division.
		0xa–0xf	Reserved

21.3 Multiplication

The multiplication function performs "A (32 bits) = B (16 bits) × C (16 bits)."
The following shows a procedure to perform a multiplication:

1. Set the mode to 0x04 (unsigned multiplication, 16 low-order bits output mode) or 0x05 (signed multiplication, 16 low-order bits output mode).
2. Send the 16-bit multiplicand (B) and 16-bit multiplier (C) to COPRO using a "`ld.ca`" instruction.
3. Read the one-half result (16 low-order bits = A[15:0]) and the flag status.
4. Set the mode to 0x13 (operation result read, 16 high-order bits output mode).
5. Read another one-half result (16 high-order bits = A[31:16]).

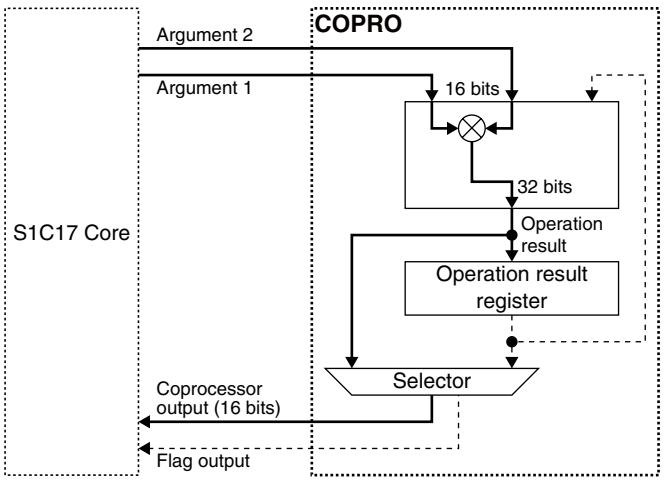


Figure 21.3.1 Data Path in Multiplication Mode

Table 21.3.1 Operation in Multiplication Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x04 or 0x05	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs %rd ← res[15:0]	psr (CVZN) ← 0b0000	The operation result register keeps the operation result until it is rewritten by other operation.
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* %rd ← res[15:0]		
0x14 or 0x15	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs %rd ← res[31:16]		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* %rd ← res[31:16]		

res: operation result register

Example:

```
ld.cw %r0,0x4   ; Sets the mode (unsigned multiplication mode and 16 low-order bits output mode).
ld.ca %r0,%r1   ; Performs "res = %r0 × %r1" and loads the 16 low-order bits of the result to %r0.
ld.cw %r0,0x13  ; Sets the mode (operation result read mode and 16 high-order bits output mode).
ld.ca %r1,%r0   ; Loads the 16 high-order bits of the result to %r1.
```

21.4 Division

The division function performs "B (16 bits) ÷ C (16 bits) = A (16 bits), residue D (16 bits)."
The following shows a procedure to perform a division:

1. Set the mode to 0x08 (unsigned division, 16 low-order bits output mode) or 0x09 (signed division, 16 low-order bits output mode).
2. Send the 16-bit dividend (B) and 16-bit divisor (C) to COPRO using a "`ld.ca`" instruction.
3. Read the one-half result (16 low-order bits = quotient) and the flag status.
4. Set the mode to 0x13 (operation result read, 16 high-order bits output mode).
5. Read another one-half result (16 high-order bits = residue).

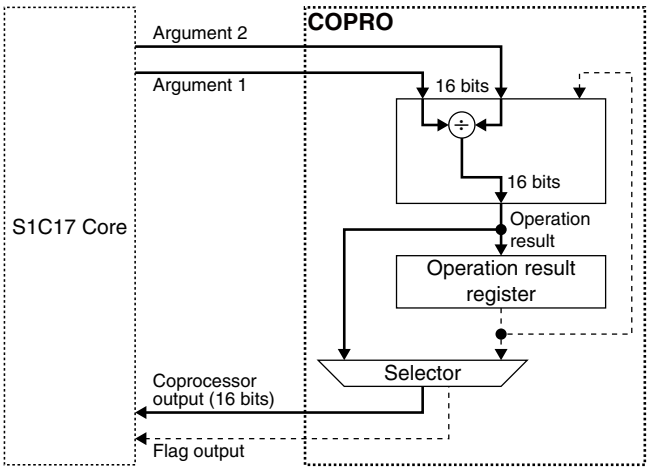


Figure 21.4.1 Data Path in Division Mode

Table 21.4.1 Operation in Division Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x08 or 0x09	`ld.ca %rd,%rs`	res[31:0] ← %rd ÷ %rs %rd ← res[15:0] (quotient)	psr (CVZN) ← 0b0000	The operation result register keeps the operation result until it is rewritten by other operation.
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd ÷ *imm7/16* %rd ← res[15:0] (quotient)		
0x018 or 0x19	`ld.ca %rd,%rs`	res[31:0] ← %rd ÷ %rs %rd ← res[31:16] (residue)		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd ÷ *imm7/16* %rd ← res[31:16] (residue)		

res: operation result register

Example:

```
ld.cw %r0,0x8  ; Sets the mode (unsigned division mode and 16 low-order bits output mode).
ld.ca %r0,%r1  ; Performs "res = %r0 ÷ %r1" and loads the 16 low-order bits of the result (quotient) to %r0.
ld.cw %r0,0x13 ; Sets the mode (operation result read mode and 16 high-order bits output mode).
ld.ca %r1,%r0  ; Loads the 16 high-order bits of the result (residue) to %r1.
```

21.5 MAC

The MAC (multiplication and accumulation) function performs "A (32 bits) = B (16 bits) × C (16 bits) + A (32 bits)."
The following shows a procedure to perform a MAC operation:

1. Set the initial value (A) to the operation result register.
   - To clear the operation result register (A = 0):
     Set the mode to 0x00 (initialize mode 0). (It is not necessary to send 0x00 to COPRO with another instruction.)
   - To load a 16-bit value to the operation result register:
     Set the operation mode to 0x01 (initialize mode 1) and then send the initial value (16 bits) to COPRO using a "`ld.cf`" instruction.
   - To load a 32-bit value to the operation result register:
     Set the operation mode to 0x02 (initialize mode 2) and then send the initial value (32 bits) to COPRO using a "`ld.cf`" instruction.
2. Set the mode to 0x07 (signed MAC, 16 low-order bits output mode).
3. Repeat sending the 16-bit multiplicand (B) and 16-bit multiplier (C) to COPRO the number of times required using a "`ld.ca`" instruction.
4. Read the one-half result (16 low-order bits = A[15:0]) and the flag status.
5. Set the mode to 0x13 (operation result read, 16 high-order bits output mode).
6. Read another one-half result (16 high-order bits = A[31:16]).

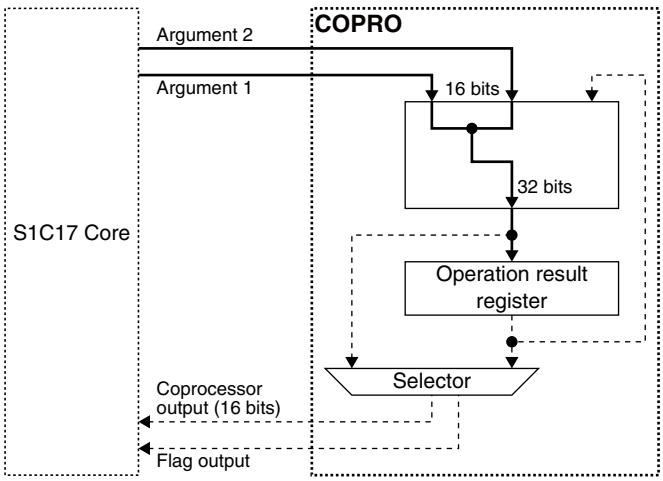


Figure 21.5.1 Data Path in Initialize Mode

Table 21.5.1 Initializing the Operation Result Register

Mode setting value	Instruction	Operations	Remarks
0x00	–	res[31:0] ← 0x0	Setting the operating mode executes the initialization without sending data.
0x01	`ld.cf %rd,%rs`	res[31:16] ← 0x0 res[15:0] ← %rs	
	(`ext` *imm9*) `ld.cf %rd,`*imm7*	res[31:16] ← 0x0 res[15:0] ← *imm7/16*	
0x02	`ld.cf %rd,%rs`	res[31:16] ← %rd res[15:0] ← %rs	
	(`ext` *imm9*) `ld.cf %rd,`*imm7*	res[31:16] ← %rd res[15:0] ← *imm7/16*	

res: operation result register

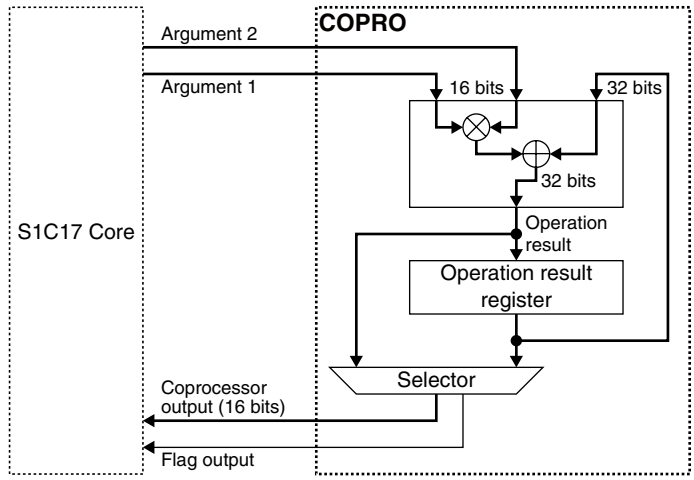


Figure 21.5.2 Data Path in MAC Mode

Table 21.5.2 Operation in MAC Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x07	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs + res[31:0] %rd ← res[15:0]	psr (CVZN) ← 0b0100 if an overflow has occurred Otherwise psr (CVZN) ← 0b0000	The operation result register keeps the operation result until it is rewritten by other operation.
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* + res[31:0] %rd ← res[15:0]		
0x17	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs + res[31:0] %rd ← res[31:16]		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* + res[31:0] %rd ← res[31:16]		

res: operation result register

Example:

```
ld.cw %r0,0x0   ; Sets the mode (initialize mode 0).
ld.cw %r0,0x7   ; Sets the mode (signed MAC mode and 16 low-order bits output mode).
ld.ca %r0,%r1   ; Performs "res = %r0 × %r1 + res" and loads the 16 low-order bits of the result to %r0.
ld.cw %r0,0x13  ; Sets the mode (operation result read mode and 16 high-order bits output mode).
ld.ca %r1,%r0   ; Loads the 16 high-order bits of the result to %r1.
```

Conditions to set the overflow (V) flag

An overflow occurs in a MAC operation and the overflow (V) flag is set to 1 when the signs of the multiplication result, operation result register value, and multiplication & accumulation result match the following conditions:

Table 21.5.3 Conditions to Set the Overflow (V) Flag

Mode setting value	Sign of multiplication result	Sign of operation result register value	Sign of multiplication & accumulation result
0x07	0 (positive)	0 (positive)	1 (negative)
0x07	1 (negative)	1 (negative)	0 (positive)

An overflow occurs when a MAC operation performs addition of positive values and a negative value results, or it performs addition of negative values and a positive value results. The coprocessor holds the operation result until the overflow (V) flag is cleared.

Conditions to clear the overflow (V) flag

The overflow (V) flag that has been set will be cleared when an overflow has not been occurred during execution of the "`ld.ca`" instruction for MAC operation or when the "`ld.ca`" or "`ld.cf`" instruction is executed in an operation mode other than operation result read mode.

21.6 Reading Operation Results

The "`ld.ca`" instruction cannot load a 32-bit operation result to a CPU register, so a multiplication, division or MAC operation returns the one-half (16 bits according to the output mode) result (A[15:0] or A[31:16]) and the flag status to the CPU registers. Another one-half should be read by setting COPRO into operation result read mode. The operation result register keeps the loaded operation result until it is rewritten by other operation.

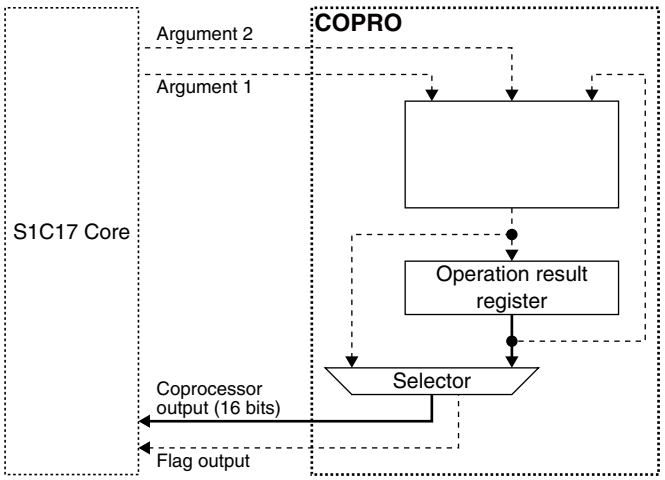


Figure 21.6.1 Data Path in Operation Result Read Mode

Table 21.6.1 Operation in Operation Result Read Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x03	`ld.ca %rd,%rs`	%rd ← res[15:0]	psr (CVZN) ← 0b0000	This operation mode does not affect the operation result register.
	`ld.ca %rd,`*`imm7`*	%rd ← res[15:0]		
0x13	`ld.ca %rd,%rs`	%rd ← res[31:16]		
	`ld.ca %rd,`*`imm7`*	%rd ← res[31:16]		

res: operation result register

22 Electrical Characteristics

22.1 Absolute Maximum Ratings

(V_{SS} = 0 V)

Item	Symbol	Condition		Rated value	Unit
Power supply voltage	V_{DD}			-0.3 to 4.0	V
Input voltage	V_I	P10–17, P31, #RESET		-0.3 to V_{DD} + 0.5	V
		P00–07, P20–27, P30, P32–37, P40–41, PD0–D1		-0.3 to 4.0	V
Output voltage	V_O	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D2		-0.3 to V_{DD} + 0.5	V
High level output current	I_{OH}	1 pin	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D2	-10	mA
		Total of all pins		-20	mA
Low level output current	I_{OL}	1 pin	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D2	10	mA
		Total of all pins		20	mA
Operating temperature	Ta			-20 to 70	°C
Storage temperature	Tstg			-65 to 125	°C

22.2 Recommended Operating Conditions

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Power supply voltage	V_{DD}		2.0	–	3.6	V
Flash programming voltage	V_{PP}		7.3	7.5	7.7	V
OSC1A oscillator oscillation frequency	f_{OSC1A}	Crystal resonator	–	32.768	–	kHz
OSC3A oscillator oscillation frequency	f_{OSC3A}	Crystal or ceramic resonator	0.2	–	20	MHz
EXOSC external clock frequency	f_{EXOSC}	When supplied from an external oscillator	0.016	–	16.3	MHz
Capacitor between V_{SS} and V_{DD}	C_{PW1-1}		–	3.3	–	µF
	C_{PW1-2}		–	10	–	µF
	C_{PW1-3}		–	0.1	–	µF
Capacitor between V_{SS} and V_{D1}	C_{PW2}		–	1.0	–	µF
Capacitor between V_{SS} and V_{OSC}	C_{PW3}		–	0.1	–	µF
Capacitor between C_{1P} and C_{1N}	C_{PW4}	*1	–	0.22	–	µF
Capacitor between V_{SS} and C_{1H}	C_{PW5}	*1	–	0.22	–	µF
Capacitor between C_{2P} and C_{2N}	C_{PW6}	*1	–	0.22	–	µF
Capacitor between V_{SS} and V_{PP}	C_{PW7}	*1	–	0.22	–	µF
DSIO pull-up resistor	R_{DBG}		–	10	–	kΩ

*1 The C_{1P}, C_{1N}, C_{1H}, C_{2P}, and C_{2N} pins can be left open when Flash programming is not performed.

22.3 Current Consumption

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C, EXOSC = OFF,
PWGVD1CTL.REGMODE[1:0] bits = 0x0 (automatic mode), FLASHCWAIT.RDWAIT[1:0] bits = 0x0 (1 cycle)

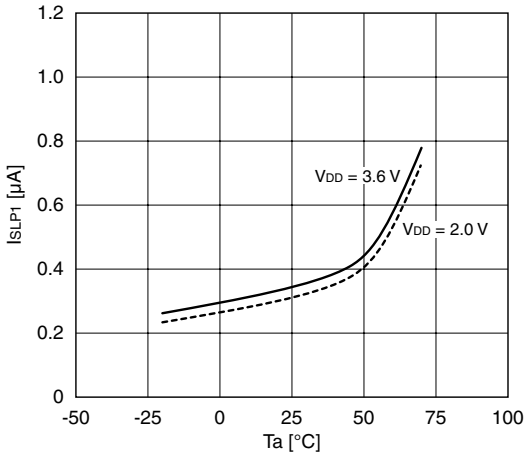
Item	Symbol	Condition	V_{DD}	Ta	Min.	Typ.	Max.	Unit
Current consumption in SLEEP mode	I_{SLP1}	OSC1A = OFF, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF	3.6 V	25 °C	–	0.35	0.70	µA
				70 °C	–	1.20	3.00	µA
	I_{SLP2}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, RTCCTL.RTCRUN bit = 1			–	0.55	3.20	µA
Current consumption in HALT mode	I_{HALT1}	OSC1A = OFF, OSC1B = ON, OSC3A = OFF, OSC3B = OFF			–	9.3	13.6	µA
	I_{HALT2}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF			–	0.78	3.38	µA
	I_{HALT3}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = 20 MHz			–	466	535	µA
	I_{HALT4}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = 20 MHz, OSC3B = OFF			–	375	453	µA
Current consumption in RUN mode	I_{RUN10}	OSC1A = 32 kHz, OSC1B = ON, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1B, executed on Flash*1			–	20.3	25.5	µA
		OSC1A = 32 kHz, OSC1B = ON, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1B/2, executed on Flash*1			–	14.8	19.7	µA
	I_{RUN20}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1A, executed on Flash*1			–	11.9	14.7	µA
		OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1A, executed on Flash*1, PWGVD1CTL.REGMODE[1:0] bits = 0x2 (normal mode)			–	19.6	24.1	µA
		OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1A/2, executed on Flash*1			–	6.35	9.13	µA
	I_{RUN30}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = 20 MHz, SYSCLK = OSC3B, executed on Flash*1			–	5,500	5,900	µA
		OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = 8 MHz, SYSCLK = OSC3B/2, executed on Flash*1			–	1,620	1,710	µA
	I_{RUN40}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = 20 MHz, OSC3B = OFF, SYSCLK = OSC3A, executed on Flash*1			–	5,430	5,740	µA
		OSC1A = 32 kHz, OSC1B = OFF, OSC3A = 4 MHz, OSC3B = OFF, SYSCLK = OSC3A, executed on Flash*1			–	1,450	1,540	µA
		OSC1A = 32 kHz, OSC1B = OFF, OSC3A = 20 MHz, OSC3B = OFF, SYSCLK = OSC3A/8, executed on Flash*1			–	83.0	94.4	µA
	I_{RUN11}	OSC1A = OFF, OSC1B = ON, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1B, executed on RAM*2			–	16.1	21.1	µA
	I_{RUN21}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, SYSCLK = OSC1A, executed on RAM*2			–	7.7	10.4	µA
	I_{RUN31}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = 20 MHz, SYSCLK = OSC3B, executed on RAM*2			–	4,580	4,860	µA
	I_{RUN41}	OSC1A = 32 kHz, OSC1B = OFF, OSC3A = 20 MHz, OSC3B = OFF, SYSCLK = OSC3A, executed on RAM*2			–	4,650	5,010	µA

*1 The current consumption values were measured when a test program consisting of 60.5 % ALU instructions, 17 % branch instructions, 12 % RAM read instructions, and 10.5 % RAM write instructions was executed continuously in the Flash memory.

*2 The current consumption values were measured when a test program consisting of 60.5 % ALU instructions, 17 % branch instructions, 12 % RAM read instructions, and 10.5 % RAM write instructions was executed continuously in the RAM.

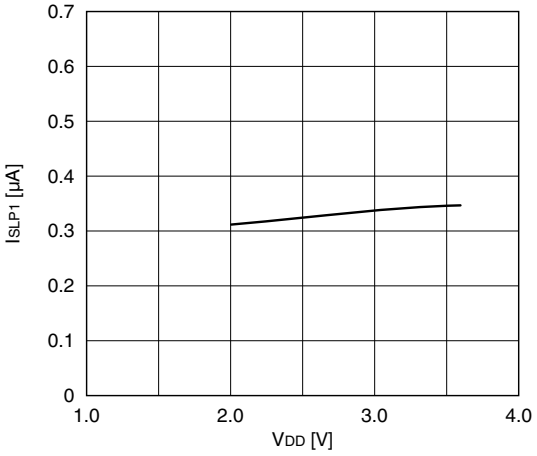
Current consumption-temperature characteristic in SLEEP mode

OSC1A = OFF, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, Typ. value



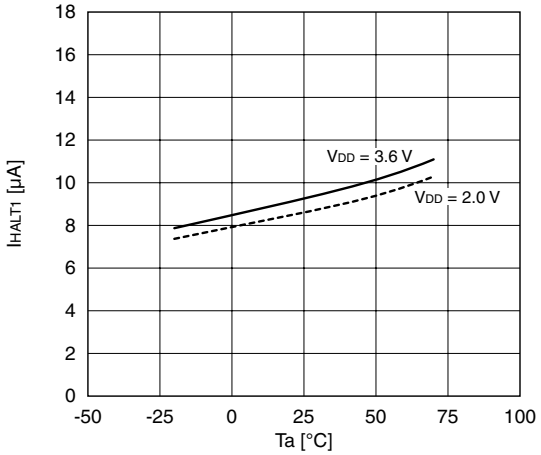
Current consumption-power supply voltage characteristic in SLEEP mode

OSC1A = OFF, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, Ta = 25 °C, Typ. value



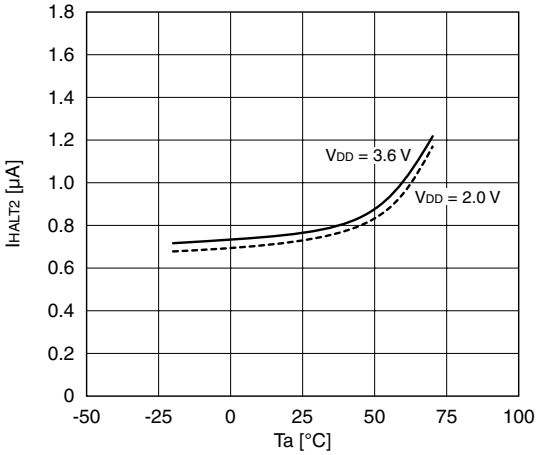
Current consumption-temperature characteristic in HALT mode (OSC1B operation)

OSC1A = OFF, OSC1B = ON, OSC3A = OFF, OSC3B = OFF, Typ. value



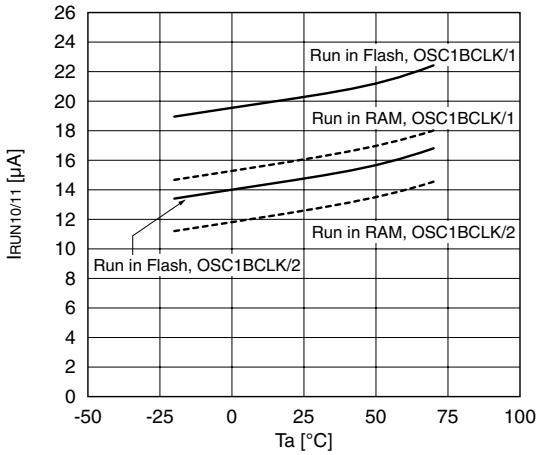
Current consumption-temperature characteristic in HALT mode (OSC1A operation)

OSC1A = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, Typ. value



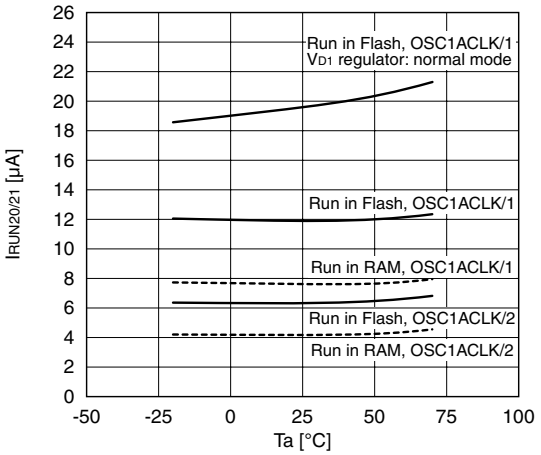
Current consumption-temperature characteristic in RUN mode (OSC1B operation)

OSC1A = OFF, OSC1B = ON, OSC3A = OFF, OSC3B = OFF, Typ. value



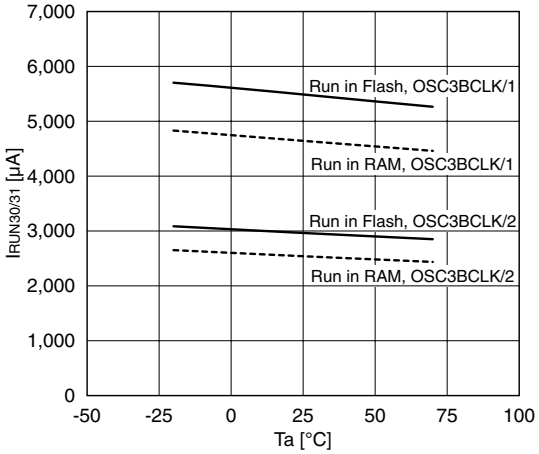
Current consumption-temperature characteristic in RUN mode (OSC1A operation)

OSC1 = 32 kHz, OSC1B = OFF, OSC3A = OFF, OSC3B = OFF, Typ. value



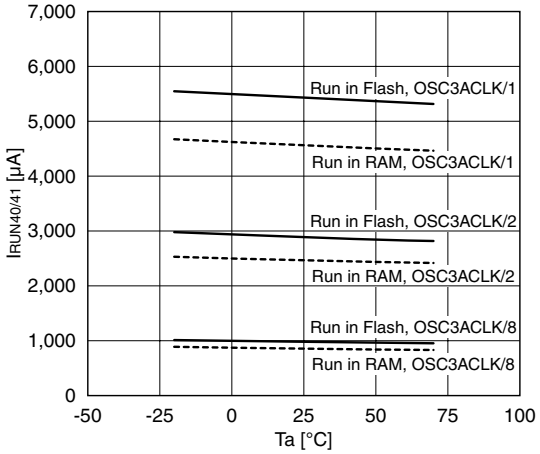
Current consumption-temperature characteristic in RUN mode (OSC3B operation)

OSC1A = OFF, OSC1B = OFF, OSC3A = OFF,
OSC3B = 20 MHz, Typ. value



Current consumption-temperature characteristic in RUN mode (OSC3A operation)

OSC1A = OFF, OSC1B = OFF, OSC3A = 20 MHz,
OSC3B = OFF, Typ. value

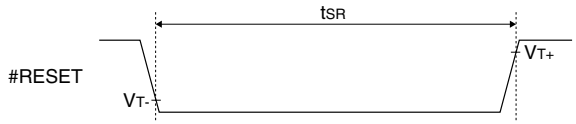


22.4 System Reset Controller (SRC) Characteristics

#RESET pin characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

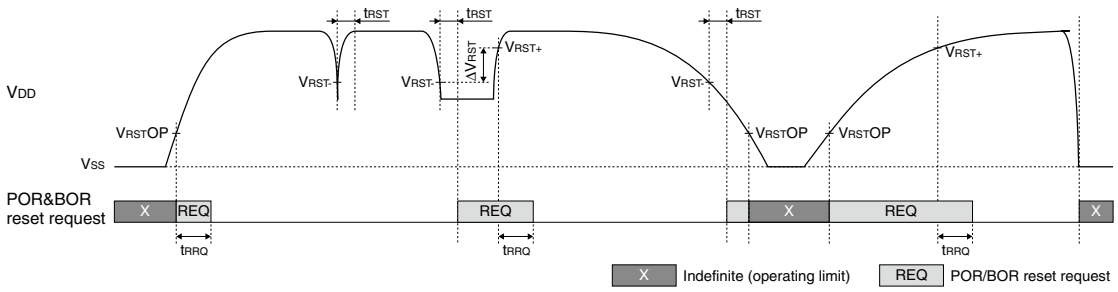
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
High level Schmitt input threshold voltage	V_{T+}		0.5 × V_{DD}	–	0.9 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}		0.1 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T		180	–	–	mV
Input pull-up resistance	R_{IN}		100	270	500	kΩ
Pin capacitance	C_{IN}		–	–	15	pF
Reset Low pulse width	t_{SR}		2	–	–	µs



POR/BOR characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
POR/BOR canceling voltage	V_{RST+}		–	1.51	1.75	V
POR/BOR detection voltage	V_{RST-}		1.25	1.45	–	V
POR/BOR hysteresis voltage	ΔV_{RST}		50	60	–	mV
POR/BOR detection response time	t_{RST}		–	–	20	µs
POR/BOR operating limit voltage	$V_{RST}OP$		–	–	0.95	V
POR/BOR reset request hold time	t_{RRQ}		0.01	–	4	ms



Note: When performing a power-on-reset again after the power is turned off, decrease the V_{DD} voltage to 1 V or less.

Reset hold circuit characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Reset hold time*1	t_{RSTR}		–	–	150	µs

*1 Time until the internal reset signal is negated after the reset request is canceled.

22.5 Clock Generator (CLG) Characteristics

Oscillator circuit characteristics including resonators change depending on conditions (board pattern, components used, etc.). Use these characteristic values as a reference and perform matching evaluation using the actual printed circuit board.

OSC1B oscillator circuit characteristics

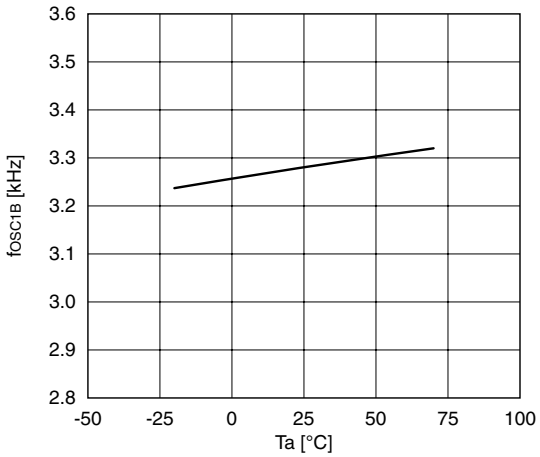
Unless otherwise specified: V_{DD} = 2.0 to 3.6V, V_{SS} = 0V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{sta}1B				200	µs
Oscillation frequency *1 *2	f_{OSC1B}		Typ. × 0.94	32.00	Typ. × 1.06	kHz
Dependence of oscillation frequency on temperature *2	Tf_{OSC1B}	Frequency accuracy per ±1°C change in temperature (with reference to 25°C)		±0.12	±0.3	%/°C

*1 In chip mounting, the value may exceed the range shown above according to the mount condition on the board.
*2 Reference value

OSC1B oscillation frequency-temperature characteristic

Typ. value



OSC1A oscillator circuit characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{sta}1A		–	–	3	s
Internal gate capacitance	C_{GI1}		–	7	–	pF
Internal drain capacitance	C_{DI1}		–	3	–	pF

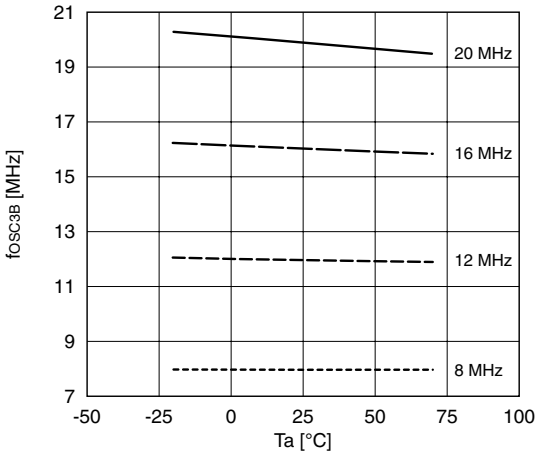
OSC3B oscillator circuit characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{sta}3B		–	–	5	µs
Oscillation frequency	f_{OSC3B}	CLGOSC3B.OSC3BFREQ[1:0] bits = 0x3	18.0	20.0	22.0	MHz
		CLGOSC3B.OSC3BFREQ[1:0] bits = 0x2	14.4	16.0	17.6	MHz
		CLGOSC3B.OSC3BFREQ[1:0] bits = 0x1	10.8	12.0	13.2	MHz
		CLGOSC3B.OSC3BFREQ[1:0] bits = 0x0	7.2	8.0	8.8	MHz

OSC3B oscillation frequency-temperature characteristic

Typ. value



OSC3A oscillator circuit characteristics

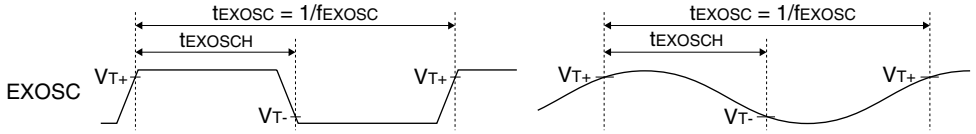
Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{sta}3A	Crystal resonator	–	–	20	ms
		Ceramic resonator	–	–	1	ms
Internal gate capacitance	C_{GI3}		–	5	–	pF
Internal drain capacitance	C_{DI3}		–	5	–	pF

EXOSC external clock input characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
EXOSC external clock duty ratio	t_{EXOSCD}	$t_{EXOSCD} = t_{EXOSCH}/t_{EXOSC}$	46	–	54	%
High level Schmitt input threshold voltage	V_{T+}		$0.5 \times V_{DD}$	–	$0.9 \times V_{DD}$	V
Low level Schmitt input threshold voltage	V_{T-}		$0.1 \times V_{DD}$	–	$0.5 \times V_{DD}$	V
Schmitt input hysteresis voltage	ΔV_T		180	–	–	mV



22.6 Flash Memory Characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

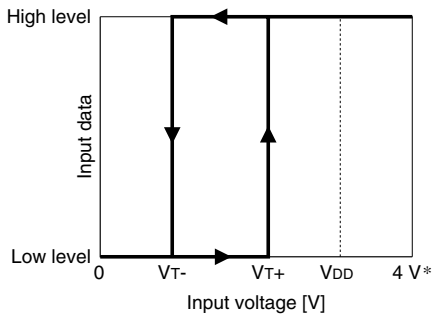
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Programming count *1	C_{FEP}	Programmed data is guaranteed to be retained for 10 years.	50	–	–	times

*1 Assumed that Erasing + Programming as count of 1. The count includes programming in the factory for shipment with ROM data programmed.

22.7 Input/Output Port (PPORT) Characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

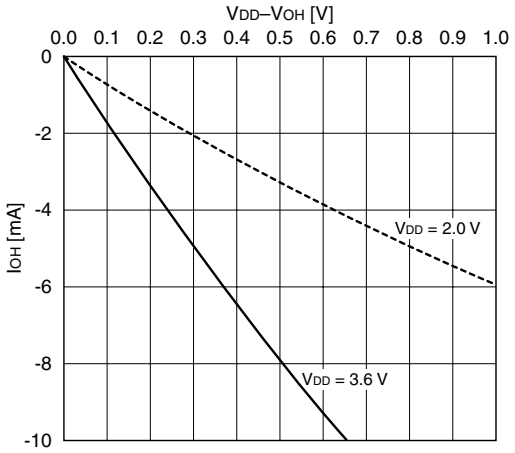
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
High level Schmitt input threshold voltage	V_{T+}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D1	0.5 × V_{DD}	–	0.9 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D1	0.1 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D1	180	–	–	mV
High level output current	I_{OH}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D2, V_{OH} = 0.9 × V_{DD}	–	–	-0.5	mA
Low level output current	I_{OL}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D2, V_{OL} = 0.1 × V_{DD}	0.5	–	–	mA
Leakage current	I_{LEAK}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D1	-150	–	150	nA
Input pull-up resistance	R_{INU}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D1	75	150	300	kΩ
Input pull-down resistance	R_{IND}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D1	75	150	300	kΩ
Pin capacitance	C_{IN}	P00–07, P10–17, P20–27, P30–37, P40–41, PD0–D2	–	–	15	pF



(* For over voltage tolerant fail-safe type port)

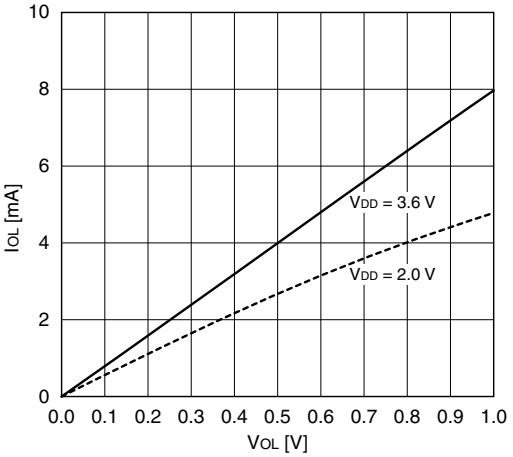
High-level output current characteristic

Ta = 70 °C, Max. value



Low-level output current characteristic

Ta = 70 °C, Min. value

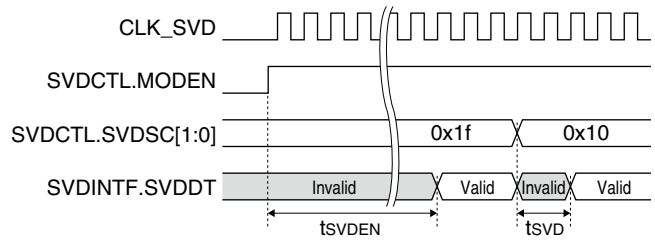


22.8 Supply Voltage Detector (SVD) Characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

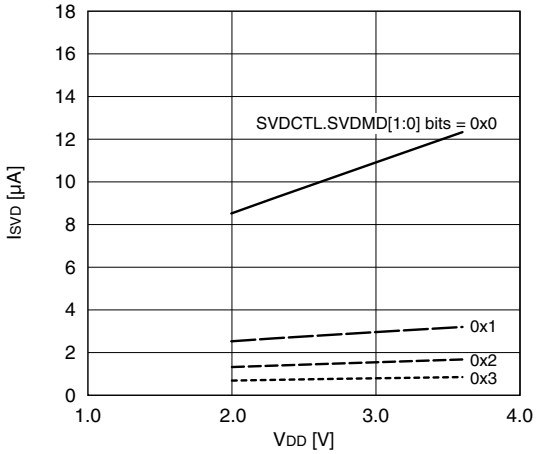
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
EXSVD pin input voltage range	V_{EXSVD}		0	–	V_{DD}	V
EXSVD input impedance	R_{EXSVD}	SVDCTL.SVDC[4:0] bits = 0x0c	309	442	575	kΩ
		SVDCTL.SVDC[4:0] bits = 0x0d	327	467	607	kΩ
		SVDCTL.SVDC[4:0] bits = 0x0e	344	492	640	kΩ
		SVDCTL.SVDC[4:0] bits = 0x0f	362	517	672	kΩ
		SVDCTL.SVDC[4:0] bits = 0x10	379	542	705	kΩ
		SVDCTL.SVDC[4:0] bits = 0x11	397	567	737	kΩ
		SVDCTL.SVDC[4:0] bits = 0x12	414	592	770	kΩ
		SVDCTL.SVDC[4:0] bits = 0x13	432	617	802	kΩ
		SVDCTL.SVDC[4:0] bits = 0x14	449	642	835	kΩ
		SVDCTL.SVDC[4:0] bits = 0x15	467	667	867	kΩ
		SVDCTL.SVDC[4:0] bits = 0x16	484	692	900	kΩ
		SVDCTL.SVDC[4:0] bits = 0x17	502	717	932	kΩ
		SVDCTL.SVDC[4:0] bits = 0x18	519	742	965	kΩ
		SVDCTL.SVDC[4:0] bits = 0x19	537	767	997	kΩ
		SVDCTL.SVDC[4:0] bits = 0x1a	554	792	1,030	kΩ
		SVDCTL.SVDC[4:0] bits = 0x1b	572	817	1,062	kΩ
		SVDCTL.SVDC[4:0] bits = 0x1c	589	842	1,095	kΩ
		SVDCTL.SVDC[4:0] bits = 0x1d	607	867	1,127	kΩ
		SVDCTL.SVDC[4:0] bits = 0x1e	624	892	1,160	kΩ
SVD detection voltage	V_{SVD}	SVDCTL.SVDC[4:0] bits = 0x0c	1.728	1.800	1.872	V
		SVDCTL.SVDC[4:0] bits = 0x0d	1.824	1.900	1.976	V
		SVDCTL.SVDC[4:0] bits = 0x0e	1.920	2.000	2.080	V
		SVDCTL.SVDC[4:0] bits = 0x0f	2.016	2.100	2.184	V
		SVDCTL.SVDC[4:0] bits = 0x10	2.112	2.200	2.288	V
		SVDCTL.SVDC[4:0] bits = 0x11	2.208	2.300	2.392	V
		SVDCTL.SVDC[4:0] bits = 0x12	2.304	2.400	2.496	V
		SVDCTL.SVDC[4:0] bits = 0x13	2.400	2.500	2.600	V
		SVDCTL.SVDC[4:0] bits = 0x14	2.496	2.600	2.704	V
		SVDCTL.SVDC[4:0] bits = 0x15	2.592	2.700	2.808	V
		SVDCTL.SVDC[4:0] bits = 0x16	2.688	2.800	2.912	V
		SVDCTL.SVDC[4:0] bits = 0x17	2.784	2.900	3.016	V
		SVDCTL.SVDC[4:0] bits = 0x18	2.880	3.000	3.120	V
		SVDCTL.SVDC[4:0] bits = 0x19	2.976	3.100	3.224	V
		SVDCTL.SVDC[4:0] bits = 0x1a	3.072	3.200	3.328	V
		SVDCTL.SVDC[4:0] bits = 0x1b	3.168	3.300	3.432	V
		SVDCTL.SVDC[4:0] bits = 0x1c	3.264	3.400	3.536	V
		SVDCTL.SVDC[4:0] bits = 0x1d	3.360	3.500	3.640	V
		SVDCTL.SVDC[4:0] bits = 0x1e	3.456	3.600	3.744	V
SVD circuit enable response time	t_{SVDEN}	*1	–	–	500	µs
SVD circuit response time	t_{SVD}		–	–	60	µs
SVD circuit current	I_{SVD}	SVDCTL.SVDMD[1:0] bits = 0x0, SVDCTL.SVDC[4:0] bits = 0x0c, CLK_SVD = 32 kHz, Ta = 25 °C	–	17	30	µA
		SVDCTL.SVDMD[1:0] bits = 0x1, SVDCTL.SVDC[4:0] bits = 0x0c, CLK_SVD = 32 kHz, Ta = 25 °C	–	4	6	µA
		SVDCTL.SVDMD[1:0] bits = 0x2, SVDCTL.SVDC[4:0] bits = 0x0c, CLK_SVD = 32 kHz, Ta = 25 °C	–	2	4	µA
		SVDCTL.SVDMD[1:0] bits = 0x3, SVDCTL.SVDC[4:0] bits = 0x0c, CLK_SVD = 32 kHz, Ta = 25 °C	–	1	3	µA

*1 If CLK_SVD is configured in the neighborhood of 32 kHz, the SVDINTF.SVDDT bit is masked during the t_{SVDEN} period and it retains the previous value.



SVD circuit current - power supply voltage characteristic

Ta = 25 °C, SVDCTL.SVDC[4:0] bits = 0x0c, CLK_SVD = 32 kHz, Typ. value



22.9 UART (UART) Characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Transfer baud rate	U_{BRT1}	Normal mode	150	–	460,800	bps
	U_{BRT2}	IrDA mode	150	–	115,200	bps

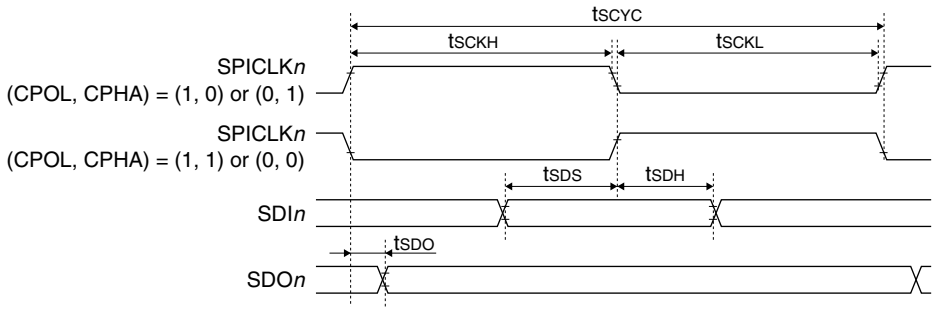
22.10 Synchronous Serial Interface (SPIA) Characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

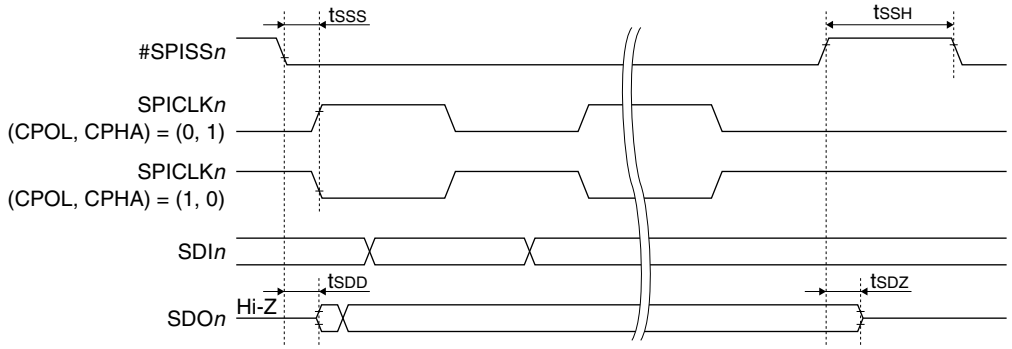
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
SPICLK*n* cycle time	t_{SCYC}		500	–	–	ns
SPICLK*n* High pulse width	t_{SCKH}		200	–	–	ns
SPICLK*n* Low pulse width	t_{SCKL}		200	–	–	ns
SDI*n* setup time	t_{SDS}		70	–	–	ns
SDI*n* hold time	t_{SDH}		10	–	–	ns
SDO*n* output delay time	t_{SDO}	C_L = 30 pF *1	–	–	100	ns
#SPISS*n* setup time	t_{SSS}		70	–	–	ns
#SPISS*n* High pulse width	t_{SSH}		80	–	–	ns
SDO*n* output start time	t_{SDD}	C_L = 30 pF *1	–	–	100	ns
SDO*n* output stop time	t_{SDZ}	C_L = 30 pF *1	–	–	80	ns

*1 C_L = Pin load

Common to master and slave modes



Slave mode

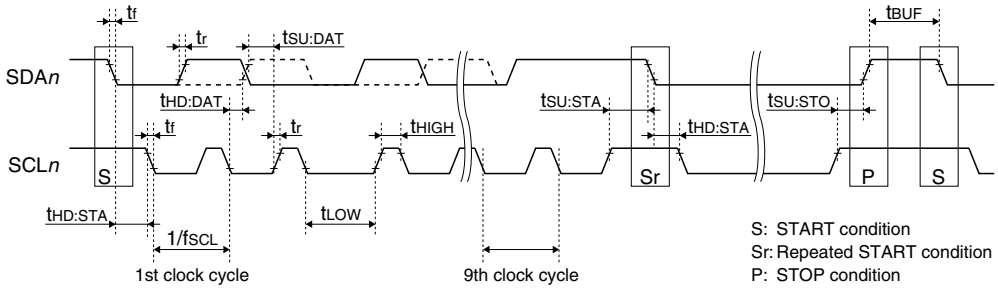


22.11 I²C (I2C) Characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Standard mode			Fast mode			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
SCL*n* frequency	f_{SCL}		0	–	100	0	–	400	kHz
Hold time (repeated) START condition *	$t_{HD:STA}$		4.0	–	–	0.6	–	–	µs
SCL*n* Low pulse width	t_{LOW}		4.7	–	–	1.3	–	–	µs
SCL*n* High pulse width	t_{HIGH}		4.0	–	–	0.6	–	–	µs
Repeated START condition setup time	$t_{SU:STA}$		4.7	–	–	0.6	–	–	µs
Data hold time	$t_{HD:DAT}$		0	–	–	0	–	–	µs
Data setup time	$t_{SU:DAT}$		250	–	–	100	–	–	ns
SDA*n*, SCL*n* rise time	t_r		–	–	1,000	–	–	300	ns
SDA*n*, SCL*n* fall time	t_f		–	–	300	–	–	300	ns
STOP condition setup time	$t_{SU:STO}$		4.0	–	–	0.6	–	–	µs
Bus free time	t_{BUF}		4.7	–	–	1.3	–	–	µs

* After this period, the first clock pulse is generated.

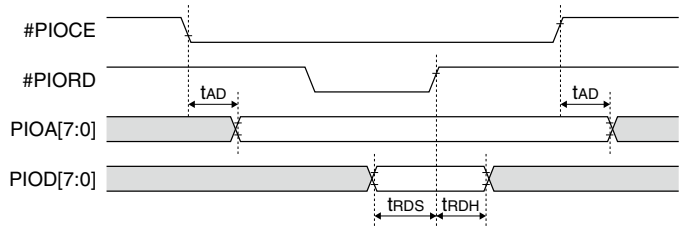


22.12 Parallel Interface (PIO) Characteristics

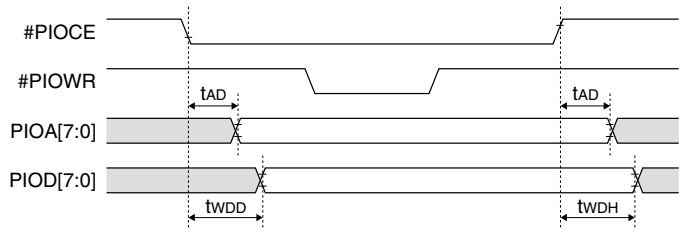
Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Address output delay time	t_{AD}		–	–	0	ns
Write data output delay time	t_{WDD}		–	–	10	ns
Write data hold time	t_{WDH}		-20	–	–	ns
Read data setup time	t_{RDS}		30	–	–	ns
Read data hold time	t_{RDH}		0	–	–	ns

ead timing



Write timing



22.13 EPD Timing Controller (EPD Tcon) Characteristics

Refer to the EPD Timing Controller S1C17F13 Manual (separately attached sheet).

22.14 R/F Converter (RFC) Characteristics

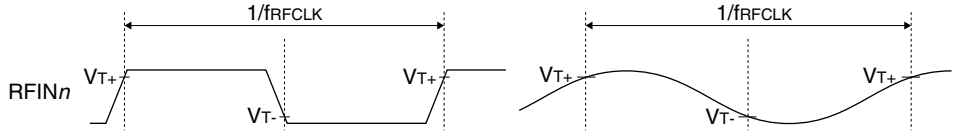
R/F converter characteristics change depending on conditions (board pattern, components used, etc.). Use these characteristic values as a reference and perform evaluation using the actual printed circuit board.

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, -20 to 70 °C

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
Reference/sensor oscillation frequency	f_{RFCLK}			1	–	4,000	kHz
Reference/sensor oscillation frequency IC deviation	$\Delta f_{RFCLK}/\Delta IC$	Ta = 25 °C ∗1	V_{DD} = 2.0 V	-30	–	30	%
			V_{DD} = 3.6 V	-40	–	40	%
Reference resistor/resistive sensor resistance	R_{REF}, R_{SEN}			10	–	–	kΩ
Reference capacitance	C_{REF}			100	–	–	pF
Time base counter clock frequency	f_{TCCLK}			–	–	8.2	MHz
High level Schmitt input threshold voltage	V_{T+}			0.5 × V_{DD}	–	0.9 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}			0.1 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T			180	–	–	mV
R/F converter operating current	I_{RFC}	C_{REF} = 1000 pF, R_{REF}/R_{SEN} = 100 kΩ, Ta = 25 °C	DC oscillation mode	–	0.9	1.5	mA
			AC oscillation mode	–	2	3.5	mA

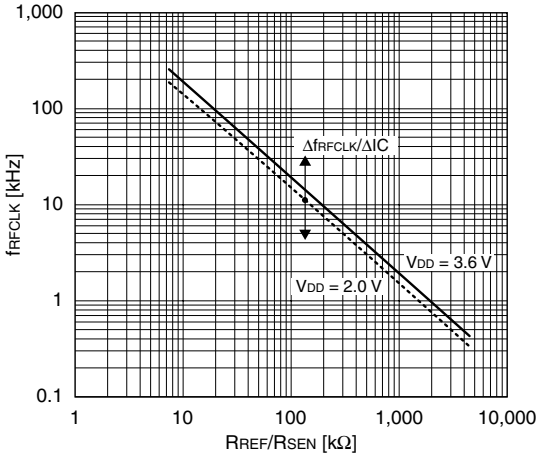
∗1 In this characteristic, unevenness between production lots, and variations in measurement board, resistances and capacitances are taken into account.

Waveforms for external clock input mode



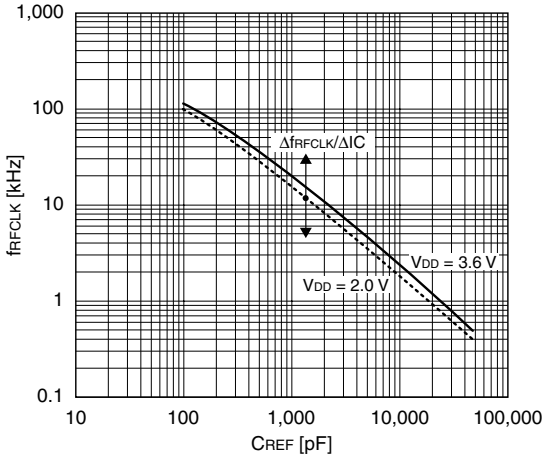
RFC reference/sensor oscillation frequency-resistance characteristic

C_{REF} = 1,000 pF, Ta = 25 °C, Typ. value



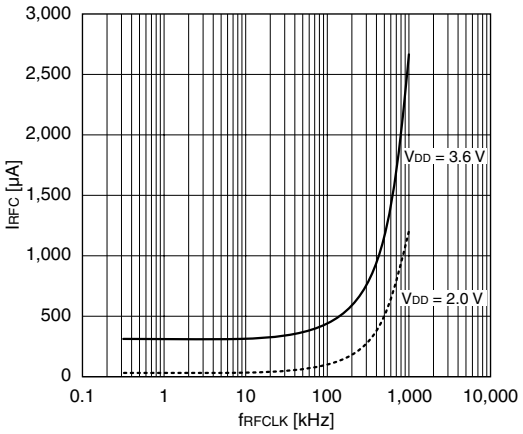
RFC reference/sensor oscillation frequency-capacitance characteristic

R_{REF}/R_{SEN} = 100 kΩ, Ta = 25 °C, Typ. value



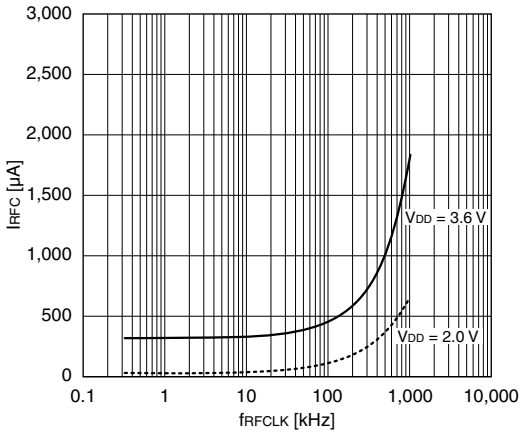
RFC reference/sensor oscillation current consumption-frequency characteristic (DC oscillation mode)

C_{REF} = 1,000 pF, Ta = 25 °C, Typ. value



RFC reference/sensor oscillation current consumption-frequency characteristic (AC oscillation mode)

C_{REF} = 1,000 pF, Ta = 25 °C, Typ. value



22.15 Temperature Detection Circuit Characteristics

Analog characteristics

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = -20 to 70 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Resolution	–			8		bit
Temperature sensor output voltage stabilization time	t_{STAB}				10	ms
Temperature sensor output voltage comparison time	t_{CMP}		150			µs
Temperature detection range	T_{RNG}		-40		85	°C
Temperature detection error	E_{TEM}	T_{RNG} = 0 to 50 °C		±2	±5	°C

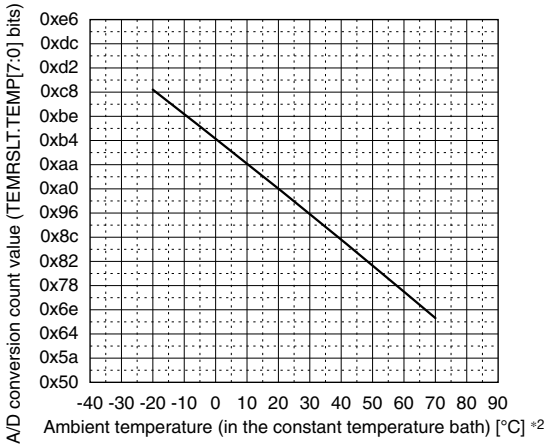
Temperature detection circuit current consumption

Unless otherwise specified: V_{DD} = 2.0 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Temperature detection circuit operating current *1	I_{TEM}			6	12	µA

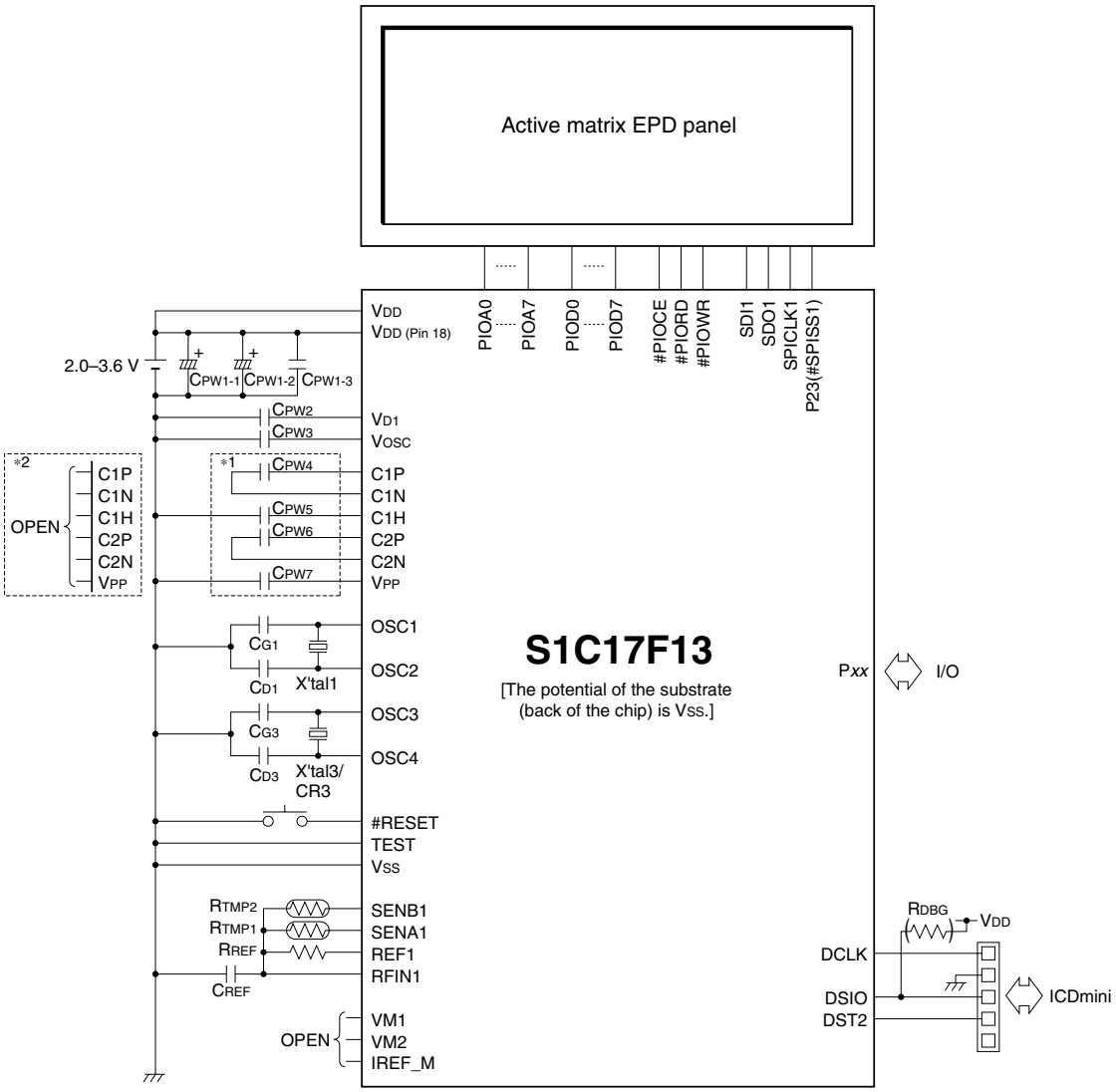
*1 This value is added to the current consumption during HALT/execution (with or without heavy load protection mode) when the temperature detection circuit is active.

Temperature detection circuit conversion value -ambient temperature characteristic



*2 This characteristic was measured in 150 µs of the conversion time (comparison time) after canceling SLEEP status by placing the device in the constant temperature bath.
The temperature detection circuit measures temperature inside the device. Depending on the use environment, the difference between the measured temperature and the ambient temperature may be increased. When using the measured temperature as the ambient temperature, prepare an appropriate conversion table according to the use environment.

23 Basic External Connection Diagram



*1 When the Flash voltage booster is used
*2 When the Flash voltage booster is not used

Sample external parts

Symbol	Name	Recommended parts
C_{PW1-1}	Bypass capacitor between V_{DD}–V_{SS}	Electrolytic capacitor
C_{PW1-2}	Capacitor between V_{DD}–V_{SS}	Electrolytic capacitor (Place as closer to V_{DD} (pin 18) as possible.)
C_{PW1-3}	Capacitor between V_{DD}–V_{SS}	Ceramic capacitor (Place as closer to V_{DD} (pin 18) as possible.)
C_{PW2}	V_{D1} stabilization capacitor	Ceramic capacitor
C_{PW3}	V_{OSC} stabilization capacitor	Ceramic capacitor
$C_{PW4–6}$	Flash power supply voltage boosting capacitors	Ceramic capacitor
C_{PW7}	V_{PP} stabilization capacitor	Ceramic capacitor
R_{DBG}	DSIO pull-up resistor	Thick film chip resistor
R_{REF}	RFC reference resistor	Thick film chip resistor
$R_{TMP1, 2}$	Resistive sensors	Temperature sensor 103AP-2 manufactured by SEMITEC Corporation
		Humidity sensor C15-M53R manufactured by SHINYEI Technology Co.,Ltd. (* In AC oscillation mode for resistive sensor measurements)
C_{REF}	RFC oscillating capacitor	Ceramic capacitor

* For recommended component values, refer to “Recommended Operating Conditions” in the “Electrical Characteristics” chapter. The C_{G1}, C_{D1}, C_{G3}, and C_{D3} capacitance values should be determined after performing matching evaluation of the resonator mounted on the printed circuit board actually used.

24 Package

TQFP13-64pin package

(Unit: mm)

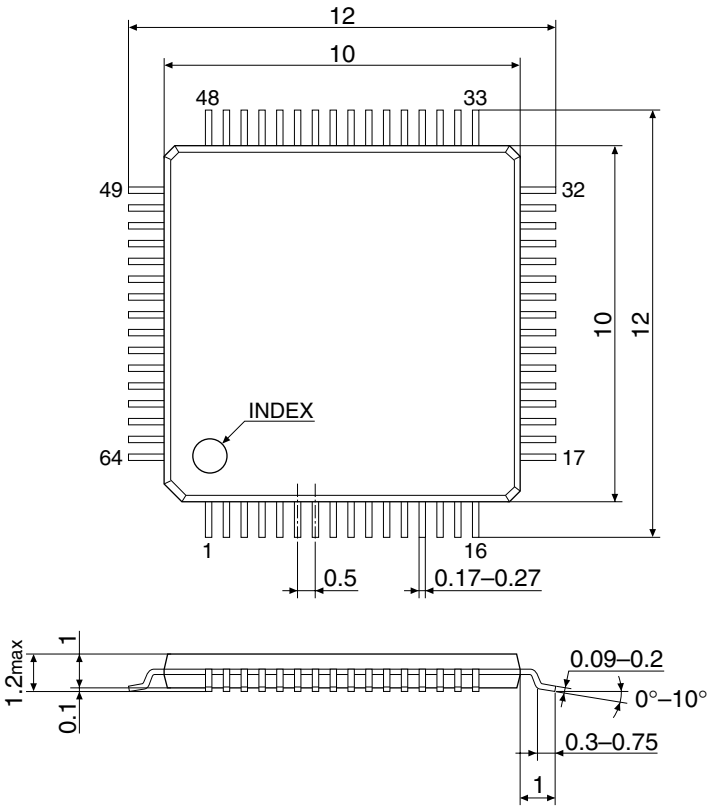


Figure 24.1 TQFP13-64pin Package Dimensions

Appendix A List of Peripheral Circuit Control Registers

0x4000–0x4008 Misc Registers (MISC)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4000	MSCPROT (MISC System Protect Register)	15–0	PROT[15:0]	0x0000	H0	R/W	–
0x4002	MSCIRAMSZ (MISC IRAM Size Register)	15–9	–	0x00	–	R	–
		8	(reserved)	0	H0	R/WP	Always set to 0.
		7	–	0	–	R	–
		6–4	(reserved)	0x4	–	R	
		3	–	0	–	R	
		2–0	IRAMSZ[2:0]	0x4	H0	R/WP	
0x4004	MSCTTBRL (MISC Vector Table Address Low Register)	15–8	TTBR[15:8]	0x80	H0	R/WP	–
		7–0	TTBR[7:0]	0x00	H0	R	
0x4006	MSCTTBRH (MISC Vector Table Address High Register)	15–8	–	0x00	–	R	–
		7–0	TTBR[23:16]	0x00	H0	R/WP	
0x4008	MSCPSR (MISC PSR Register)	15–8	–	0x00	–	R	–
		7–5	PSRIL[2:0]	0x0	H0	R	
		4	PSRIE	0	H0	R	
		3	PSRC	0	H0	R	
		2	PSRV	0	H0	R	
		1	PSRZ	0	H0	R	
		0	PSRN	0	H0	R	

0x4020 Power Generator (PWG)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4020	PWGVD1CTL (PWG V_{D1} Regulator Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1–0	REGMODE[1:0]	0x0	H0	R/WP	

0x4040–0x404e Clock Generator (CLG)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4040	CLGSCLK (CLG System Clock Control Register)	15	WUPMD	0	H0	R/WP	–
		14	–	0	–	R	
		13–12	WUPDIV[1:0]	0x0	H0	R/WP	
		11–10	–	0x0	–	R	
		9–8	WUPSRC[1:0]	0x0	H0	R/WP	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/WP	
0x4042	CLGOSC (CLG Oscillation Control Register)	15–12	–	0x0	–	R	–
		11	EXOSCSLPC	1	H0	R/W	
		10	OSC3ASLPC	1	H0	R/W	
		9	OSC1SLPC	1	H0	R/W	
		8	OSC3BSLPC	1	H0	R/W	
		7–4	–	0x0	–	R	
		3	EXOSCEN	0	H0	R/W	
		2	OSC3AEN	0	H0	R/W	
		1	OSC1EN	0	H0	R/W	
		0	OSC3BEN	1	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4044	CLGOSC3B (CLG OSC3B Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1–0	OSC3BFREQ[1:0]	0x0	H0	R/WP	
0x4046	CLGOSC1 (CLG OSC1 Control Register)	15–8	–	0x1a	–	R	–
		7–4	–	0xc	–	R	
		3	–	0	–	R	
		2	OSC1SEL	1	H0	R/WP	
		1–0	OSC1WT[1:0]	0x2	H0	R/WP	
0x4048	CLGOSC3A (CLG OSC3A Control Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5–4	INVN[1:0]	0x1	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	OSC3AWT[1:0]	0x2	H0	R/WP	
0x404a	CLGINTF (CLG Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	OSC3ASTAIF	0	H0	R/W	Cleared by writing 1.
		1	OSC1STAIF	0	H0	R/W	
		0	OSC3BSTAIF	0	H0	R/W	
0x404c	CLGINTE (CLG Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	OSC3ASTAIE	0	H0	R/W	
		1	OSC1STAIE	0	H0	R/W	
		0	OSC3BSTAIE	0	H0	R/W	
0x404e	CLGFOUT (CLG FOUT Control Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–4	FOUTDIV[2:0]	0x0	H0	R/W	
		3–2	FOUTSRC[1:0]	0x0	H0	R/W	
		1	–	0	–	R	
		0	FOUTEN	0	H0	R/W	

0x4052 Theoretical Regulation (TR)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4052	TRCTL (Theoretical Regulation Control Register)	15–10	–	0x00	–	R	–
		9	REGFREQ	0	H0	R/W	
		8	REGMONEN	0	H0	R/W	
		7	REGTRIG	0	H0	W	Always read as 0.
		6	–	0	–	R	–
		5–0	TRIM[5:0]	0x00	H0	R/W	

0x4080–0x4092 Interrupt Controller (ITC)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4080	ITCLV0 (ITC Interrupt Level Setup Register 0)	15–11	–	0x00	–	R	–
		10–8	ILV1[2:0]	0x0	H0	R/W	Port interrupt (ILVPPORT)
		7–3	–	0x00	–	R	–
		2–0	ILV0[2:0]	0x0	H0	R/W	Supply voltage detector interrupt (ILVSVD)
0x4082	ITCLV1 (ITC Interrupt Level Setup Register 1)	15–11	–	0x00	–	R	–
		10–8	ILV3[2:0]	0x0	H0	R/W	Real-time clock interrupt (ILVRTC)
		7–3	–	0x00	–	R	–
		2–0	ILV2[2:0]	0x0	H0	R/W	Clock generator interrupt (ILVCLG)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4084	ITCLV2 (ITC Interrupt Level Setup Register 2)	15–11	–	0x00	–	R	–
		10–8	ILV5[2:0]	0x0	H0	R/W	UART interrupt (ILVUART_0)
		7–3	–	0x00	–	R	–
		2–0	ILV4[2:0]	0x0	H0	R/W	16-bit timer Ch.0 interrupt (ILVT16_0)
0x4086	ITCLV3 (ITC Interrupt Level Setup Register 3)	15–11	–	0x00	–	R	–
		10–8	ILV7[2:0]	0x0	H0	R/W	SPI Ch.0 interrupt (ILVSPI_0)
		7–3	–	0x00	–	R	–
		2–0	ILV6[2:0]	0x0	H0	R/W	16-bit timer Ch.1 interrupt (ILVT16_1)
0x4088	ITCLV4 (ITC Interrupt Level Setup Register 4)	15–11	–	0x00	–	R	–
		10–8	ILV9[2:0]	0x0	H0	R/W	Clock timer interrupt (ILVCT)
		7–3	–	0x00	–	R	–
		2–0	ILV8[2:0]	0x0	H0	R/W	I²C interrupt (ILVI2C_0)
0x408a	ITCLV5 (ITC Interrupt Level Setup Register 5)	15–11	–	0x00	–	R	–
		10–8	ILV11[2:0]	0x0	H0	R/W	SPI Ch.1 interrupt (ILVSPI_1)
		7–3	–	0x00	–	R	–
		2–0	ILV10[2:0]	0x0	H0	R/W	16-bit timer Ch.2 interrupt (ILVT16_2)
0x408c	ITCLV6 (ITC Interrupt Level Setup Register 6)	15–11	–	0x00	–	R	–
		10–8	ILV13[2:0]	0x0	H0	R/W	SPI Ch.2 interrupt (ILVSPI_2)
		7–3	–	0x00	–	R	–
		2–0	ILV12[2:0]	0x0	H0	R/W	16-bit timer Ch.3 interrupt (ILVT16_3)
0x408e	ITCLV7 (ITC Interrupt Level Setup Register 7)	15–11	–	0x00	–	R	–
		10–8	ILV15[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.1 interrupt (ILVT16A3_1)
		7–3	–	0x00	–	R	–
		2–0	ILV14[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.0 interrupt (ILVT16A3_0)
0x4090	ITCLV8 (ITC Interrupt Level Setup Register 8)	15–11	–	0x00	–	R	–
		10–8	ILV17[2:0]	0x0	H0	R/W	R/F converter Ch.1 interrupt (ILVRFC_1)
		7–3	–	0x00	–	R	–
		2–0	ILV16[2:0]	0x0	H0	R/W	R/F converter Ch.0 interrupt (ILVRFC_0)
0x4092	ITCLV9 (ITC Interrupt Level Setup Register 9)	15–11	–	0x00	–	R	–
		10–8	ILV19[2:0]	0x0	H0	R/W	Temperature detection circuit interrupt (ILVTEM)
		7–3	–	0x00	–	R	–
		2–0	ILV18[2:0]	0x0	H0	R/W	EPD timing controller interrupt (ILVEPD_Tcon)

0x40a0–0x40a2 Watchdog Timer (WDT)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x40a0	WDTCLK (WDT Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/WP	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/WP	
0x40a2	WDTCTL (WDT Control Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	WDTCNTRST	0	H0	WP	Always read as 0.
		3–0	WDTRUN[3:0]	0xa	H0	R/WP	–

0x40c0–0x40c8 Real-time Clock (RTC)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x40c0	RTCCTL (RTC Control Register)	15–9	–	0x00	–	R	–
		8	RTCST	0	H0	R	
		7–6	–	0x0	–	R	
		5	BCDMD	0	H0	R/W	
		4	RTC24H	0	H0	R/W	
		3–1	–	0x0	–	R	
		0	RTCRUN	0	H0	R/W	
0x40c2	RTCINTE (RTC Interrupt Enable Register)	15–10	–	0x00	–	R	–
		9	1DIE	0	H0	R/W	
		8	HDIE	0	H0	R/W	
		7	1HIE	0	H0	R/W	
		6	10MIE	0	H0	R/W	
		5	1MIE	0	H0	R/W	
		4	10SIE	0	H0	R/W	
		3	1HZIE	0	H0	R/W	
		2	4HZIE	0	H0	R/W	
		1	8HZIE	0	H0	R/W	
		0	32HZIE	0	H0	R/W	
0x40c4	RTCINTF (RTC Interrupt Flag Register)	15–10	–	0x00	–	R	–
		9	1DIF	0	H0	R/W	Cleared by writing 1.
		8	HDIF	0	H0	R/W	
		7	1HIF	0	H0	R/W	
		6	10MIF	0	H0	R/W	
		5	1MIF	0	H0	R/W	
		4	10SIF	0	H0	R/W	
		3	1HZIF	0	H0	R/W	
		2	4HZIF	0	H0	R/W	
		1	8HZIF	0	H0	R/W	
		0	32HZIF	0	H0	R/W	
0x40c6	RTCMIN (RTC Minute/Second Register)	15	–	0	–	R	–
		14–8	RTCMIN[6:0]	x	–	R/W	
		7	–	0	–	R	
		6–0	RTCSEC[6:0]	x	–	R/W	
0x40c8	RTCHUR (RTC Hour Register)	15–8	–	0x00	–	R	–
		7	AMPM	x	–	R/W	
		6	–	0	–	R	
		5–0	RTCHUR[5:0]	x	–	R/W	

0x4100–0x4106 Supply Voltage Detector (SVD)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4100	SVDCLK (SVD Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	1	H0	R/WP	
		7	–	0	–	R	
		6–4	CLKDIV[2:0]	0x0	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/WP	
0x4102	SVDCTL (SVD Control Register)	15	VDSEL	0	H1	R/WP	–
		14–13	SVDSC[1:0]	0x0	H0	R/WP	Writing takes effect when the SVDCTL.SVDMD[1:0] bits are not 0x0.
		12–8	SVDC[4:0]	0x00	H1	R/WP	–
		7–4	SVDRE[3:0]	0x0	H1	R/WP	
		3	–	0	–	R	
		2–1	SVDMD[1:0]	0x0	H0	R/WP	
		0	MODEN	0	H1	R/WP	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4104	SVDINTF (SVD Status and Interrupt Flag Register)	15–9	–	0x00	–	R	–
		8	SVDDT	x	–	R	
		7–1	–	0x00	–	R	
		0	SVDIF	0	H1	R/W	Cleared by writing 1.
0x4106	SVDINTE (SVD Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	SVDIE	0	H0	R/W	

0x4160–0x416c 16-bit Timer (T16) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4160	T16_0CLK (T16 Ch.0 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x4162	T16_0MOD (T16 Ch.0 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x4164	T16_0CTL (T16 Ch.0 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7–2	–	0x00	–	R	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x4166	T16_0TR (T16 Ch.0 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x4168	T16_0TC (T16 Ch.0 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x416a	T16_0INTF (T16 Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x416c	T16_0INTE (T16 Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x41b0 Flash Controller (FLASHC)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x41b0	FLASHCWAIT (FLASHC Flash Read Cycle Register)	15–8	–	0x00	–	R	–
		7	XBUSY	0	H0	R	
		6–2	–	0x00	–	R	
		1–0	RDWAIT[1:0]	0x0	H0	R/WP	

0x4200–0x42e2 I/O Ports (PPORT)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4200	P0DAT (P0 Port Data Register)	15–8	P0OUT[7:0]	0x00	H0	R/W	–
		7–0	P0IN[7:0]	0x00	H0	R	
0x4202	P0IOEN (P0 Port Enable Register)	15–8	P0IEN[7:0]	0x00	H0	R/W	–
		7–0	P0OEN[7:0]	0x00	H0	R/W	
0x4204	P0RCTL (P0 Port Pull-up/down Control Register)	15–8	P0PDPU[7:0]	0x00	H0	R/W	–
		7–0	P0REN[7:0]	0x00	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4206	P0INTF (P0 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–0	P0IF[7:0]	0x00	H0	R/W	Cleared by writing 1.
0x4208	P0INTCTL (P0 Port Interrupt Control Register)	15–8	P0EDGE[7:0]	0x00	H0	R/W	–
		7–0	P0IE[7:0]	0x00	H0	R/W	
0x420a	P0CHATEN (P0 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–
		7–0	P0CHATEN[7:0]	0x00	H0	R/W	
0x420c	P0MODSEL (P0 Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–0	P0SEL[7:0]	0x00	H0	R/W	
0x420e	P0FNCSEL (P0 Port Function Select Register)	15–14	P07MUX[1:0]	0x0	H0	R/W	–
		13–12	P06MUX[1:0]	0x0	H0	R/W	
		11–10	P05MUX[1:0]	0x0	H0	R/W	
		9–8	P04MUX[1:0]	0x0	H0	R/W	
		7–6	P03MUX[1:0]	0x0	H0	R/W	
		5–4	P02MUX[1:0]	0x0	H0	R/W	
		3–2	P01MUX[1:0]	0x0	H0	R/W	
		1–0	P00MUX[1:0]	0x0	H0	R/W	
0x4210	P1DAT (P1 Port Data Register)	15–8	P1OUT[7:0]	0x00	H0	R/W	–
		7–0	P1IN[7:0]	x	H0	R	
0x4212	P1IOEN (P1 Port Enable Register)	15–8	P1IEN[7:0]	0x00	H0	R/W	–
		7–0	P1OEN[7:0]	0x00	H0	R/W	
0x4214	P1RCTL (P1 Port Pull-up/down Control Register)	15–8	P1PDPU[7:0]	0x00	H0	R/W	–
		7–0	P1REN[7:0]	0x00	H0	R/W	
0x4216	P1INTF (P1 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–0	P1IF[7:0]	0x00	H0	R/W	Cleared by writing 1.
0x4218	P1INTCTL (P1 Port Interrupt Control Register)	15–8	P1EDGE[7:0]	0x00	H0	R/W	–
		7–0	P1IE[7:0]	0x00	H0	R/W	
0x421a	P1CHATEN (P1 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–
		7–0	P1CHATEN[7:0]	0x00	H0	R/W	
0x421c	P1MODSEL (P1 Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–0	P1SEL[7:0]	0x00	H0	R/W	
0x421e	P1FNCSEL (P1 Port Function Select Register)	15–14	P17MUX[1:0]	0x0	H0	R/W	–
		13–12	P16MUX[1:0]	0x0	H0	R/W	
		11–10	P15MUX[1:0]	0x0	H0	R/W	
		9–8	P14MUX[1:0]	0x0	H0	R/W	
		7–6	P13MUX[1:0]	0x0	H0	R/W	
		5–4	P12MUX[1:0]	0x0	H0	R/W	
		3–2	P11MUX[1:0]	0x0	H0	R/W	
		1–0	P10MUX[1:0]	0x0	H0	R/W	
0x4220	P2DAT (P2 Port Data Register)	15–8	P2OUT[7:0]	0x00	H0	R/W	–
		7–0	P2IN[7:0]	x	H0	R	
0x4222	P2IOEN (P2 Port Enable Register)	15–8	P2IEN[7:0]	0x00	H0	R/W	–
		7–0	P2OEN[7:0]	0x00	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4224	P2RCTL (P2 Port Pull-up/down Control Register)	15–8	P2PDPU[7:0]	0x00	H0	R/W	–
		7–0	P2REN[7:0]	0x00	H0	R/W	
0x422c	P2MODSEL (P2 Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–0	P2SEL[7:0]	0x00	H0	R/W	
0x422e	P2FNCSEL (P2 Port Function Select Register)	15–14	P27MUX[1:0]	0x0	H0	R/W	–
		13–12	P26MUX[1:0]	0x0	H0	R/W	
		11–10	P25MUX[1:0]	0x0	H0	R/W	
		9–8	P24MUX[1:0]	0x0	H0	R/W	
		7–6	P23MUX[1:0]	0x0	H0	R/W	
		5–4	P22MUX[1:0]	0x0	H0	R/W	
		3–2	P21MUX[1:0]	0x0	H0	R/W	
		1–0	P20MUX[1:0]	0x0	H0	R/W	
0x4230	P3DAT (P3 Port Data Register)	15–8	P3OUT[7:0]	0x00	H0	R/W	–
		7–0	P3IN[7:0]	x	H0	R	
0x4232	P3IOEN (P3 Port Enable Register)	15–8	P3IEN[7:0]	0x00	H0	R/W	–
		7–0	P3OEN[7:0]	0x00	H0	R/W	
0x4234	P3RCTL (P3 Port Pull-up/down Control Register)	15–8	P3PDPU[7:0]	0x00	H0	R/W	–
		7–0	P3REN[7:0]	0x00	H0	R/W	
0x423c	P3MODSEL (P3 Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–0	P3SEL[7:0]	0x00	H0	R/W	
0x423e	P3FNCSEL (P3 Port Function Select Register)	15–14	P37MUX[1:0]	0x0	H0	R	–
		13–12	P36MUX[1:0]	0x0	H0	R/W	
		11–10	P35MUX[1:0]	0x0	H0	R/W	
		9–8	P34MUX[1:0]	0x0	H0	R/W	
		7–6	P33MUX[1:0]	0x0	H0	R/W	
		5–4	P32MUX[1:0]	0x0	H0	R/W	
		3–2	P31MUX[1:0]	0x0	H0	R/W	
		1–0	P30MUX[1:0]	0x0	H0	R/W	
0x4240	P4DAT (P4 Port Data Register)	15–10	–	0x00	–	R	–
		9–8	P4OUT[1:0]	0x0	H0	R/W	
		7–2	–	0x00	–	R	
		1–0	P4IN[1:0]	x	H0	R	
0x4242	P4IOEN (P4 Port Enable Register)	15–10	–	0x00	–	R	–
		9–8	P4IEN[1:0]	0x0	H0	R/W	
		7–2	–	0x00	–	R	
		1–0	P4OEN[1:0]	0x0	H0	R/W	
0x4244	P4RCTL (P4 Port Pull-up/down Control Register)	15–10	–	0x00	–	R	–
		9–8	P4PDPU[1:0]	0x0	H0	R/W	
		7–2	–	0x00	–	R	
		1–0	P4REN[1:0]	0x0	H0	R/W	
0x424c	P4MODSEL (P4 Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1–0	P4SEL[1:0]	0x0	H0	R/W	
0x424e	P4FNCSEL (P4 Port Function Select Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–2	P41MUX[1:0]	0x0	H0	R/W	
		1–0	P40MUX[1:0]	0x0	H0	R/W	
0x42d0	PDDAT (Pd Port Data Register)	15–11	–	0x00	–	R	–
		10–8	PDOUT[2:0]	0x0	H0	R/W	
		7–3	–	0x00	–	R	
		2–0	PDIN[2:0]	x	H0	R	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x42d2	PDIOEN (Pd Port Enable Register)	15–11	–	0x00	–	R	–
		10–8	PDIEN[2:0]	0x0	H0	R/W	
		7–3	–	0x00	–	R	
		2–0	PDOEN[2:0]	0x0	H0	R/W	
0x42d4	PDRCTL (Pd Port Pull-up/down Control Register)	15–11	–	0x00	–	R	–
		10–8	PDPDPU[2:0]	0x0	H0	R/W	
		7–3	–	0x00	–	R	
		2–0	PDREN[2:0]	0x0	H0	R/W	
0x42dc	PDMODSEL (Pd Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2–0	PDSEL[2:0]	0x7	H0	R/W	
0x42de	PDFNCSEL (Pd Port Function Select Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5–4	PD2MUX[1:0]	0x0	H0	R/W	
		3–2	PD1MUX[1:0]	0x0	H0	R/W	
		1–0	PD0MUX[1:0]	0x0	H0	R/W	
0x42e0	PCLK (P Port Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/WP	
		7–4	CLKDIV[3:0]	0x0	H0	R/WP	
		3–2	KRSTCFG[1:0]	0x0	H0	R/WP	
		1–0	CLKSRC[1:0]	0x0	H0	R/WP	
0x42e2	PINTFGRP (P Port Interrupt Flag Group Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	P1INT	0	H0	R	
		0	P0INT	0	H0	R	

0x4380–0x438e UART (UART)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4380	UA0CLK (UART Ch.0 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x4382	UA0MOD (UART Ch.0 Mode Register)	15–10	–	0x00	–	R	–
		9	INVIRRX	0	H0	R/W	
		8	INVIRTX	0	H0	R/W	
		7	–	0	–	R	
		6	PUEN	0	H0	R/W	
		5	OUTMD	0	H0	R/W	
		4	IRMD	0	H0	R/W	
		3	CHLN	0	H0	R/W	
		2	PREN	0	H0	R/W	
		1	PRMD	0	H0	R/W	
		0	STPB	0	H0	R/W	
0x4384	UA0BR (UART Ch.0 Baud-Rate Register)	15–12	–	0x0	–	R	–
		11–8	FMD[3:0]	0x0	H0	R/W	
		7–0	BRT[7:0]	0x00	H0	R/W	
0x4386	UA0CTL (UART Ch.0 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x4388	UA0TXD (UART Ch.0 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x438a	UA0RXD (UART Ch.0 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x438c	UA0INTF (UART Ch.0 Status and Interrupt Flag Register)	15–10	–	0x00	–	R	–
		9	RBSY	0	H0/S0	R	
		8	TBSY	0	H0/S0	R	
		7	–	0	–	R	
		6	TENDIF	0	H0/S0	R/W	Cleared by writing 1.
		5	FEIF	0	H0/S0	R/W	Cleared by writing 1 or reading the UA0RXD register.
		4	PEIF	0	H0/S0	R/W	
		3	OEIF	0	H0/S0	R/W	Cleared by writing 1.
		2	RB2FIF	0	H0/S0	R	Cleared by reading the UA0RXD register.
		1	RB1FIF	0	H0/S0	R	
		0	TBEIF	1	H0/S0	R	Cleared by writing to the UA0TXD register.
0x438e	UA0INTE (UART Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6	TENDIE	0	H0	R/W	
		5	FEIE	0	H0	R/W	
		4	PEIE	0	H0	R/W	
		3	OEIE	0	H0	R/W	
		2	RB2FIE	0	H0	R/W	
		1	RB1FIE	0	H0	R/W	
		0	TBEIE	0	H0	R/W	

0x43a0–0x43ac 16-bit Timer (T16) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x43a0	T16_1CLK (T16 Ch.1 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x43a2	T16_1MOD (T16 Ch.1 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x43a4	T16_1CTL (T16 Ch.1 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7–2	–	0x00	–	R	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x43a6	T16_1TR (T16 Ch.1 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x43a8	T16_1TC (T16 Ch.1 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x43aa	T16_1INTF (T16 Ch.1 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x43ac	T16_1INTE (T16 Ch.1 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x43b0–0x43ba SPI (SPI) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x43b0	SPI0MOD (SPI Ch.0 Mode Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	PUEN	0	H0	R/W	
		4	NOCLKDIV	0	H0	R/W	
		3	LSBFST	0	H0	R/W	
		2	CPHA	0	H0	R/W	
		1	CPOL	0	H0	R/W	
		0	MST	0	H0	R/W	
0x43b2	SPI0CTL (SPI Ch.0 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x43b4	SPI0TXD (SPI Ch.0 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x43b6	SPI0RXD (SPI Ch.0 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x43b8	SPI0INTF (SPI Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	BSY	0	H0	R	
		2	TENDIF	0	H0/S0	R/W	Cleared by writing 1.
		1	RBFIF	0	H0/S0	R	Cleared by reading the SPI0RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the SPI0TXD register.
0x43ba	SPI0INTE (SPI Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	TENDIE	0	H0	R/W	
		1	RBFIE	0	H0	R/W	
		0	TBEIE	0	H0	R/W	

0x43c0–0x43d2 I²C (I2C)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x43c0	I2C0CLK (I2C Ch.0 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x43c2	I2C0MOD (I2C Ch.0 Mode Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	OADR10	0	H0	R/W	
		1	GCEN	0	H0	R/W	
		0	–	0	–	R	
0x43c4	I2C0BR (I2C Ch.0 Baud-Rate Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	BRT[6:0]	0x7f	H0	R/W	
0x43c8	I2C0OADR (I2C Ch.0 Own Address Register)	15–10	–	0x00	–	R	–
		9–0	OADR[9:0]	0x000	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x43ca	I2C0CTL (I2C Ch.0 Control Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	MST	0	H0	R/W	
		4	TXNACK	0	H0/S0	R/W	
		3	TXSTOP	0	H0/S0	R/W	
		2	TXSTART	0	H0/S0	R/W	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x43cc	I2C0TXD (I2C Ch.0 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x43ce	I2C0RXD (I2C Ch.0 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x43d0	I2C0INTF (I2C Ch.0 Status and Interrupt Flag Register)	15–13	–	0x0	–	R	–
		12	SDALOW	0	H0	R	
		11	SCLLOW	0	H0	R	
		10	BSY	0	H0/S0	R	
		9	TR	0	H0	R	
		8	–	0	–	R	
		7	BYTEENDIF	0	H0/S0	R/W	Cleared by writing 1.
		6	GCIF	0	H0/S0	R/W	
		5	NACKIF	0	H0/S0	R/W	
		4	STOPIF	0	H0/S0	R/W	
		3	STARTIF	0	H0/S0	R/W	
		2	ERRIF	0	H0/S0	R/W	
		1	RBFIF	0	H0/S0	R	Cleared by reading the I2C0RXD register.
		0	TBEIF	0	H0/S0	R	Cleared by writing to the I2C0TXD register.
0x43d2	I2C0INTE (I2C Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7	BYTEENDIE	0	H0	R/W	
		6	GCIE	0	H0	R/W	
		5	NACKIE	0	H0	R/W	
		4	STOPIE	0	H0	R/W	
		3	STARTIE	0	H0	R/W	
		2	ERRIE	0	H0	R/W	
		1	RBFIE	0	H0	R/W	
		0	TBEIE	0	H0	R/W	

0x5000–0x500e 16-bit PWM Timer (T16A3) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5000	T16A0CLK (T16A3 Ch.0 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3	–	0	–	R	
		2	MULTIMD	0	H0	R/W	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5002	T16A0CTL (T16A3 Counter Ch.0 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7	–	X	–	R	Read value is undefined.
		6	HCM	0	H0	R/W	–
		5–4	CCABCNT[1:0]	0x0	H0	R/W	
		3	CBUFEN	0	H0	R/W	
		2	TRMD	0	H0	R/W	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5004	T16A0TC (T16A3 Counter Ch.0 Data Register)	15–0	T16ATC[15:0]	0x0000	H0	R	–
0x5006	T16A0CCCTL (T16A3 Comparator/ Capture Ch.0 Control Register)	15–14	CAPBTRG[1:0]	0x0	H0	R/W	–
		13–12	TOUTBMD[1:0]	0x0	H0	R/W	
		11–10	–	0x0	–	R	
		9	TOUTBINV	0	H0	R/W	
		8	CCBMD	0	H0	R/W	
		7–6	CAPATRG[1:0]	0x0	H0	R/W	
		5–4	TOUTAMD[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1	TOUTAINV	0	H0	R/W	
		0	CCAMD	0	H0	R/W	
0x5008	T16A0CCA (T16A3 Comparator/ Capture Ch.0 A Data Register)	15–0	CCA[15:0]	0x0000	H0	R/W	–
0x500a	T16A0CCB (T16A3 Comparator/ Capture Ch.0 B Data Register)	15–0	CCB[15:0]	0x0000	H0	R/W	–
0x500c	T16A0INTF (T16A3 Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPBOWIF	0	H0	R/W	Cleared by writing 1.
		4	CAPAOWIF	0	H0	R/W	
		3	CAPBIF	0	H0	R/W	
		2	CAPAIF	0	H0	R/W	
		1	CMPBIF	0	H0	R/W	
		0	CMPAIF	0	H0	R/W	
0x500e	T16A0INTE (T16A3 Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPBOWIE	0	H0	R/W	
		4	CAPAOWIE	0	H0	R/W	
		3	CAPBIE	0	H0	R/W	
		2	CAPAIE	0	H0	R/W	
		1	CMPBIE	0	H0	R/W	
		0	CMPAIE	0	H0	R/W	

0x5020–0x502e 16-bit PWM Timer (T16A3) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5020	T16A1CLK (T16A3 Ch.1 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5022	T16A1CTL (T16A3 Counter Ch.1 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7	–	X	–	R	Read value is undefined.
		6	HCM	0	H0	R/W	–
		5–4	CCABCNT[1:0]	0x0	H0	R/W	
		3	CBUFEN	0	H0	R/W	
		2	TRMD	0	H0	R/W	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5024	T16A1TC (T16A3 Counter Ch.1 Data Register)	15–0	T16ATC[15:0]	0x0000	H0	R	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5026	T16A1CCCTL (T16A3 Comparator/ Capture Ch.1 Control Register)	15–14	CAPBTRG[1:0]	0x0	H0	R/W	–
		13–12	TOUTBMD[1:0]	0x0	H0	R/W	
		11–10	–	0x0	–	R	
		9	TOUTBINV	0	H0	R/W	
		8	CCBMD	0	H0	R/W	
		7–6	CAPATRG[1:0]	0x0	H0	R/W	
		5–4	TOUTAMD[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1	TOUTAINV	0	H0	R/W	
		0	CCAMD	0	H0	R/W	
0x5028	T16A1CCA (T16A3 Comparator/ Capture Ch.1 A Data Register)	15–0	CCA[15:0]	0x0000	H0	R/W	–
0x502a	T16A1CCB (T16A3 Comparator/ Capture Ch.1 B Data Register)	15–0	CCB[15:0]	0x0000	H0	R/W	–
0x502c	T16A1INTF (T16A3 Ch.1 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPBOWIF	0	H0	R/W	Cleared by writing 1.
		4	CAPAOWIF	0	H0	R/W	
		3	CAPBIF	0	H0	R/W	
		2	CAPAIF	0	H0	R/W	
		1	CMPBIF	0	H0	R/W	
		0	CMPAIF	0	H0	R/W	
0x502e	T16A1INTE (T16A3 Ch.1 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPBOWIE	0	H0	R/W	
		4	CAPAOWIE	0	H0	R/W	
		3	CAPBIE	0	H0	R/W	
		2	CAPAIE	0	H0	R/W	
		1	CMPBIE	0	H0	R/W	
		0	CMPAIE	0	H0	R/W	

0x5180–0x5186 Clock Timer (CT)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5180	CTCTL (CT Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	W	
		0	MODEN	0	H0	R/W	
0x5182	CTDAT (CT Counter Data Register)	15–8	–	0x00	–	R	–
		7–0	CTCNT[7:0]	0x00	H0/S0	R	
0x5184	CTINTF (CT Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	CT32HZIF	0	H0	R/W	Cleared by writing 1.
		2	CT8HZIF	0	H0	R/W	
		1	CT2HZIF	0	H0	R/W	
		0	CT1HZIF	0	H0	R/W	
0x5186	CTINTE (CT Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	CT32HZIE	0	H0	R/W	
		2	CT8HZIE	0	H0	R/W	
		1	CT2HZIE	0	H0	R/W	
		0	CT1HZIE	0	H0	R/W	

0x5260–0x526c 16-bit Timer (T16) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5260	T16_2CLK (T16 Ch.2 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5262	T16_2MOD (T16 Ch.2 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x5264	T16_2CTL (T16 Ch.2 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7–2	–	0x00	–	R	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5266	T16_2TR (T16 Ch.2 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x5268	T16_2TC (T16 Ch.2 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x526a	T16_2INTF (T16 Ch.2 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x526c	T16_2INTE (T16 Ch.2 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x5270–0x527a SPI (SPI) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5270	SPI1MOD (SPI Ch.1 Mode Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	PUEN	0	H0	R/W	
		4	NOCLKDIV	0	H0	R/W	
		3	LSBFST	0	H0	R/W	
		2	CPHA	0	H0	R/W	
		1	CPOL	0	H0	R/W	
		0	MST	0	H0	R/W	
0x5272	SPI1CTL (SPI Ch.1 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5274	SPI1TXD (SPI Ch.1 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x5276	SPI1RXD (SPI Ch.1 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x5278	SPI1INTF (SPI Ch.1 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	BSY	0	H0	R	
		2	TENDIF	0	H0/S0	R/W	Cleared by writing 1.
		1	RBFIF	0	H0/S0	R	Cleared by reading the SPI1RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the SPI1TXD register.

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x527a	SPI1INTE (SPI Ch.1 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	TENDIE	0	H0	R/W	
		1	RBFIE	0	H0	R/W	
		0	TBEIE	0	H0	R/W	

0x5280–0x528c 16-bit Timer (T16) Ch.3

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5280	T16_3CLK (T16 Ch.3 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5282	T16_3MOD (T16 Ch.3 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x5284	T16_3CTL (T16 Ch.3 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7–2	–	0x00	–	R	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5286	T16_3TR (T16 Ch.3 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x5288	T16_3TC (T16 Ch.3 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x528a	T16_3INTF (T16 Ch.3 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x528c	T16_3INTE (T16 Ch.3 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x5290–0x529a SPI (SPI) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5290	SPI2MOD (SPI Ch.2 Mode Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	PUEN	0	H0	R/W	
		4	NOCLKDIV	0	H0	R/W	
		3	LSBFST	0	H0	R/W	
		2	CPHA	0	H0	R/W	
		1	CPOL	0	H0	R/W	
		0	MST	0	H0	R/W	
0x5292	SPI2CTL (SPI Ch.2 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5294	SPI2TXD (SPI Ch.2 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x5296	SPI2RXD (SPI Ch.2 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5298	SPI2INTF (SPI Ch.2 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	BSY	0	H0	R	
		2	TENDIF	0	H0/S0	R/W	Cleared by writing 1.
		1	RBFIF	0	H0/S0	R	Cleared by reading the SPI2RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the SPI2TXD register.
0x529a	SPI2INTE (SPI Ch.2 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	TENDIE	0	H0	R/W	
		1	RBFIE	0	H0	R/W	
		0	TBEIE	0	H0	R/W	

0x52e0–0x52ea Parallel Interface (PIO)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x52e0	PIOCLK (PIO Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x52e2	PIOMOD (PIO Mode Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	PUL	0	H0	R/W	
		0	GPIOMD	0	H0	R/W	
0x52e4	PIOCTL (PIO Control Register)	15–9	–	0x00	–	R	–
		8	RACC	0	H0	W	Always read as 0.
		7–2	–	0x00	–	R	–
		1	SFTRST	0	H0	W	Always read as 0.
		0	MODEN	0	H0	R/W	–
0x52e6	PIOWRDAT (PIO Address/Write Data Register)	15–8	PADDR[7:0]	0x00	H0	R/W	–
		7–0	PWDATA[7:0]	0x00	H0	R/W	
0x52e8	PIORDDAT (PIO Read Data Register)	15–8	–	0x00	–	R	–
		7–0	PRDATA[7:0]	0x00	H0	R	
0x52ea	PIOSTAT (PIO Status Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	WBUSY	0	H0/S0	R	
		0	RBUSY	0	H0/S0	R	

0x5380–0x5384 EPD Timing Controller (EPD Tcon)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5380	EPDCTL (EPD Tcon Control Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	MODEN	0	H0	R/W	
0x5382	EPDINTF (EPD Tcon Interrupt Flag and Status Register)	15–9	–	0x00	–	R	–
		8	BUSY	1	H0	R	
		7–1	–	0x00	–	R	
		0	ENDIF	0	H0	R/W	Cleared by writing 1.
0x5384	EPDINTE (EPD Tcon Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	ENDIE	0	H0	R/W	

0x5440–0x5450 R/F Converter (RFC) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5440	RFC0CLK (RFC Ch.0 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	1	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5442	RFC0CTL (RFC Ch.0 Control Register)	15–9	–	0x00	–	R	–
		8	RFCLKMD	0	H0	R/W	
		7	CONEN	0	H0	R/W	
		6	EVTEN	0	H0	R/W	
		5–4	SMODE[1:0]	0x0	H0	R/W	
		3–1	–	0x0	–	R	
		0	MODEN	0	H0	R/W	
0x5444	RFC0TRG (RFC Ch.0 Oscillation Trigger Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	SSENB	0	H0	R/W	
		1	SSENA	0	H0	R/W	
		0	SREF	0	H0	R/W	
0x5446	RFC0MCL (RFC Ch.0 Measurement Counter Low Register)	15–0	MC[15:0]	0x0000	H0	R/W	–
0x5448	RFC0MCH (RFC Ch.0 Measurement Counter High Register)	15–8	–	0x00	–	R	–
		7–0	MC[23:16]	0x00	H0	R/W	
0x544a	RFC0TCL (RFC Ch.0 Time Base Counter Low Register)	15–0	TC[15:0]	0x0000	H0	R/W	–
0x544c	RFC0TCH (RFC Ch.0 Time Base Counter High Register)	15–8	–	0x00	–	R	–
		7–0	TC[23:16]	0x00	H0	R/W	
0x544e	RFC0INTF (RFC Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	OVTCIF	0	H0	R/W	Cleared by writing 1.
		3	OVMCIF	0	H0	R/W	
		2	ESENBIF	0	H0	R/W	
		1	ESENAIF	0	H0	R/W	
		0	EREFIF	0	H0	R/W	
0x5450	RFC0INTE (RFC Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	OVTCIE	0	H0	R/W	
		3	OVMCIE	0	H0	R/W	
		2	ESENBIE	0	H0	R/W	
		1	ESENAIE	0	H0	R/W	
		0	EREFIE	0	H0	R/W	

0x5460–0x5470 R/F Converter (RFC) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5460	RFC1CLK (RFC Ch.1 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	1	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5462	RFC1CTL (RFC Ch.1 Control Register)	15–9	–	0x00	–	R	–
		8	RFCLKMD	0	H0	R/W	
		7	CONEN	0	H0	R/W	
		6	EVTEN	0	H0	R/W	
		5–4	SMODE[1:0]	0x0	H0	R/W	
		3–1	–	0x0	–	R	
		0	MODEN	0	H0	R/W	
0x5464	RFC1TRG (RFC Ch.1 Oscillation Trigger Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2	SSENB	0	H0	R/W	
		1	SSENA	0	H0	R/W	
		0	SREF	0	H0	R/W	
0x5466	RFC1MCL (RFC Ch.1 Measurement Counter Low Register)	15–0	MC[15:0]	0x0000	H0	R/W	–
0x5468	RFC1MCH (RFC Ch.1 Measurement Counter High Register)	15–8	–	0x00	–	R	–
		7–0	MC[23:16]	0x00	H0	R/W	
0x546a	RFC1TCL (RFC Ch.1 Time Base Counter Low Register)	15–0	TC[15:0]	0x0000	H0	R/W	–
0x546c	RFC1TCH (RFC Ch.1 Time Base Counter High Register)	15–8	–	0x00	–	R	–
		7–0	TC[23:16]	0x00	H0	R/W	
0x546e	RFC1INTF (RFC Ch.1 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	OVTCIF	0	H0	R/W	Cleared by writing 1.
		3	OVMCIF	0	H0	R/W	
		2	ESENBIF	0	H0	R/W	
		1	ESENAIF	0	H0	R/W	
		0	EREFIF	0	H0	R/W	
0x5470	RFC1INTE (RFC Ch.1 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	OVTCIE	0	H0	R/W	
		3	OVMCIE	0	H0	R/W	
		2	ESENBIE	0	H0	R/W	
		1	ESENAIE	0	H0	R/W	
		0	EREFIE	0	H0	R/W	

0x54c0–0x54ca Temperature Detection Circuit (TEM)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x54c0	TEMCLK (TEM Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	CLKDIV[1:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x54c2	TEMTMG (TEM Timing Register)	15–8	–	0x00	–	R	–
		7–0	CVTM[7:0]	0x00	H0	R/W	
0x54c4	TEMCTL (TEM Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	TEMTRG	0	H0	W	Always read as 0.
		0	MODEN	0	H0	R/W	–
0x54c6	TEMRSLT (TEM Conversion Result Register)	15–8	–	0x00	–	R	–
		7–0	TEMP[7:0]	0x00	H0	R	
0x54c8	TEMINTF (TEM Interrupt Flag and Status Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	TEMST	0	H0	R	
		3–1	–	0x0	–	R	
		0	TEMIF	0	H0	R/W	Cleared by reading the TEMRSLT register.
0x54ca	TEMINTE (TEM Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TEMIE	0	H0	R/W	

0xffff90 Debugger (DBG)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0xffff90	DBRAM (Debug RAM Base Register)	31–24	–	0x00	–	R	–
		23–0	DBRAM[23:0]	0x00 17c0	H0	R	

Appendix B Power Saving

Current consumption will vary dramatically, depending on CPU operating mode, operation clock frequency, peripheral circuits being operated, and V_{D1} regulator operating mode. Listed below are the control methods for saving power.

B.1 Operating Status Configuration Examples for Power Saving

Table B.1.1 lists typical examples of operating status configuration with consideration given to power saving.

Table B.1.1 Typical Operating Status Configuration Examples

Operating status configuration	Current consumption	V_{D1}	OSC1	OSC3A/ OSC3B/ EXOSC	RTC	CPU	Current consumption listed in electrical characteristics
Standby	↑		OFF		OFF	SLEEP	I_{SLP1}
Clock counting	Low	Economy		OFF		SLEEP	I_{SLP2}
						HALT	I_{HALT2}
Low-speed processing			ON		ON	OSC1 RUN	I_{RUN20}
Peripheral circuit operations						SLEEP or HALT	I_{HALT1}
High-speed processing	High ↓	Normal		ON		OSC3A/OSC3B/ EXOSC RUN	I_{RUN10}

If the current consumption order by the operating status configuration shown in Table B.1.1 is different from one that is listed in "Electrical Characteristics," check the settings shown below.

PWGVD1CTL.REGMODE[1:0] bits of the power generator

If the PWGVD1CTL.REGMODE[1:0] bits of the power generator is 0x2 (normal mode) when the CPU enters SLEEP mode, current consumption in SLEEP mode will be larger than I_{SLP1} or I_{SLP2} that is listed in "Electrical Characteristics." Set the PWGVD1CTL.REGMODE[1:0] bits to 0x3 (economy mode) or 0x0 (automatic mode) before executing the slp instruction.

CLGOSC. *xxxx*SLPC bits of the clock generator

Setting the CLGOSC.OSC3BSLPC, OSC3ASLPC, OSC1SLPC, or EXOSCSLPC bit of the clock generator to 0 disables the oscillator circuit stop control when the slp instruction is executed. To stop the oscillator circuits during SLEEP mode, set these bits to 1.

MODEN bits of the peripheral circuits

Setting the MODEN bit of each peripheral circuit to 1 starts supplying the operating clock enabling the peripheral circuit to operate. To reduce current consumption, set the MODEN bits of unnecessary peripheral circuits to 0.

External components for OSC1A oscillator circuit

External components for the OSC1A oscillator circuit affect current consumption.

- Using lower OSC1A external gate and drain capacitances decreases current consumption.
- Using a crystal resonator with lower C_L value decreases current consumption.

However, these selections may reduce the oscillation margin and increase the frequency error, therefore, be sure to perform matching evaluation using the actual printed circuit board.

OSC3A oscillator circuit configurations

The OSC3A oscillator circuit provides some configuration items to support various crystal and ceramic resonators. These configurations trade off current consumption for performance as shown below.

- The lower oscillation inverter gain setting (CLGOSC3A.INVN[1:0] bits) decreases current consumption.
- Using lower OSC3A external gate and drain capacitances decreases current consumption.
- Using a resonator with lower C_L value decreases current consumption.

However, these configurations may reduce the oscillation margin and increase the frequency error, therefore, be sure to perform matching evaluation using the actual printed circuit board.

B.2 Other Power Saving Methods

Supply voltage detector configuration

Continuous operation mode (SVDCTL.SVDMD[1:0] bits = 0x0) always detects the power supply voltage, therefore, it increases current consumption. Set the supply voltage detector to intermittent operation mode or turn it on only when required.

Appendix C Mounting Precautions

This section describes various precautions for circuit board design and IC mounting.

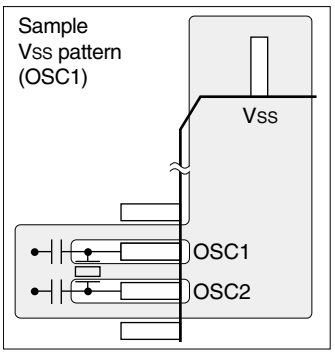
Oscillator circuit

- Oscillation characteristics depend on factors such as components used (resonator, C_G, C_D) and circuit board patterns. In particular, with crystal resonators, select the appropriate capacitors (C_G, C_D) only after fully evaluating components actually mounted on the circuit board.
- Oscillator clock disturbances caused by noise may cause malfunctions. To prevent such disturbances, consider the following points.

(1) Components such as a resonator, resistors, and capacitors connected to the OSC1 (OSC3) and OSC2 (OSC4) pins should have the shortest connections possible.

(2) Wherever possible, avoid locating digital signal lines within 3 mm of the OSC1 (OSC3) and OSC2 (OSC4) pins or related circuit components and wiring. Rapidly-switching signals, in particular, should be kept at a distance from these components. Since the spacing between layers of multi-layer printed circuit boards is a mere 0.1 mm to 0.2 mm, the above precautions also apply when positioning digital signal lines on other layers.
Never place digital signal lines alongside such components or wiring, even if more than 3 mm distance or located on other layers. Avoid crossing wires.

(3) Use Vss to shield OSC1 (OSC3) and OSC2 (OSC4) pins and related wiring (including wiring for adjacent circuit board layers). Layers wired should be adequately shielded as shown to the right. Fully ground adjacent layers, where possible. At minimum, shield the area at least 5 mm around the above pins and wiring.
Even after implementing these precautions, avoid configuring digital signal lines in parallel, as described in (2) above. Avoid crossing even on discrete layers, except for lines carrying signals with low switching frequencies.



(4) After implementing these precautions, check the FOUT pin output clock waveform by running the actual application program within the product.
For OSC3ACLK, confirm that the frequency is as designed, is free of noise, and has minimal jitter.
For OSC1CLK, in particular, enlarge the areas before and after the clock rising and falling edges and take special care to confirm that the regions approximately 100 ns to either side are free of clock or spiking noise.

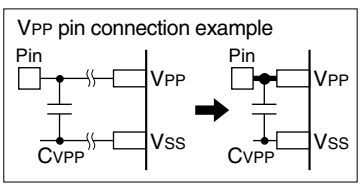
Failure to observe precautions (1) to (3) adequately may lead to jitter in OSC3ACLK and noise in the OSC1CLK. Jitter in OSC3ACLK will reduce operating frequencies, while noise in the OSC1CLK will destabilize timers that use OSC1CLK as well as CPU Core operations.

#RESET pin

Components such as a switch and resistor connected to the #RESET pin should have the shortest connections possible to prevent noise-induced resets.

V_{PP} pin

If fluctuations in the Flash programming voltage V_{PP} is large, connect a capacitor C_{VPP} between the V_{SS} and V_{PP} pins to suppress fluctuations within $V_{PP} \pm 1$ V. The C_{VPP} should be placed as close to the V_{PP} pin as possible and use a sufficiently thick wiring pattern that allows current of several tens of mA to flow.

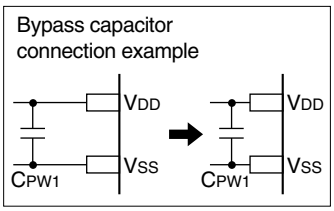


Power supply circuit

Sudden power supply fluctuations due to noise will cause malfunctions. Consider the following issues.

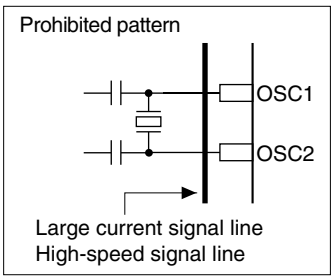
(1) Connections from the power supply to the VDD and VSS pins should be implemented via the shortest, thickest patterns possible.

(2) If a bypass capacitor is connected between VDD and VSS, connections between the VDD and VSS pins should be as short as possible.



Signal line location

- To prevent electromagnetically-induced noise arising from mutual induction, large-current signal lines should not be positioned close to pins susceptible to noise, such as oscillator and analog measurement pins.
- Locating signal lines in parallel over significant distances or crossing signal lines operating at high speed will cause malfunctions due to noise generated by mutual interference.
- The voltage boost capacitor drive lines are more likely to generate noise, therefore keep a distance between the lines and pins susceptible to noise.



Handling of light (for bare chip mounting)

The characteristics of semiconductor components can vary when exposed to light. ICs may malfunction or nonvolatile memory data may be corrupted if ICs are exposed to light.
Consider the following precautions for circuit boards and products in which this IC is mounted to prevent IC malfunctions attributable to light exposure.

(1) Design and mount the product so that the IC is shielded from light during use.

(2) Shield the IC from light during inspection processes.

(3) Shield the IC on the upper, underside, and side faces of the IC chip.

(4) Mount the IC chip within one week of opening the package. If the IC chip must be stored before mounting, take measures to ensure light shielding.

(5) Adequate evaluations are required to assess nonvolatile memory data retention characteristics before product delivery if the product is subjected to heat stress exceeding regular reflow conditions during mounting processes.

Unused pins

(1) I/O port (P) pins
Unused pins should be left open. The control registers should be fixed at the initial status.

(2) OSC1, OSC2, OSC3, OSC4, and EXOSC pins
If the OSC1A oscillator circuit, OSC3A oscillator circuit, or EXOSC input circuit is not used, the OSC1 and OSC2 pins, the OSC3 and OSC4 pins, or the EXOSC pin should be left open. The control registers should be fixed at the initial status (disabled).

Miscellaneous

Minor variations over time may result in electrical damage arising from disturbances in the form of voltages exceeding the absolute maximum rating when mounting the product in addition to physical damage. The following factors can give rise to these variations:

(1) Electromagnetically-induced noise from industrial power supplies used in mounting reflow, reworking after mounting, and individual characteristic evaluation (testing) processes
(2) Electromagnetically-induced noise from a solder iron when soldering

In particular, during soldering, take care to ensure that the soldering iron GND (tip potential) has the same potential as the IC GND.

Appendix D Measures Against Noise

To improve noise immunity, take measures against noise as follows:

Noise Measures for V_{DD} and V_{SS} Power Supply Pins

When noise falling below the rated voltage is input, the POR/BOR circuit issues a reset request and the system is reset. If desired operations cannot be achieved, take measures against noise on the circuit board, such as designing close patterns for circuit board power supply circuits, adding noise-filtering decoupling capacitors, and adding surge/noise prevention components on the power supply line.
For the recommended patterns on the circuit board, see "Mounting Precautions" in Appendix.

Noise Measures for #RESET Pin

If noise is input to the #RESET pin, the IC may be reset. Therefore, the circuit board must be designed properly taking noise measures into consideration.
For the recommended patterns on the circuit board, see "Mounting Precautions" in Appendix.

Noise Measures for Oscillator Pins

The oscillator input pins must pass a signal of small amplitude, so they are hypersensitive to noise. Therefore, the circuit board must be designed properly taking noise measures into consideration.
For the recommended patterns on the circuit board, see "Mounting Precautions" in Appendix.

Noise Measures for Debug Pins

This product provides the input/output pins (DCLK, DST2, and DSIO) to connect ICDmini (S5U1C17001H) for debugging. If noise is input to these pins with the debugging function enabled, the S1C17 Core may enter DEBUG mode. To prevent unexpected transitions to DEBUG mode caused by extraneous noise, switch the DCLK, DST2, and DSIO pins to general-purpose I/O port pins within the initialization routine when the debug functions are not used.
For details of the pin functions and the function switch control, see the "I/O Ports" chapter.

Note: Do not perform the function switching shown above when the application is under development, as the debug functions must be used. The debugging cannot be performed after the pin function is switched. The above processing must be added after the application development has completed and debugging is no longer necessary.

The DSIO pin should be pulled up with a 10 kΩ resistor when using the debug pin functions.

Noise Measures for Interrupt Input Pins

This product is able to generate a port input interrupt when the input signal changes. The interrupt is generated when an input signal edge is detected, therefore, an interrupt may occur if the signal changes due to extraneous noise. To prevent occurrence of unexpected interrupts due to extraneous noise, enable the chattering filter circuit when using the port input interrupt.
For details of the port input interrupt and chattering filter circuit, see the "I/O Ports" chapter.

Noise Measures for UART Pins

This product includes a UART for asynchronous communications. The UART starts receive operation when it detects a low level input from the SIN*n* pin. Therefore, a receive operation may be started if the SIN*n* pin is set to low due to extraneous noise. In this case, a receive error will occur or invalid data will be received.
To prevent the UART from malfunction caused by extraneous noise, take the following measures:

- Stop the UART operations while asynchronous communication is not performed.
- Execute the resending process via software after executing the receive error handler with a parity check.

For details of the pin functions and the function switch control, see the "I/O Ports" chapter. For the UART control and details of receive errors, see the "UART" chapter.

Appendix E Initialization Routine

The following lists typical vector tables and initialization routines:

boot.s

```
.org    0x8000
.section .rodata                                                          ...(1)

; ======================================================================
;       Vector table
; ======================================================================
                                ; interrupt  vector  interrupt
                                ; number     offset  source

.long BOOT                      ; 0x00       0x00    reset                ...(2)
.long unalign_handler           ; 0x01       0x04    unalign
.long nmi_handler               ; 0x02       0x08    NMI
.long int03_handler             ; 0x03       0x0c    -
.long svd_handler               ; 0x04       0x10    SVD
.long pport_handler             ; 0x05       0x14    PPORT
.long clg_handler               ; 0x06       0x18    CLG
.long rtc_handler               ; 0x07       0x1c    RTC
.long t16_0_handler             ; 0x08       0x20    T16 ch0
.long uart_handler              ; 0x09       0x24    UART
.long t16_1_handler             ; 0x0a       0x28    T16 ch1
.long spi_0_handler             ; 0x0b       0x2c    SPI ch0
.long i2c_handler               ; 0x0c       0x30    I2C
.long ct_handler                ; 0x0d       0x34    CT
.long t16_2_handler             ; 0x0e       0x38    T16 ch2
.long spi_1_handler             ; 0x0f       0x3c    SPI ch1
.long t16_3_handler             ; 0x10       0x40    T16 ch3
.long spi_2_handler             ; 0x11       0x44    SPI ch2
.long t16a3_0_handler           ; 0x12       0x48    T16A3 ch0
.long t16a3_1_handler           ; 0x13       0x4c    T16A3 ch1
.long rfc_0_handler             ; 0x14       0x50    RFC ch0
.long rfc_1_handler             ; 0x15       0x54    RFC ch1
.long epd_tcon_handler          ; 0x16       0x58    EPD Tcon
.long tem_handler               ; 0x17       0x5c    TEM
.long int18_handler             ; 0x18       0x60    -
.long int19_handler             ; 0x19       0x64    -
.long int1a_handler             ; 0x1a       0x68    -
.long int1b_handler             ; 0x1b       0x6c    -
.long int1c_handler             ; 0x1c       0x70    -
.long int1d_handler             ; 0x1d       0x74    -
.long int1e_handler             ; 0x1e       0x78    -
.long int1f_handler             ; 0x1f       0x7c    -

; ======================================================================
;       Program code
; ======================================================================

.text                                                                     ...(3)
.align 1

BOOT:
        ; ===== Initialize ============================================

        ; ----- Stack pointer --------------------
        Xld.a   %sp, 0x17c0                                               ...(4)

        ; ----- Memory controller ----------------
        Xld.a   %r1, 0x41b0     ; FLASHC register address

        ; Flash read wait cycle
        Xld.a   %r0, 0x00       ; 0x00 = No wait, 0x01 = 1 wait, or 0x02 = 2 wait
        ld.b    [%r1], %r0      ; [0x41b0] <= 0x00                        ...(5)

        ; ===== Main routine ==========================================
         ...
```

```
; ========================================================================
;       Interrupt handler
; ========================================================================
; ----- Address unalign --------------------------
unalign_handler:
        ...

; ----- NMI ----------------------------------
nmi_handler:
        ...
```

---

(1) A "`.rodata`" section is declared to locate the vector table in the "`.vector`" section.

(2) Interrupt handler routine addresses are defined as vectors.
"`int`*`XX`*`_handler`" can be used for software interrupts.

(3) The program code is written in the "`.text`" section.

(4) Sets the stack pointer.

(5) Sets the number of Flash memory read cycles.
(See the "Memory and Bus" chapter.)

Revision History

Code No.	Page	Contents
412486300	All	New establishment
412486300b	8-2	8.2.2 External Connection (Old) Figure 8.2.2.1 For the EXSVD pin input voltage range, refer to “Supply Voltage Detector Characteristics, EXSVD pin input voltage range V_{EXSVD}” in the “Electrical Characteristics” chapter. (New) Modified the figure 8.2.2.1 (added resistors). R_{EXT} resistance value must be determined so that it will be sufficiently smaller than the EXSVD input impedance R_{EXSVD}. For the EXSVD pin input voltage range and the EXSVD input impedance, refer to “Supply Voltage Detector Characteristics” in the “Electrical Characteristics” chapter.
	8-4	8.4.2 SVD Operations (Old) Continuous operation mode ... Furthermore, an interrupt (if the SVDCTL.SVDRE[3:0] bits ≠ 0xa) or a reset (if the SVDCTL.SVDRE[3:0] bits = 0xa) can be generated when the SVDINTF.SVDDT bit is set to 1 (low power supply voltage is detected). (New) Continuous operation mode ... Furthermore, an interrupt (if the SVDCTL.SVDRE[3:0] bits ≠ 0xa) or a reset (if the SVDCTL.SVDRE[3:0] bits = 0xa) can be generated when the SVDINTF.SVDDT bit is set to 1 (low power supply voltage is detected). This mode can keep detecting power supply voltage drop after the voltage detection masking time has elapsed even if the IC is placed into SLEEP status or accidental clock stoppage has occurred.
	12-5	12.4.2 Data Transmission in Master Mode (Old) Generating a STOP/repeated START condition When setting the I2CnCTL.TXSTOP bit to 1 while the I2CnINTF.TBEIF bit = 1 (transmit buffer empty) or the I2CnINTF.NACKIF bit = 1 (NACK received), the I2C Ch.n generates a STOP condition. (New) Generating a STOP/repeated START condition After the I2CnINTF.TBEIF bit is set to 1 (transmit buffer empty) or the I2CnINTF.NACKIF bit is set to 1 (NACK received), setting the I2CnCTL.TXSTOP bit to 1 generates a STOP condition.
	12-12	12.4.6 Data Reception in Slave Mode (Old) Data receiving procedure ... 7. Repeat Steps 4 to 6 until the end of data reception. (New) Data receiving procedure ... 7. Repeat Steps 4 to 6 until the end of data reception. 8. Wait for a STOP condition interrupt (I2CnINTF.STOPIF bit = 1) or a START condition interrupt (I2CnINTF.STARTIF bit = 1). i. Go to Step 9 when a STOP condition interrupt has occurred. ii. Go to Step 2 when a START condition interrupt has occurred. 9. Clear the I2CnINTF.STOPIF bit and then terminate data receiving operations.
	22-8	22.8 Supply Voltage Detector (SVD) Characteristics (Old) – (New) Modified the table (added EXSVD input impedance).
412486301	1-2	Features: Table 1.1.1 Features (Old) Watchdog timer (WDT) | Generates NMI or watchdog timer reset. (New) Watchdog timer (WDT) | Generates watchdog timer reset.
	2-4	Power Supply, Reset, and Clocks: Watchdog timer reset (Old) Setting the watchdog timer into reset mode will issue a reset request when the counter overflows. (New) The watchdog timer issues a reset request when the counter overflows.
	2-12	Power Supply, Reset, and Clocks: Figure 2.4.2.1 Operating Mode-to-Mode State Transition Diagram Modified the figure (Interrupt → HALT/SLEEP cancelation signal)
		Power Supply, Reset, and Clocks: Canceling HALT or SLEEP mode (Old) The conditions listed below cancel HALT or SLEEP mode and put the CPU into RUN mode. • Interrupt request from the interrupt controller • NMI from the watchdog timer • Debug interrupt or address misaligned interrupt (New) The conditions listed below generate the HALT/SLEEP cancelation signal to cancel HALT or SLEEP mode and put the CPU into RUN mode. This transition is executed even if the CPU does not accept the interrupt request. • Interrupt request from a peripheral circuit • NMI • Debug interrupt

REVISION HISTORY

Code No.	Page	Contents
412486301	2-13	Power Supply, Reset, and Clocks: CLG System Clock Control Register - Bit 15 WUPMD (Old) No description (New) Notes: ... • When the CLKSCLK.WUPMD bit = 1, be sure to avoid setting both the CLGSCLK.WUPSRC[1:0] bits and the CLGSCLK.WUPDIV[1:0] bits to the same values as the CLGSCLK.CLKSRC[1:0] bits and the CLGSCLK.CLKDIV[1:0] bits, respectively. If the same clock source and division ratio as those that are configured before placing the IC into SLEEP mode are used at wake-up, set the CLKSCLK.WUPMD bit to 0.
	2-14	Power Supply, Reset, and Clocks: CLG System Clock Control Register - Bits 9–8 WUPSRC[1:0] (Old) These bits select the SYSCLK clock source for resetting the CLGSCLK.CLKSRC[1:0] bits at wake-up. When a currently stopped clock source is selected, it will automatically start oscillating or clock input at wake-up. However, this setting is ineffective when the CLGSCLK.WUPMD bit = 0. (New) These bits select the SYSCLK clock source for resetting the CLGSCLK.CLKSRC[1:0] bits at wake-up. However, this setting is ineffective when the CLGSCLK.WUPMD bit = 0. Note: Do not select a clock source that has stopped. When selecting it, set the clock source enable bit to 1 before executing the slp instruction.
	5-1	ITC: Figure 5.1.1 ITC Configuration Modified the figure (The HALT/SLEEP cancelation signal was added. The watchdog timer block was deleted. → GND)
	5-1, 5-3	ITC: Table 5.2.1 Vector Table (Old) TTBR + 0x00 | • Watchdog timer overflow ∗2 TTBR + 0x08 | Watchdog timer overflow ∗2 ∗2 Either reset or NMI can be selected as the watchdog timer interrupt with software. (New) TTBR + 0x00 | • Watchdog timer overflow TTBR + 0x08 | – (Note ∗2 was deleted.)
	5-4	ITC: Peripheral Circuit Interrupt Control (Old) Note: To prevent occurrence of unnecessary interrupts, always clear the corresponding interrupt flag before setting the interrupt enable bit to 1 (interrupt enabled) and before terminating the interrupt handler routine. (New) Note: To prevent occurrence of unnecessary interrupts, the corresponding interrupt flag should be cleared before setting the interrupt enable bit to 1 (interrupt enabled) and before terminating the interrupt handler routine.
		ITC: ITC Interrupt Request Processing (Old) Note: Wake-up operations (SLEEP/HALT cancellation) by an interrupt cannot be disabled even if the interrupt level is set to 0. (New) Deleted
		ITC: NMI (Old) The watchdog timer embedded in this IC can generate a non-maskable interrupt (NMI). This interrupt takes precedence over other interrupts and is unconditionally accepted by the CPU. For detailed information on generating NMI, refer to the “Watchdog Timer” chapter. (New) This IC cannot generate non-maskable interrupts (NMI).
	5-5	ITC: Interrupt Processing by the CPU (Old) Note: At wake-up from HALT or SLEEP mode, the CPU jumps to the interrupt handler routine after executing one instruction. (New) Note: When HALT or SLEEP mode is canceled, the CPU jumps to the interrupt handler routine after executing one instruction.
	6-5	PPORT: Reading input data from a GPIO port (Old) No description (New) Note: The P*x*DAT.P*x*INy bit retains the input port status at 1 clock before being read from the CPU.
		PPORT: Chattering filter function (Old) Input sampling time [second] = 2 / CLK_PPORT frequency [Hz] (Eq.6.2) (New) Input sampling time [second] = 2 to 3 / CLK_PPORT frequency [Hz] (Eq.6.2)
	6-8	PPORT: P*x* Port Interrupt Control Register (Old) Note: To prevent generating unnecessary interrupts, clear the corresponding interrupt flag before enabling interrupts. (New) Note: To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.
	7-1	WDT: Overview (Old) • Includes a 10-bit up counter to count NMI/reset generation cycle. ... • Counter overflow generates a reset or NMI. (New) • Includes a 10-bit up counter to count reset generation cycle. ... • Counter overflow generates a reset.
		WDT: Figure 7.1.1 WDT Configuration Modified the figure (NMIXRST, STATNMI, and the NMI output were deleted.)

Code No.	Page	Contents
412486301	7-1	WDT: WDT Operating Clock (Old) Use the following equation to calculate the WDT counter overflow cycle (NMI/reset generation cycle). (New) Use the following equation to calculate the WDT counter overflow cycle (reset generation cycle).
	7-2	WDT: Starting up WDT (Old) 3. Configure the WDTCTL.NMIXRST bit. (Select NMI or reset mode) 4. Write 1 to the WDTCTL.WDTCNTRST bit. (Reset WDT counter) 5. Write a value other than 0xa to the WDTCTL.WDTRUN[3:0] bits. (Start up WDT) 6. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection) (New) 3. Write 1 to the WDTCTL.WDTCNTRST bit. (Reset WDT counter) 4. Write a value other than 0xa to the WDTCTL.WDTRUN[3:0] bits. (Start up WDT) 5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
		WDT: Resetting WDT (Old) WDT generates a system reset (WDTCTL.NMIXRST bit = 0) or NMI (WDTCTL.NMIXRST bit = 1) when the counter overflows. ... After resetting, WDT starts counting with a new NMI/reset generation cycle. If WDT is not reset within the t_{WDT} cycle for any reason, the CPU is switched to interrupt processing by NMI or reset, the interrupt vector is read out, and the interrupt handler routine is executed. If the counter overflows and generates an NMI without WDT being reset, the WDTCTL.STATNMI bit is set to 1. (New) WDT generates a system reset when the counter overflows. ... After resetting, WDT starts counting with a new reset generation cycle. If WDT is not reset within the t_{WDT} cycle for any reason, a system reset is generated.
		WDT: During HALT mode (Old) WDT operates in HALT mode. HALT mode is therefore cleared by an NMI or reset if it continues for more than the NMI/reset generation cycle and the NMI or reset handler is executed. (New) WDT operates in HALT mode. HALT mode is therefore cleared by a reset if it continues for more than the reset generation cycle and the reset handler is executed.
		WDT: During SLEEP mode (Old) WDT operates in SLEEP mode if the selected clock source is running. In this case SLEEP mode is cleared by an NMI or reset if it continues for more than the NMI/reset generation cycle and the NMI or reset handler is executed. ... If the clock source stops in SLEEP mode, WDT stops. To prevent generation of an unnecessary NMI or reset after clearing SLEEP mode, reset WDT before executing the slp instruction. (New) WDT operates in SLEEP mode if the selected clock source is running. In this case SLEEP mode is cleared by a reset if it continues for more than the reset generation cycle and the reset handler is executed. ... If the clock source stops in SLEEP mode, WDT stops. To prevent generation of an unnecessary reset after clearing SLEEP mode, reset WDT before executing the slp instruction.
	7-3	WDT: WDT Control Register Modified the register table (NMIXRST, STATNMI → Reserved)
		WDT: WDT Control Register (Old) Bits 15–10 Reserved Bit 9 N MIXRST ... Bit 8 STATNMI ... Bits 7–5 Reserved (New) Bits 15–5 Reserved
	7-4	WDT: WDT Control Register - Bits 3–0 WDTRUN[3:0] (Old) Since an NMI or reset may be generated immediately after running depending on the counter value, WDT should also be reset concurrently when running WDT. (New) Since a reset may be generated immediately after running depending on the counter value, WDT should also be reset concurrently when running WDT.
	8-3	SVD: Starting detection (Old) 3. Set the following SVDCTL register bits: ... - SVDCTL.SVDC[4:0] bits (Set comparison voltage) (New) 3. Set the following SVDCTL register bits: ... - VDCTL.SVDC[4:0] bits (Set SVD detection voltage V_{SVD})

REVISION HISTORY

Code No.	Page	Contents
412486301	8-3	Reading detection results (Old) • Power supply voltage (V_{DD} or EXSVD) ≥ Comparison voltage when SVDINTF.SVDDT bit = 0 • Power supply voltage (V_{DD} or EXSVD) < Comparison voltage when SVDINTF.SVDDT bit = 1 ... After the SVDCTL.SVDC[4:0] bits setting value is altered to change the comparison voltage when the SVDCTL.MODEN bit = 1, wait for at least SVD circuit response time before reading the SVDINTF.SVDDT bit ... (New) • Power supply voltage (V_{DD} or EXSVD) ≥ SVD detection voltage V_{SVD} when SVDINTF.SVDDT bit = 0 • Power supply voltage (V_{DD} or EXSVD) < SVD detection voltage V_{SVD} when SVDINTF.SVDDT bit = 1 ... After the SVDCTL.SVDC[4:0] bits setting value is altered to change the SVD detection voltage V_{SVD} when the SVDCTL.MODEN bit = 1, wait for at least SVD circuit response time before reading the SVDINTF.SVDDT bit ...
	8-5	SVD: SVD Interrupt (Old) Once the SVDINTF.SVDIF bit is set, it will not be cleared even if the power supply voltage subsequently returns to a value exceeding the comparison voltage value. (New) Once the SVDINTF.SVDIF bit is set, it will not be cleared even if the power supply voltage subsequently returns to a value exceeding the SVD detection voltage V_{SVD}.
	8-6	SVD: SVD Control Register - Bits 12–8 SVDC[4:0] (Old) These bits select a comparison voltage for detecting low voltage. Table 8.6.3 Comparison Voltage Setting SVDCTL.SVDC[4:0] bits | Comparison voltage [V] (New) These bits select an SVD detection voltage V_{SVD} for detecting low voltage. Table 8.6.3 Setting of SVD Detection Voltage V_{SVD} SVDCTL.SVDC[4:0] bits | SVD detection voltage V_{SVD} [V]
	8-7	SVD: SVD Status and Interrupt Flag Register - Bit 8 SVDDT (Old) 1 (R): Power supply voltage (V_{DD} or EXSVD) < comparison voltage 0 (R): Power supply voltage (V_{DD} or EXSVD) ≥ comparison voltage (New) 1 (R): Power supply voltage (V_{DD} or EXSVD) < SVD detection voltage V_{SVD} 0 (R): Power supply voltage (V_{DD} or EXSVD) ≥ SVD detection voltage V_{SVD}
	8-8	SVD: SVD Interrupt Enable Register - Bit 0 SVDIE (Old) Notes: ... • To prevent generating unnecessary interrupts, clear the corresponding interrupt flag before enabling interrupts. (New) Notes: ... • To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.
	9-1	T16: Figure 9.1.1 Configuration of a T16 Channel Modified the figure (The I/O port (chattering filter) was deleted.)
		T16: Input Pin (Old) If the port is shared with the EXCL*m* pin and other functions, the EXCL*m* input function must be assigned to the port before using the event counter function. The EXCL*m* signal can be input through the chattering filter. For more information, refer to the "I/O Ports" chapter. (New) If the port is shared with the EXCL*m* pin and other functions, the EXCL*m* input function must be assigned to the port before using the event counter function. For more information, refer to the "I/O Ports" chapter.
	9-6	T16: T16 Ch.*n* Reload Data Register (Old) Note: The T16_*n*TR register cannot be altered while the timer is running (T16_*n*CTL.PRUN bit = 1), as an incorrect initial value may be preset to the counter. (New) Notes: • The T16_*n*TR register cannot be altered while the timer is running (T16_*n*CTL.PRUN bit = 1), as an incorrect initial value may be preset to the counter. • When one-shot mode is set, the T16_*n*TR.TR[15:0] bits should be set to a value equal to or greater than 0x0001.
	9-7	T16: T16 Ch.*n* Interrupt Enable Register - Bit 0 UFIE (Old) Note: To prevent generating unnecessary interrupts, clear the corresponding interrupt flag before enabling interrupts. (New) Note: To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.
	12-14	I2C: Figure 12.4.7.1 Example of Data Transfer Starting Operations in 10-bit Address Mode (Slave Mode) (Old) Operations by I2C (master mode) Operations by the external slave (New) Operations by the external master Operations by I2C (slave mode)
	AP-A-3	List of Peripheral Circuit Control Registers: WDT Control Register Modified the register table (NMIXRST, STATNMI → Reserved)
	AP-C-1	Mounting Precautions: V_{PP} pin (Old) No description (New) If fluctuations in the Flash programming voltage V_{PP} is large, connect a capacitor C_{VPP} between the V_{SS} and V_{PP} pins to suppress fluctuations within $V_{PP} \pm 1$ V. The C_{VPP} should be placed as close to the V_{PP} pin as possible and use a sufficiently thick wiring pattern that allows current of several tens of mA to flow. Added a figure (V_{PP} pin connection example)

EPSON

International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.
214 Devcon Drive,
San Jose, CA 95112, USA
Phone: +1-800-228-3964 Fax: +1-408-922-0238

EUROPE

EPSON EUROPE ELECTRONICS GmbH
Riesstrasse 15, 80992 Munich,
GERMANY
Phone: +49-89-14005-0 Fax: +49-89-14005-110

ASIA

EPSON (CHINA) CO., LTD.
7F, Jinbao Bldg., No.89 Jinbao St.,
Dongcheng District,
Beijing 100005, CHINA
Phone: +86-10-8522-1199 Fax: +86-10-8522-1125

SHANGHAI BRANCH
7F, Block B, Hi-Tech Bldg., 900 Yishan Road,
Shanghai 200233, CHINA
Phone: +86-21-5423-5577 Fax: +86-21-5423-4677

SHENZHEN BRANCH
12F, Dawning Mansion, Keji South 12th Road,
Hi-Tech Park, Shenzhen 518057, CHINA
Phone: +86-755-2699-3828 Fax: +86-755-2699-3838

EPSON HONG KONG LTD.
Unit 715-723, 7/F Trade Square, 681 Cheung Sha Wan Road,
Kowloon, Hong Kong
Phone: +852-2585-4600 Fax: +852-2827-4346

EPSON TAIWAN TECHNOLOGY & TRADING LTD.
14F, No. 7, Song Ren Road,
Taipei 110, TAIWAN
Phone: +886-2-8786-6688 Fax: +886-2-8786-6660

EPSON SINGAPORE PTE., LTD.
1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 Fax: +65-6271-3182

SEIKO EPSON CORP.
KOREA OFFICE
5F, KLI 63 Bldg., 60 Yoido-dong,
Youngdeungpo-Ku, Seoul 150-763, KOREA
Phone: +82-2-784-6027 Fax: +82-2-767-3677

SEIKO EPSON CORP.
MICRODEVICES OPERATIONS DIVISION

IC Sales & Marketing Department
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 Fax: +81-42-587-5117

Document Code: 412486301
First Issue January 2013
Revised December 2014 in JAPAN Ⓛ