EPSON

CMOS 16-BIT SINGLE CHIP MICROCONTROLLER

S1C17W15
Technical Manual

SEIKO EPSON CORPORATION

Rev. 1.4

NOTICE: PLEASE READ THE FOLLOWING NOTICE CAREFULLY BEFORE USING THIS DOCUMENT

The contents of this document are subject to change without notice.

1. This document may not be copied, reproduced, or used for any other purpose, in whole or in part, without the consent of the Seiko Epson Corporation (“Epson”).
2. Before purchasing or using Epson products, please contact our sales representative for the latest information and always be sure to check the latest information published on Epson’s official web sites and other sources.
3. Information provided in this document such as application circuits, programs, usage, etc., are for reference purposes only. Using the application circuits, programs, usage, etc. in the design of your equipment or systems is your own responsibility. Epson makes no guarantees against any infringements or damages to any third parties’ intellectual property rights or any other rights resulting from the information. This document does not grant you any licenses, intellectual property rights or any other rights with respect to Epson products owned by Epson or any third parties.
4. Epson is committed to constantly improving quality and reliability, but semiconductor products in general are subject to malfunction and failure. By using Epson products, you shall be responsible for your hardware. Software and systems must be designed well enough to prevent death or injury as well as any property damage even if any of the malfunctions or failures might be caused by Epson products. When designing your products using Epson products, please be sure to check and comply with the latest information regarding Epson products (this document, specifications, data sheets, manuals, Epson’s web site, etc.). When using the information included above materials such as product data, charts, technical contents, programs, algorithms and application circuit examples, you shall evaluate your products both on a stand-alone basis as well as within your overall systems. You shall be solely responsible for deciding whether or not to adopt and use Epson products.
5. Epson has prepared this document and programs provided in this document carefully to be accurate and dependable, but Epson does not guarantee that the information and the programs are always accurate and complete. Epson assumes no responsibility for any damages which you incur due to misinformation in this document and the programs.
6. No dismantling, analysis, reverse engineering, modification, alteration, adaptation, reproduction, etc., of Epson products is allowed.
7. Epson products have been designed, developed and manufactured to be used in general electronic applications (office equipment, communications equipment, measuring instruments, home electronics, etc.) (“General Purpose”) and applications which is individually listed in this document or designated by Epson (“Designated Purpose”). Epson products are NOT intended for any use beyond the General Purpose and Designated Purpose uses that requires particular/higher quality or reliability in order to refrain from causing any malfunction or failure leading to death, injury, serious property damage or severe impact on society, including, but not limited to those listed below (“Particular Purpose”). Therefore, you are advised to use Epson products only for General Purpose and Designated Purpose uses. Should you desire to buy and use Epson products for a Particular Purpose, Epson makes no warranty and disclaims with respect to Epson products, whether express or implied, including without limitation any implied warranty of merchantability or fitness for any Particular Purpose. Please be sure to contact our sales representative and obtain approval in advance.

   [Examples of Particular Purpose]
   Space equipment (artificial satellites, rockets, etc.) /
   Transportation vehicles and their control equipment (automobiles, aircraft, trains, ships, etc.) /
   Medical equipment / Relay equipment to be placed on ocean floor /
   Power station control equipment / Disaster or crime prevention equipment / Traffic control equipment / Financial equipment

   Other applications requiring similar levels of reliability as those listed above. Please be sure to contact our sales representative for details of the other applications.
8. Epson products listed in this document and our associated technologies shall not be used in any equipment or systems that laws and regulations in Japan or any other countries prohibit to manufacture, use or sell. Furthermore, Epson products and our associated technologies shall not be used for developing weapons of mass destruction, or any other military purposes or applications. If exporting Epson products or our associated technologies, you shall comply with the Foreign Exchange and Foreign Trade Control Act in Japan, Export Administration Regulations in the U.S.A. (EAR) and other export-related laws and regulations in Japan and any other countries and follow the required procedures as provided by the relevant laws and regulations.
9. Epson assumes no responsibility for any damages (whether direct or indirect) caused by or in relation with your non-compliance with the terms and conditions in this document.
10. Epson assumes no responsibility for any damages (whether direct or indirect) incurred by any third party that you assign, transfer, loan, etc., Epson products to.
11. For more details or other concerns about this document, please contact our sales representative.
12. Company names and product names listed in this document are trademarks or registered trademarks of their respective companies.

Evaluation board/kit and Development tool important notice

1. Epson evaluation board/kit or development tool is designed for use for engineering evaluation, demonstration, or development purposes only. Do not use it for other purposes. It is not intended to meet the requirements of design for finished products.
2. Epson evaluation board/kit or development tool is intended for use by an electronic engineer and is not a consumer product. The user should use it properly and in a safe manner. Epson dose not assume any responsibility or liability of any kind of damage and/or fire coursed by the use of it. The user should cease to use it when any abnormal issue occurs even during proper and safe use.
3. The part used for Epson evaluation board/kit or development tool may be changed without any notice.

Rev. e1.4, 2023. 4

Preface

This is a technical manual for designers and programmers who develop a product using the S1C17W15. 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 –

1 Overview

The S1C17W15 is a 16-bit MCU that features low-voltage operation from 1.2 V even though Flash memory is included. The embedded high-efficiency DC-DC converter generates the constant-voltage to drive the IC with lower power consumption than 4-bit MCUs. This IC includes a real-time clock, a stopwatch, an LCD driver, and a PWM timer capable of being used to generate drive waveforms for a motor driver as well as a high-performance 16-bit CPU. It is suitable for battery-driven applications that require an LCD display and timers.

1.1 Features

Table 1.1.1 Features

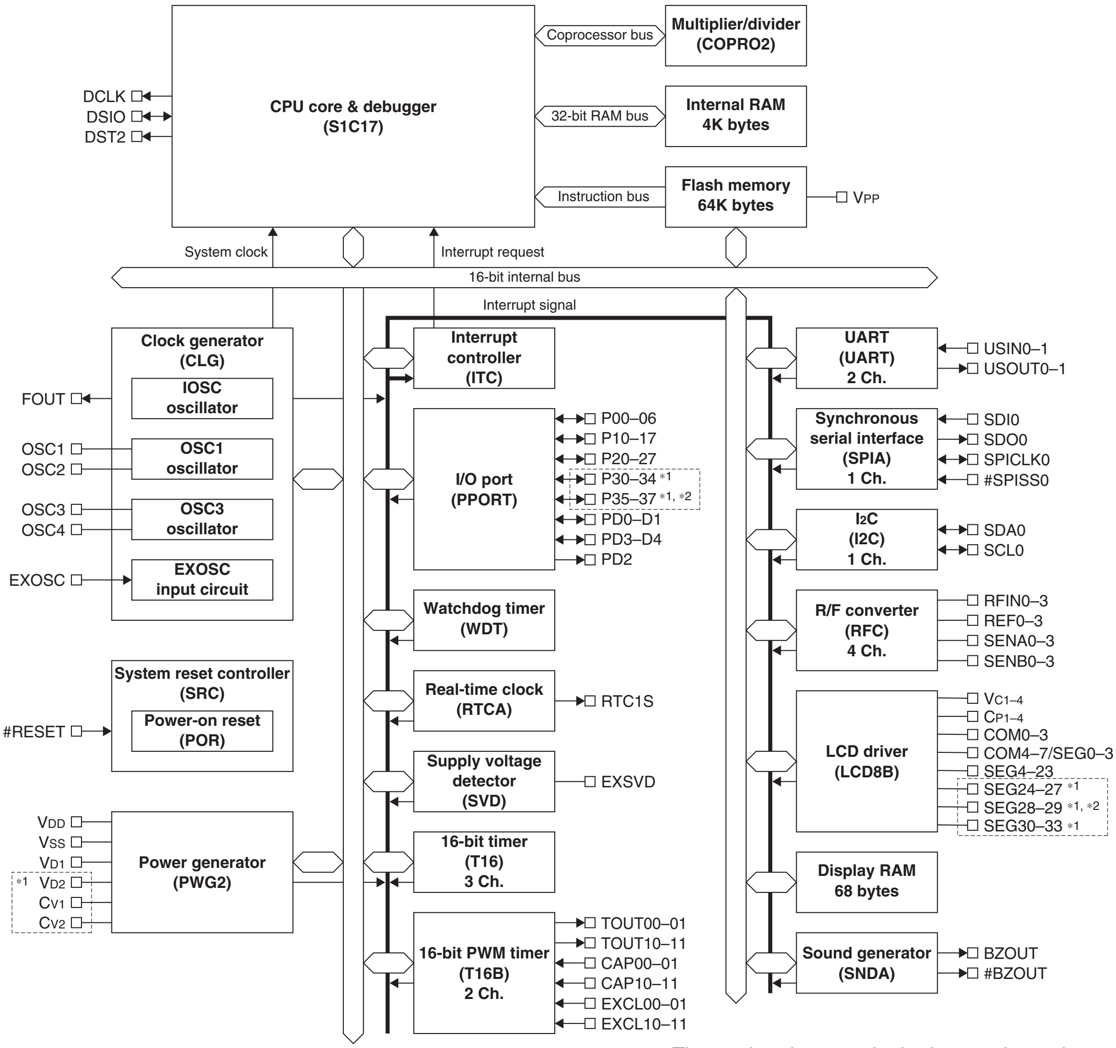
Model	S1C17W15
CPU	
CPU core	Seiko Epson original 16-bit RISC CPU core S1C17
Other	On-chip debugger
Embedded Flash memory	
Capacity	64K bytes (for both instructions and data)
Erase/program count	50 times (min.) * Programming by the debugging tool ICDmini
Other	Security function to protect from reading/programming by ICDmini
	On-board programming function using ICDmini
Embedded RAM	
Capacity	4K bytes
Embedded display RAM	
Capacity	68 bytes
Clock generator (CLG)	
System clock source	4 sources (IOSC/OSC1/OSC3/EXOSC)
System clock frequency (operating frequency)	1.1 MHz (max.) V_{DD} = 1.2 to 1.6 V 4.2 MHz (max.) V_{DD} = 1.6 to 3.6 V
IOSC oscillator circuit (boot clock source)	700 kHz (typ.) embedded oscillator
	23 µs (max.) starting time (time from cancelation of SLEEP state to vector table read by the CPU)
OSC1 oscillator circuit	32.768 kHz (typ.) crystal oscillator
	Oscillation stop detection circuit included
OSC3 oscillator circuit	4.2 MHz (max.) crystal/ceramic oscillator
	500 kHz, 1, 2, and 4 MHz-switchable embedded oscillator
	2.1 MHz (max.) CR oscillator (an external R is required)
EXOSC clock input	4.2 MHz (max.) square or sine wave input
Other	Configurable system clock division ratio
	Configurable system clock 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	Input/output port: 35 bits (max., 100-pin package or chip) 32 bits (max., 80-pin package) 27 bits (max., 64-pin package)
	Output port: 1 bit (max.)
	Pins are shared with the peripheral I/O.
Number of input interrupt ports	31 bits (max., 100-pin package or chip) 28 bits (max., 80-pin package) 23 bits (max., 64-pin package)
Number of ports that support universal port multiplexer (UPMUX)	23 bits
	A peripheral circuit I/O function selected via software can be assigned to each port.
Timers	
Watchdog timer (WDT)	Generates watchdog timer reset.
Real-time clock (RTCA)	128–1 Hz counter, second/minute/hour/day/day of the week/month/year counters
	Theoretical regulation function for 1-second correction
	Alarm and stopwatch functions
16-bit timer (T16)	3 channels
	Generates the SPIA master clock.
16-bit PWM timer (T16B)	2 channels
	Event counter/capture function
	PWM waveform generation function
	Number of PWM output or capture input ports: 2 ports/channel

Supply voltage detector (SVD)	
Detection level	30 levels (1.2 to 3.6 V)
Detection accuracy	±3 %
Other	Intermittent operation mode
	Generates an interrupt or reset according to the detection level evaluation.
Serial interfaces	
UART (UART)	2 channels
	Baud-rate generator included, IrDA1.0 supported
Synchronous serial interface (SPIA)	1 channel
	2 to 16-bit variable data length
	The 16-bit timer (T16) can be used for the baud-rate generator in master mode.
I^2C (I2C) *1	1 channel
	Baud-rate generator included
Sound generator (SNDA)	
Buzzer output function	512 Hz to 16 kHz output frequencies
	One-shot output function
Melody generation function	Pitch: 128 Hz to 16 kHz ≈ C3 to C6
	Duration: 7 notes/rests (Half note/rest to thirty-second note/rest)
	Tempo: 16 tempos (30 to 480)
	Tie may be specified.
LCD driver (LCD8B)	
LCD output	30 SEG × 5–8 COM (max.), 34 SEG × 1–4 COM (max.) (100-pin package or chip) 28 SEG × 5–8 COM (max.), 32 SEG × 1–4 COM (max.) (80-pin package) 20 SEG × 5–8 COM (max.), 24 SEG × 1–4 COM (max.) (64-pin package)
LCD contrast	32 levels
Other	1/4 or 1/3 bias power supply included, external voltage can be applied.
R/F converter (RFC)	
Conversion method	CR oscillation type with 24-bit counters
Number of conversion channels	4 channels (Up to two sensors can be connected to each channel.)
Supported sensors	DC-bias resistive sensors, AC-bias resistive sensors (Ch.0 only)
Multiplier/divider (COPRO2)	
Arithmetic functions	16-bit × 16-bit multiplier
	16-bit × 16-bit + 32-bit multiply and accumulation unit
	32-bit ÷ 32-bit divider
Reset	
#RESET pin	Reset when the reset pin is set to low.
Power-on reset	Reset at power on.
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: 20 systems (8 levels)
Power supply voltage	
V_{DD} operating voltage	1.2 to 3.6 V
V_{DD} operating voltage for Flash programming	1.8 to 3.6 V (V_{PP} = 7.5 V external power supply is required.)
V_{DD} operating voltage for super economy mode	2.5 to 3.6 V (100-pin/80-pin package or chip)
Operating temperature	
Operating temperature range	-40 to 85 °C
Current consumption	
SLEEP mode *2	0.15 µA IOSC = OFF, OSC1 = OFF, OSC3 = OFF
HALT mode	0.5 µA OSC1 = 32 kHz, RTC = ON
	0.3 µA (100-pin/80-pin package or chip) OSC1 = 32 kHz, RTC = ON, super economy mode
	1.2 µA (100-pin/80-pin package or chip) OSC1 = 32 kHz, RTC = ON, CPU = OSC1, LCD = ON (no panel load, V_{C2} reference, 1/3 bias, all on), super economy mode

Current consumption	
RUN mode	8 µA OSC1 = 32 kHz, RTC = ON, CPU = OSC1
	4 µA (100-pin/80-pin package or chip) OSC1 = 32 kHz, RTC = ON, CPU = OSC1, super economy mode
	250 µA OSC3 = 1 MHz (ceramic oscillator), OSC1 = 32 kHz, RTC = ON, CPU = OSC3
Shipping form	
1 *3	SQFN9-64PIN (P-VQFN064-0909-0.50, 9 × 9 mm, t = 1 mm, 0.5 mm pitch)
2 *3	TQFP13-64PIN (P-TQFP064-1010-0.50, 10 × 10 mm, t = 1.2 mm, 0.5 mm pitch)
3 *3	QFP14-80PIN (P-LQFP080-1212-0.50, 12 × 12 mm, t = 1.7 mm, 0.5 mm pitch)
4 *3	QFP15-100PIN (P-LQFP100-1414-0.50, 14 × 14 mm, t = 1.7 mm, 0.5 mm pitch)
5	Chip (Die form), Pad pitch: 80 µm (min.)

*1 The input filter in I2C (SDA and SCL inputs) does not comply with the standard for removing noise spikes less than 50 ns.
*2 The RAM retains data even in SLEEP mode.
*3 Shown in parentheses are JEITA package names.

1.2 Block Diagram



*1 These pins do not exist in the 64-pin package.
*2 These pins do not exist in the 80-pin package.

Figure 1.2.1 S1C17W15 Block Diagram

1.3 Pins

1.3.1 Pin Configuration Diagram (Package)

SQFN9-64PIN

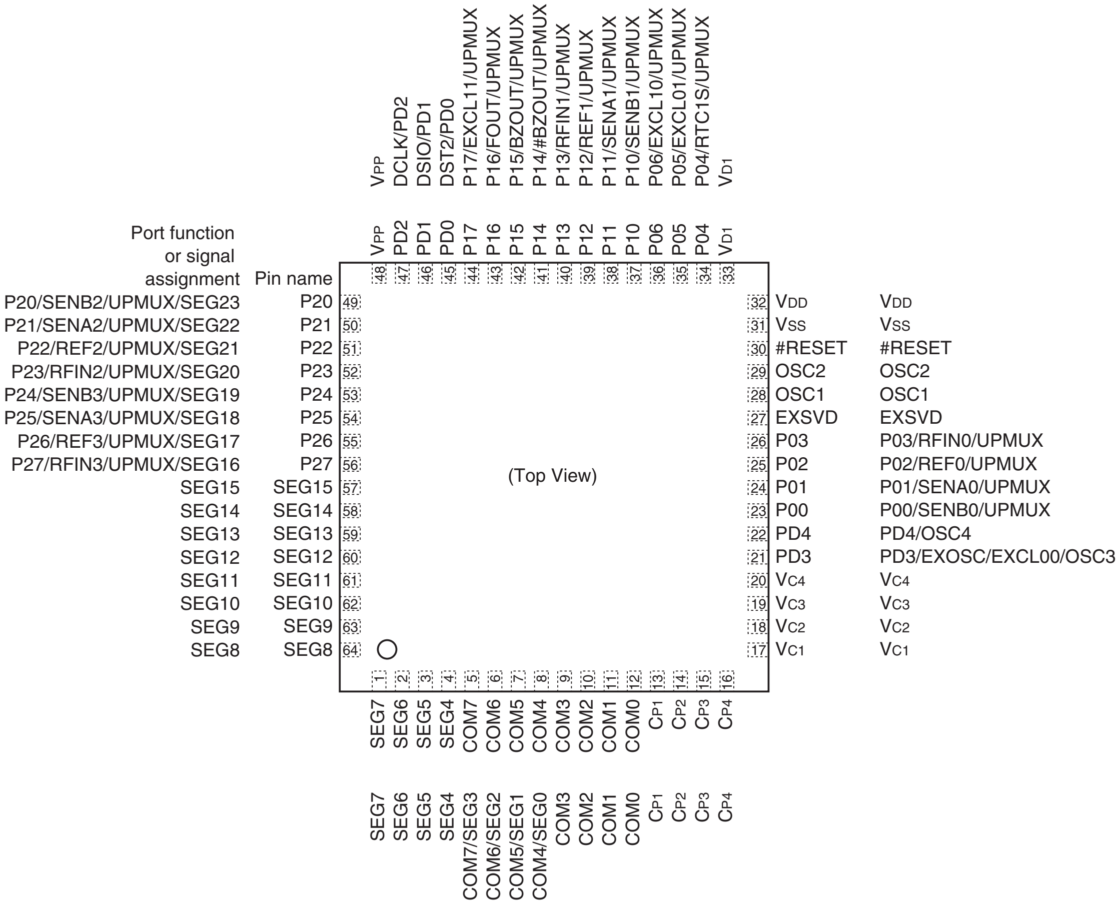


Figure 1.3.1.1 S1C17W15 Pin Configuration Diagram (SQFN9-64PIN)

TQFP13-64PIN

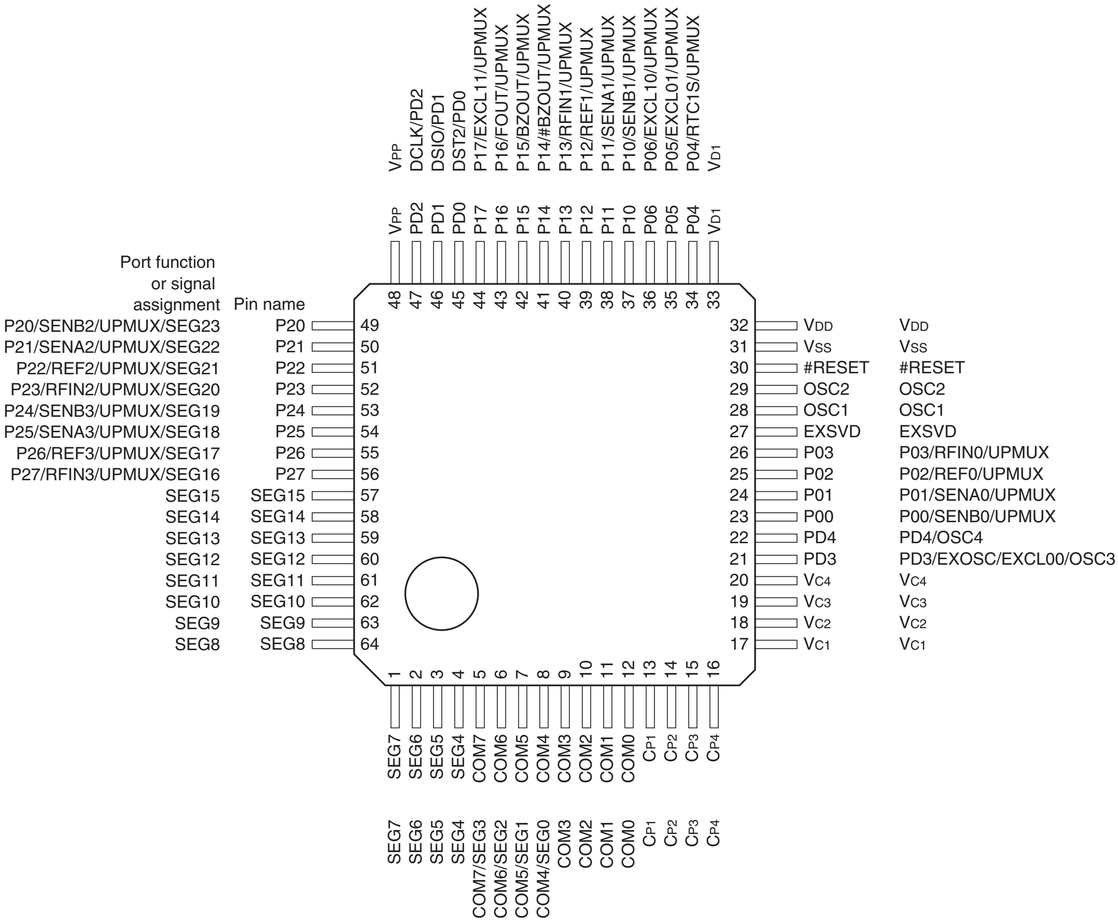


Figure 1.3.1.2 S1C17W15 Pin Configuration Diagram (TQFP13-64PIN)

QFP14-80PIN

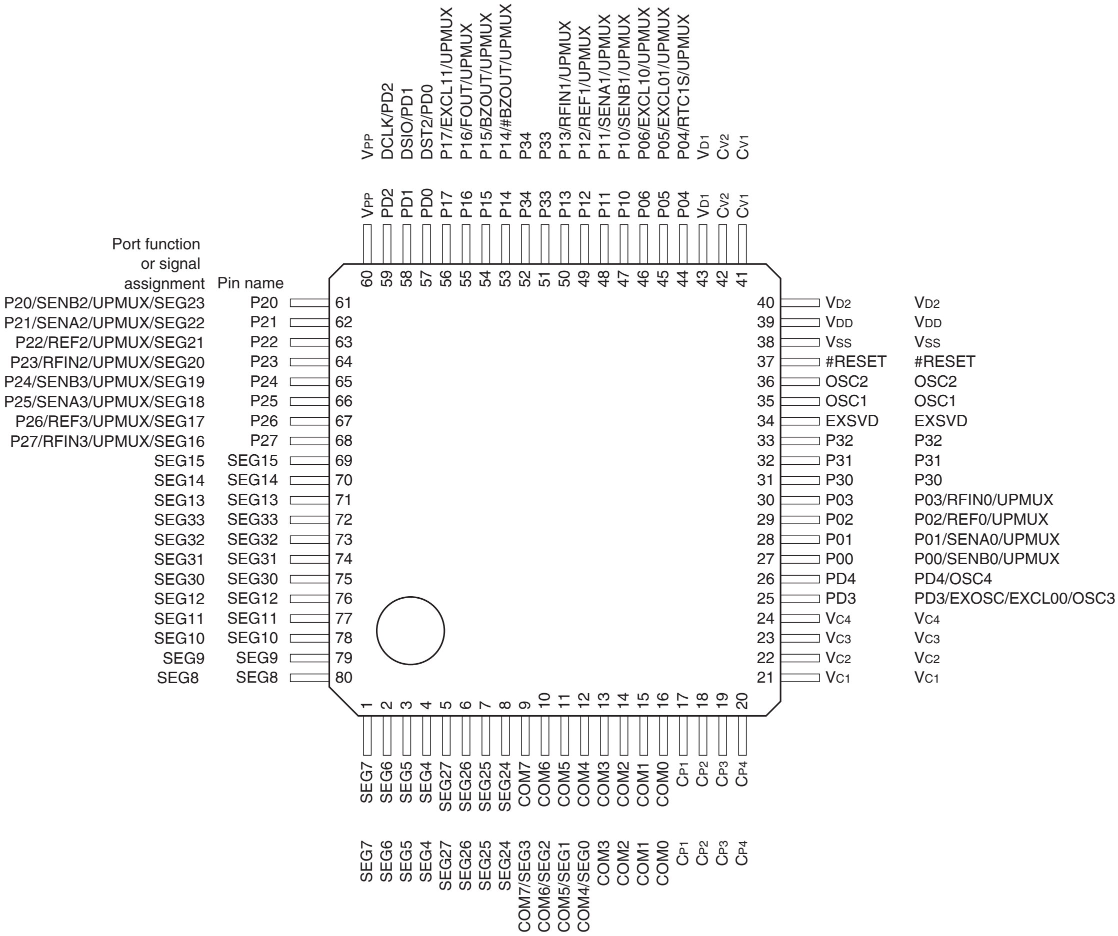


Figure 1.3.1.3 S1C17W15 Pin Configuration Diagram (QFP14-80PIN)

QFP15-100PIN

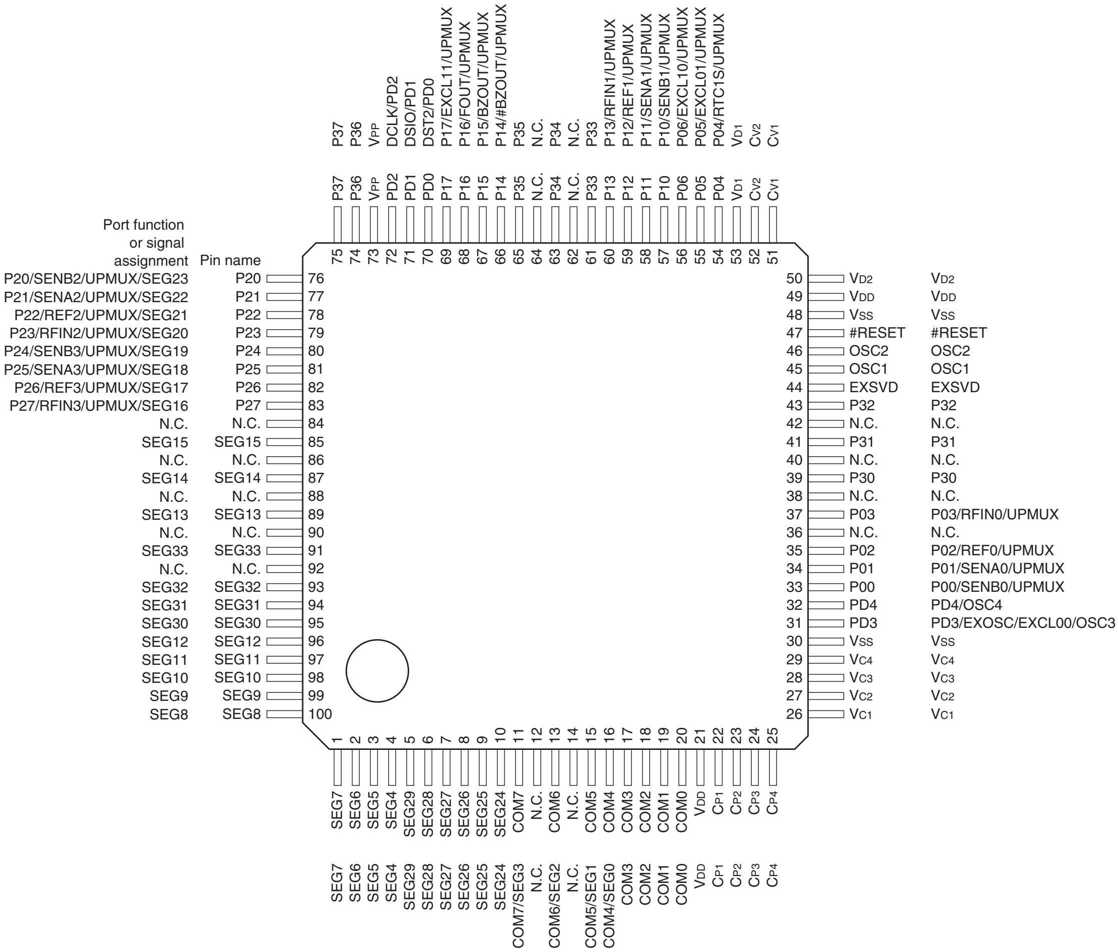


Figure 1.3.1.4 S1C17W15 Pin Configuration Diagram (QFP15-100PIN)

1.3.2 Pad Configuration Diagram (Chip)

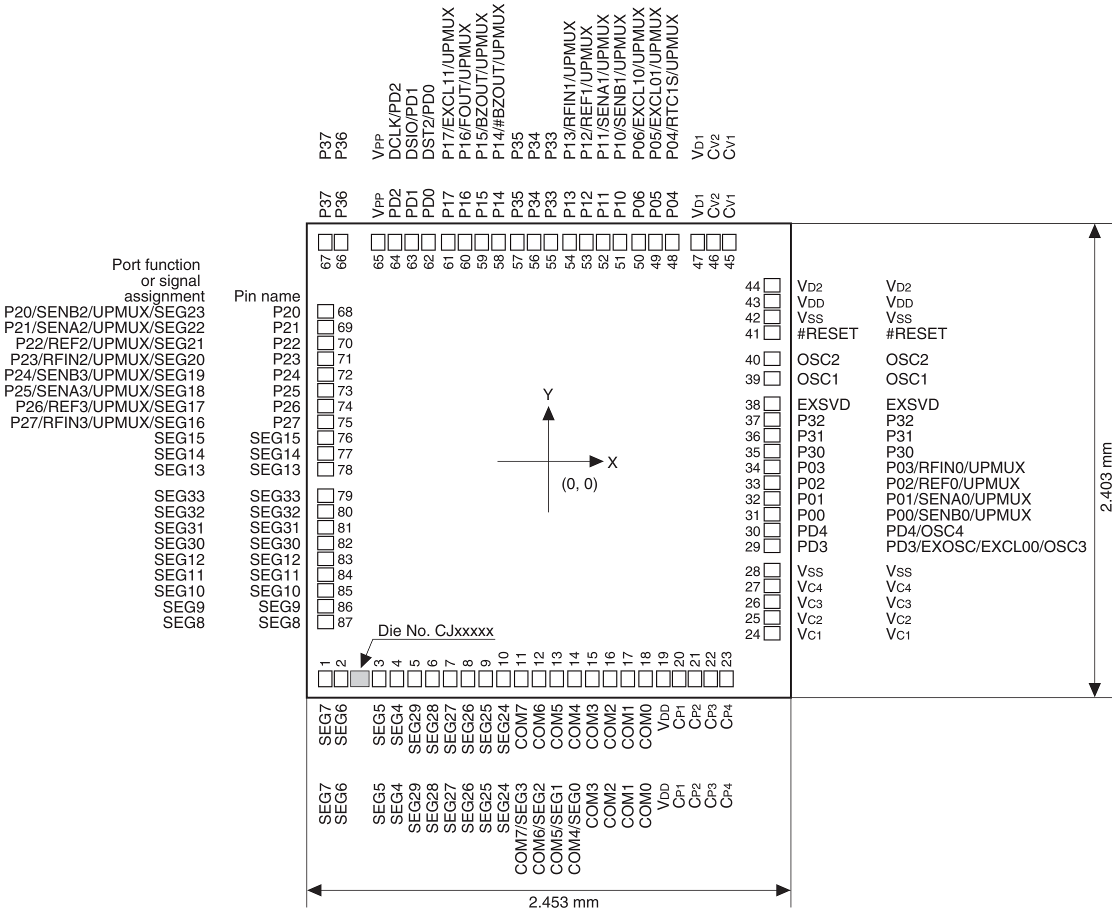


Figure 1.3.2.1 S1C17W15 Pad Configuration Diagram (Chip)

Pad opening No. 1–23, 45–67: X = 68 µm, Y = 80 µm
No. 24–44, 68–87: X = 80 µm, Y = 68 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	-1,133	-1,106	24	1,131	-875	45	914	1,106	68	-1,131	756
2	-1,053	-1,106	25	1,131	-795	46	834	1,106	69	-1,131	676
3	-860	-1,106	26	1,131	-715	47	754	1,106	70	-1,131	596
4	-770	-1,106	27	1,131	-635	48	625	1,106	71	-1,131	516
5	-680	-1,106	28	1,131	-555	49	540	1,106	72	-1,131	436
6	-590	-1,106	29	1,131	-432	50	455	1,106	73	-1,131	356
7	-500	-1,106	30	1,131	-352	51	360	1,106	74	-1,131	276
8	-410	-1,106	31	1,131	-272	52	275	1,106	75	-1,131	196
9	-320	-1,106	32	1,131	-192	53	190	1,106	76	-1,131	116
10	-230	-1,106	33	1,131	-112	54	105	1,106	77	-1,131	36
11	-140	-1,106	34	1,131	-32	55	10	1,106	78	-1,131	-44
12	-50	-1,106	35	1,131	48	56	-75	1,106	79	-1,131	-178
13	40	-1,106	36	1,131	128	57	-160	1,106	80	-1,131	-258
14	130	-1,106	37	1,131	208	58	-255	1,106	81	-1,131	-338
15	220	-1,106	38	1,131	288	59	-340	1,106	82	-1,131	-418
16	310	-1,106	39	1,131	416	60	-425	1,106	83	-1,131	-498
17	400	-1,106	40	1,131	516	61	-510	1,106	84	-1,131	-578
18	490	-1,106	41	1,131	648	62	-610	1,106	85	-1,131	-658
19	580	-1,106	42	1,131	728	63	-695	1,106	86	-1,131	-738
20	660	-1,106	43	1,131	808	64	-780	1,106	87	-1,131	-818
21	740	-1,106	44	1,131	888	65	-865	1,106	–	–	–
22	820	-1,106	–	–	–	66	-1,053	1,106	–	–	–
23	900	-1,106	–	–	–	67	-1,133	1,106	–	–	–

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 = Input without pulled up/down
- 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

Tolerant fail-safe structure:
- ✓ = Over voltage tolerant fail-safe type I/O cell included (see the “I/O Ports” chapter)

Table 1.3.3.1 Pin Description

Pin/pad name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function	Package		
						64pin	80pin	100pin/chip
V_{DD}	V_{DD}	P	–	–	Power supply (+)	✓	✓	✓
V_{SS}	V_{SS}	P	–	–	GND	✓	✓	✓
V_{PP}	V_{PP}	P	–	–	Power supply for Flash programming	✓	✓	✓
V_{D1}	V_{D1}	A	–	–	DC-DC converter output	✓	✓	✓
V_{D2}	V_{D2}	A	–	–	DC-DC converter stabilization capacitor connect pin	–	✓	✓
C_{V1-2}	C_{V1-2}	A	–	–	DC-DC converter charge pump capacitor connect pins	–	✓	✓
V_{C1-4}	V_{C1-4}	P	–	–	LCD panel driver power supply	✓	✓	✓
C_{P1-4}	C_{P1-4}	A	–	–	LCD power supply booster capacitor connect pins	✓	✓	✓
OSC1	OSC1	A	-	-	OSC1 oscillator circuit input	✓	✓	✓
OSC2	OSC2	A	-	-	OSC1 oscillator circuit output	✓	✓	✓
#RESET	#RESET	I	I (Pull-up)	–	Reset input	✓	✓	✓
P00	P00	I/O	Hi-Z	–	I/O port	✓	✓	✓
	SENB0	A			R/F converter Ch.0 sensor B oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P01	P01	I/O	Hi-Z	–	I/O port	✓	✓	✓
	SENA0	A			R/F converter Ch.0 sensor A oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P02	P02	I/O	Hi-Z	–	I/O port	✓	✓	✓
	REF0	A			R/F converter Ch.0 reference oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P03	P03	I/O	Hi-Z	–	I/O port	✓	✓	✓
	RFIN0	A			R/F converter Ch.0 oscillation input	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P04	P04	I/O	Hi-Z	–	I/O port	✓	✓	✓
	RTC1S	O			Real-time clock 1-second cycle pulse output	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P05	P05	I/O	Hi-Z	–	I/O port	✓	✓	✓
	EXCL01	I			16-bit PWM timer Ch.0 event counter input 1	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P06	P06	I/O	Hi-Z	–	I/O port	✓	✓	✓
	EXCL10	I			16-bit PWM timer Ch.1 event counter input 0	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P10	P10	I/O	Hi-Z	–	I/O port	✓	✓	✓
	SENB1	A			R/F converter Ch.1 sensor B oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P11	P11	I/O	Hi-Z	–	I/O port	✓	✓	✓
	SENA1	A			R/F converter Ch.1 sensor A oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓

Pin/pad name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function	Package 64pin	80pin	100pin/ chip
P12	P12	I/O	Hi-Z	–	I/O port	✓	✓	✓
	REF1	A			R/F converter Ch.1 reference oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P13	P13	I/O	Hi-Z	–	I/O port	✓	✓	✓
	RFIN1	A			R/F converter Ch.1 oscillation input	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P14	P14	I/O	Hi-Z	–	I/O port	✓	✓	✓
	#BZOUT	O			Sound generator inverted output	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P15	P15	I/O	Hi-Z	–	I/O port	✓	✓	✓
	BZOUT	O			Sound generator output	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P16	P16	I/O	Hi-Z	–	I/O port	✓	✓	✓
	FOUT	O			Clock external output	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P17	P17	I/O	Hi-Z	–	I/O port	✓	✓	✓
	EXCL11	I			16-bit PWM timer Ch.1 event counter input 1	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
P20	P20	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	SENB2	A			R/F converter Ch.2 sensor B oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG23	A			LCD segment output	✓	✓	✓
P21	P21	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	SENA2	A			R/F converter Ch.2 sensor A oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG22	A			LCD segment output	✓	✓	✓
P22	P22	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	REF2	A			R/F converter Ch.2 reference oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG21	A			LCD segment output	✓	✓	✓
P23	P23	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	RFIN2	A			R/F converter Ch.2 oscillation input	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG20	A			LCD segment output	✓	✓	✓
P24	P24	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	SENB3	A			R/F converter Ch.3 sensor B oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG19	A			LCD segment output	✓	✓	✓
P25	P25	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	SENA3	A			R/F converter Ch.3 sensor A oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG18	A			LCD segment output	✓	✓	✓
P26	P26	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	REF3	A			R/F converter Ch.3 reference oscillator pin	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG17	A			LCD segment output	✓	✓	✓
P27	P27	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	RFIN3	A			R/F converter Ch.3 oscillation input	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)	✓	✓	✓
	SEG16	A			LCD segment output	✓	✓	✓
P30	P30	I/O	Hi-Z	–	I/O port	–	✓	✓
P31	P31	I/O	Hi-Z	–	I/O port	–	✓	✓
P32	P32	I/O	Hi-Z	–	I/O port	–	✓	✓
P33	P33	I/O	Hi-Z	–	I/O port	–	✓	✓
P34	P34	I/O	Hi-Z	–	I/O port	–	✓	✓
P35	P35	I/O	Hi-Z	–	I/O port	–	–	✓
P36	P36	I/O	Hi-Z	–	I/O port	–	–	✓
P37	P37	I/O	Hi-Z	–	I/O port	–	–	✓
PD0	DST2	O	O (L)	–	On-chip debugger status output	✓	✓	✓
	PD0	I/O			I/O port	✓	✓	✓
PD1	DSIO	I/O	I (Pull-up)	–	On-chip debugger data input/output	✓	✓	✓
	PD1	I/O			I/O port	✓	✓	✓

Pin/pad name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function	Package 64pin	Package 80pin	Package 100pin/chip
PD2	DCLK	O	O (H)	–	On-chip debugger clock output	✓	✓	✓
	PD2	O			Output port	✓	✓	✓
PD3	PD3	I/O	Hi-Z	–	I/O port	✓	✓	✓
	EXOSC	I			Clock generator external clock input	✓	✓	✓
	EXCL00	I			16-bit PWM timer Ch.0 event counter input 0	✓	✓	✓
	OSC3	A			OSC3 oscillator circuit input	✓	✓	✓
PD4	PD4	I/O	Hi-Z	–	I/O port	✓	✓	✓
	OSC4	A			OSC3 oscillator circuit output	✓	✓	✓
COM0–3	COM0–3	A	Hi-Z	–	LCD common output	✓	✓	✓
COM4	COM4	A	Hi-Z	–	LCD common output	✓	✓	✓
	SEG0	A			LCD segment output	✓	✓	✓
COM5	COM5	A	Hi-Z	–	LCD common output	✓	✓	✓
	SEG1	A			LCD segment output	✓	✓	✓
COM6	COM6	A	Hi-Z	–	LCD common output	✓	✓	✓
	SEG2	A			LCD segment output	✓	✓	✓
COM7	COM7	A	Hi-Z	–	LCD common output	✓	✓	✓
	SEG3	A			LCD segment output	✓	✓	✓
SEG4–15	SEG4–15	A	Hi-Z	–	LCD segment output	✓	✓	✓
SEG24–27	SEG24–27	A	Hi-Z	–	LCD segment output	–	✓	✓
SEG28–29	SEG28–29	A	Hi-Z	–	LCD segment output	–	–	✓
SEG30–33	SEG30–33	A	Hi-Z	–	LCD segment output	–	✓	✓
EXSVD	EXSVD	A	I	–	External power supply voltage detection input	✓	✓	✓

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

Universal port multiplexer (UPMUX)

The universal port multiplexer (UPMUX) allows software to select the peripheral circuit input/output function to be assigned to each pin from those listed below.

Table 1.3.3.2 Peripheral Circuit Input/output Function Selectable by UPMUX

Peripheral circuit	Signal to be assigned	I/O	Channel number *n*	Function
Synchronous serial interface (SPIA)	SDI*n*	I	*n* = 0	SPIA Ch.*n* data input
	SDO*n*	O		SPIA Ch.*n* data output
	SPICLK*n*	I/O		SPIA Ch.*n* clock input/output
	#SPISS*n*	I		SPIA Ch.*n* slave-select input
I^2C (I2C)	SCL*n*	I/O	*n* = 0	I2C Ch.*n* clock input/output
	SDA*n*	I/O		I2C Ch.*n* data input/output
UART (UART)	USIN*n*	I	*n* = 0, 1	UART Ch.*n* data input
	USOUT*n*	O		UART Ch.*n* data output
16-bit PWM timer (T16B)	TOUT*n*0/CAP*n*0	I/O	*n* = 0, 1	T16B Ch.*n* PWM output/capture input 0
	TOUT*n*1/CAP*n*1	I/O		T16B Ch.*n* PWM output/capture input 1

Note: Do not assign a function to two or more pins simultaneously.

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.

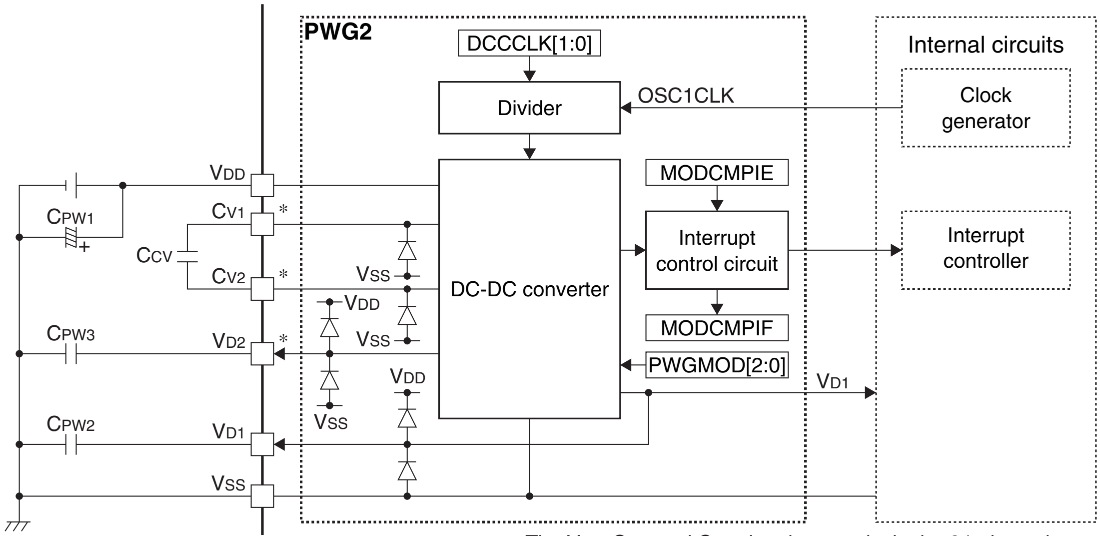
2.1 Power Generator (PWG2)

2.1.1 Overview

PWG2 is the power generator that controls the internal power supply system to drive this IC with stability and low power. The main features of PWG2 are outlined below.

- High-efficiency DC-DC converter for driving internal circuits
- Supports four operating modes including automatic transition to power-saving operations (normal mode, economy mode, automatic mode, and super economy mode).

Figure 2.1.1.1 shows the PWG2 configuration.



∗ The V_{D2}, C_{V1}, and C_{V2} pins do not exist in the 64-pin package.

Figure 2.1.1.1 PWG2 Configuration

Note: The 64-pin package model cannot be placed into super economy mode, as it does not have the V_{D2}, C_{V1}, and C_{V2} pins.

2.1.2 Pins

Table 2.1.2.1 lists the PWG2 pins.

Table 2.1.2.1 List of PWG2 Pins

Pin name	I/O	Initial status	Function
V_{DD}	P	–	Power supply (+)
V_{SS}	P	–	GND
V_{D1}	A	–	DC-DC converter output pin
V_{D2}	A	–	DC-DC converter stabilization capacitor connect pin
C_{V1}	A	–	DC-DC converter charge pump capacitor connect pin
C_{V2}	A	–	DC-DC converter charge pump capacitor connect pin

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.

Note: Be sure to avoid using the V_{D1} and V_{D2} pin outputs for driving external circuits.

2.1.3 Operations

PWG2 provides four operating modes listed in Table 2.1.3.1.

Table 2.1.3.1 PWG2 Operating Mode

Operating mode	Power consumption	Conditions of use
Normal mode	High	None
Automatic mode	↑	None
Economy mode		All the clock sources except for OSC1 are halted (RUN, HALT, or SLEEP mode) or all the clock sources are halted (SLEEP mode).
Super economy mode	↓ Low	1) V_{DD} meets the voltage requirement. *1 2) OSC1 is operating with stability and all other clock sources are halted (when OSC1 is not configured to halt in RUN, HALT, or SLEEP mode).

*1 For the V_{DD} voltage range to set super economy mode, refer to "Recommended Operating Conditions, Power supply voltage V_{DD}" in the "Electrical Characteristics" chapter.

Normal mode

Using this mode results in the highest power consumption within the four operating modes, however, it provides high-stability operations without being affected by voltage fluctuations.

Switching to normal mode from another mode (economy mode)

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Set the PWGCTL.PWGMOD[2:0] bits to 0x2. (Set to normal mode)
3. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Economy mode

PWG2 performs power-saving operations. Power consumption can be reduced in comparison with normal mode. However, this mode can be set only when the system is under light load conditions (see "Condition of use" for economy mode in Table 2.1.3.1) because of its lack of V_{D1} drive capability. Therefore, economy mode does not allow use of high-speed clocks (IOSC, OSC3, and EXOSC).

Switching to economy mode from another mode (normal mode)

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Check to see if the OSC1 oscillation has stabilized (see "Oscillation start procedure for the OSC1 oscillator circuit" in Section 2.3.4.).
3. Stop the high-speed clock sources.
4. Set the PWGCTL.PWGMOD[2:0] bits to 0x3. (Set to economy mode)
5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Note: Be sure to avoid switching to economy mode while a high-speed clock source is operating, as it may cause a malfunction.

Automatic mode

In this mode, the hardware automatically switches between normal mode and economy mode as described above. Use PWG2 in automatic mode when no special control is required.

Switching to automatic mode from another mode

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Set the PWGCTL.PWGMOD[2:0] bits to 0x0. (Set to automatic mode)
3. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

The following shows the conditions for the hardware to switch between normal mode and economy mode and its operations:

1. When all the clock sources except for OSC1 are stopped in normal mode
   After a lapse of 1 ms from stop of the clock source, the hardware switches from normal mode to economy mode and sets the PWGINTF.MODCMPIF bit to 1.
2. When a clock source other than OSC1 is started in economy mode
   The hardware switches to normal mode at the same time the clock source is started.

3. When the slp instruction is executed in normal mode (all the clocks are configured to stop during SLEEP)
   The hardware switches to economy mode at the same time the CPU enters SLEEP mode. The PWGINTF.MODCMPIF bit is not set.
4. When the slp instruction is executed in normal mode (only OSC1 operates during SLEEP)
   After a lapse of 1 ms from transition to SLEEP mode, the hardware switches from normal mode to economy mode and sets the PWGINTF.MODCMPIF bit to 1.

Note: The IC does not enter economy mode if a clock source other than OSC1 is active when the slp instruction is executed. Therefore, stop clock sources other than OSC1 before executing the slp instruction.

5. When the CPU wakes up from SLEEP state
   At the same time the CPU enters RUN mode, the hardware switches to economy mode when OSC1 only is operating or to normal mode in other conditions.

For the PWGINTF.MODCMPIF bit set conditions, refer to “Interrupts.”

Super economy mode

Super economy mode uses a charge pump to generate V_{D1} that is generated by the linear regulator in the three operating modes described above. This achieves more power-saving operation in comparison with economy mode. However, the charge pump operation requires a V_{DD} voltage that exceeds the prescribed value. Furthermore, super economy mode does not allow use of high-speed clocks (IOSC, OSC3, and EXOSC) because of its lack of drive capability.

Switching to super economy mode from another mode (automatic mode)

1. Check to see if V_{DD} meets the requirement using the supply voltage detector.
2. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
3. Check to see if the OSC1 oscillation has stabilized (see “Oscillation start procedure for the OSC1 oscillator circuit” in Section 2.3.4.).
4. Stop the high-speed clock sources.
5. Set the PWGTIM.DCCCLK[1:0] bits (first time only). (Set charge pump operating clock division ratio)
6. Set the PWGCTL.PWGMOD[2:0] bits to 0x5. (Set to super economy mode)
7. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)

Notes:
- Be sure to avoid setting to super economy mode under the conditions shown below, as it may cause a runaway CPU.
  1. V_{DD} does not meet the requirement for super economy mode.
  2. A clock source other than OSC1 is operating.
  3. OSC1 clock is not stabilized.
- The charge pump operates with the OSC1 clock. Therefore, to put the CPU into SLEEP state in super economy mode, the clock sources must be configured so that OSC1 only will operate in SLEEP mode (CLGOSC.OSC1SLPC bit = 0 and other CLGOSC.****SLPC bits = 1).

Switching to automatic mode/economy mode from super economy mode

1. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
2. Set the PWGCTL.PWGMOD[2:0] bits to 0x0. (Set to automatic mode)
   Or set the PWGCTL.PWGMOD[2:0] bits to 0x3. (Set to economy mode)
3. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
4. Check to see if the PWGINTF.MODCMPIF bit = 1 (mode transition completed).

For the PWGINTF.MODCMPIF bit set conditions, refer to “Interrupts.”

Notes:
- Be sure to avoid switching to normal mode directly from super economy mode, as it may cause a malfunction. When using a high-speed clock, first switch to automatic mode before starting the clock source.
- The PWGINTF.MODCMPIF bit is set to 1 after a lapse of 10 ms from the switching operation from super economy mode to automatic mode (or economy mode). Do not perform heavy-load operations, such as starting a high-speed clock source, before the PWGINTF.MODCMPIF bit is set to 1, as it may cause a malfunction.

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
    - 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.

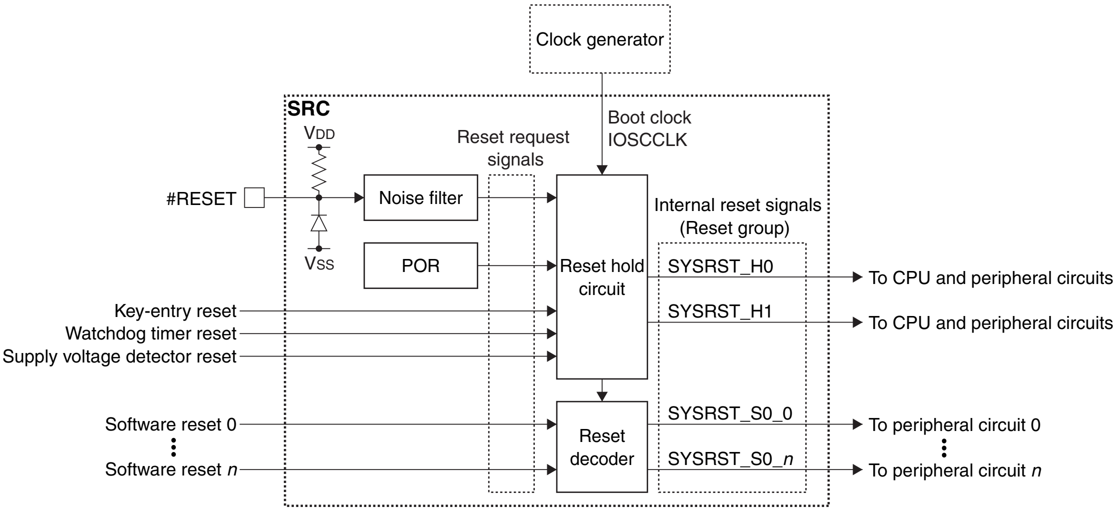


Figure 2.2.1.1 SRC Configuration

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

POR (Power On Reset) issues a reset request when the rise of VDD is detected. Reset requests from this circuit ensure that the system will be reset properly when the power is turned on. Figure 2.2.3.1 shows an example of POR internal reset operation according to variations in VDD.

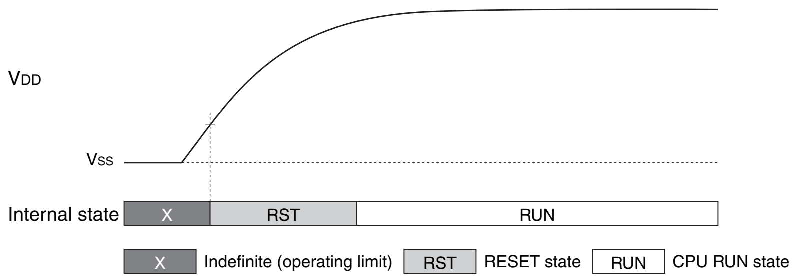


Figure 2.2.3.1 Example of Internal Reset by POR

For the POR electrical specifications, refer to “POR 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 Key-entry reset Supply voltage detector 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	
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.
  - IOSC oscillator circuit that oscillates with a fast startup and no external parts required
  - High-precision and low-power OSC1 oscillator circuit that uses a 32.768 kHz crystal resonator
  - OSC3 oscillator circuit in which the oscillator type can be specified from crystal/ceramic oscillator (an external resonator is required), CR oscillator (an external R is required), and internal oscillator
  - 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.
- IOSCCLK output from the IOSC 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.
  - 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.

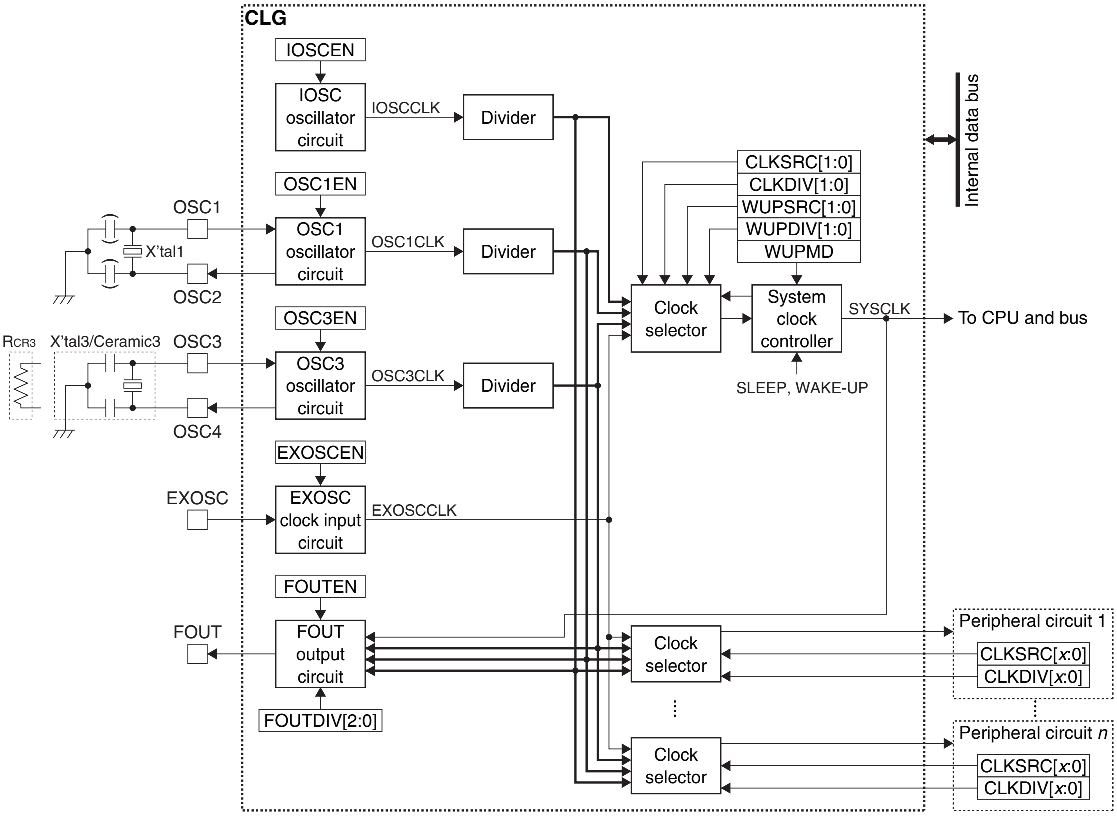


Figure 2.3.1.1 CLG Configuration

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	–	OSC1 oscillator circuit input
OSC2	A	–	OSC1 oscillator circuit output
OSC3	A	–	OSC3 oscillator circuit input
OSC4	A	–	OSC3 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

IOSC oscillator circuit

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

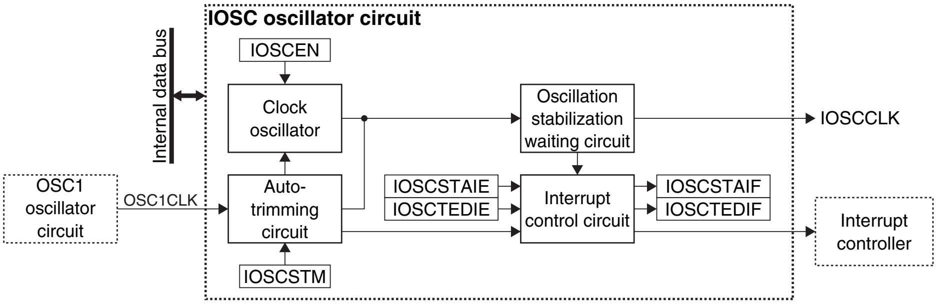


Figure 2.3.3.1 IOSC Oscillator Circuit Configuration

The IOSC oscillator circuit output clock IOSCCLK is used as SYSCLK at booting. The IOSC oscillator circuit is equipped with an auto-trimming function that automatically adjusts the frequency. This helps reduce frequency deviation due to unevenness in manufacturing quality, temperature, and changes in voltage. For more information on the auto-trimming function and the oscillation characteristics, refer to “IOSC oscillation auto-trimming function” in this chapter and “IOSC oscillator circuit characteristics” in the “Electrical Characteristics” chapter, respectively.

OSC1 oscillator circuit

The OSC1 oscillator circuit is a high-precision and low-power oscillator circuit that uses a 32.768 kHz crystal resonator. Figure 2.3.3.2 shows the configuration of the OSC1 oscillator circuit.

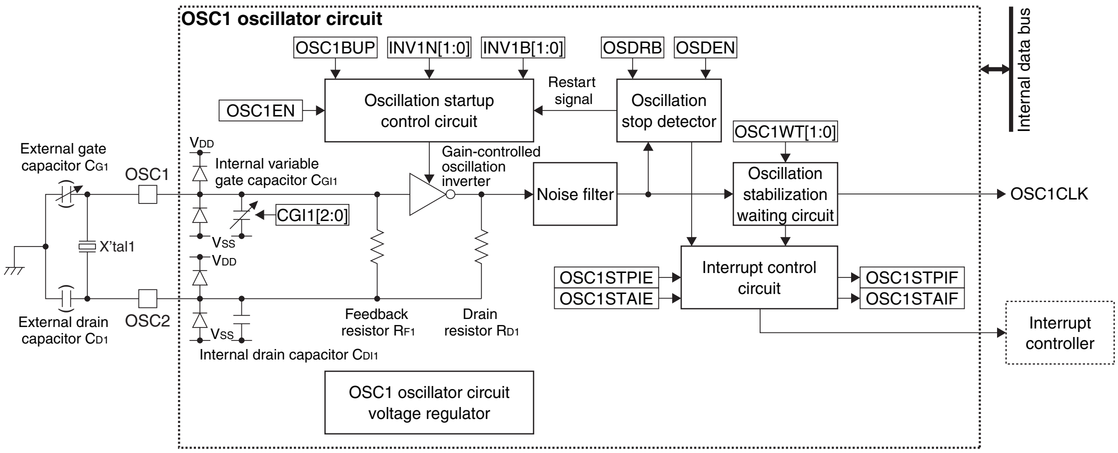


Figure 2.3.3.2 OSC1 Oscillator Circuit Configuration

This oscillator circuit includes a gain-controlled oscillation inverter and a variable gate capacitor allowing use of various crystal resonators with ranges from cylinder type through surface-mount type.
The oscillator circuit also includes a feedback resistor and a drain resistor, so no external parts are required except for a crystal resonator. The embedded oscillation stop detector, which detects oscillation stop and restarts the oscillator, allows the system to operate in safety under adverse environments that may stop the oscillation. The oscillation startup control circuit operates for a set period of time after the oscillation is enabled to assist the oscillator in initiating, this makes it possible to use a low-power resonator that is difficult to start up. For the recommended parts and the oscillation characteristics, refer to the "Basic External Connection Diagram" chapter and "OSC1 oscillator circuit characteristics" in the "Electrical Characteristics" chapter, respectively.

Note: Depending on the circuit board or the crystal resonator type used, an external gate capacitor C_{G1} and a drain capacitor C_{D1} may be required.

OSC3 oscillator circuit

The OSC3 oscillator circuit is a high-speed oscillator circuit that allows software to select the oscillator type from three types shown below. Figure 2.3.3.3 shows the configuration of the OSC3 oscillator circuit.

Crystal/ceramic oscillator

This oscillator circuit includes a feedback resistor and a drain resistor, so no external part is required except for a crystal/ceramic resonator. The embedded gain-controlled inverter allows selection of the resonator from a wide frequency range.

CR oscillator

This oscillator circuit includes an oscillation capacitor (C_{CR3}), and the frequency can be adjusted by the resistor (R_{CR3}). No external part is required except for R_{CR3}.

Internal oscillator

This oscillator circuit operates without any external parts, and its oscillation frequency can be selected via software.

Notes:
- The maximum value of the OSC3 oscillator circuit oscillation frequency f_{OSC3} depends on the supply voltage V_{DD} value. For the oscillation frequency range, refer to "Recommended Operating Conditions" in the "Electrical Characteristics" chapter.
- When the CR oscillator is selected, the changes in the signals output from the I/O pins adjacent to the OSC3 and OSC4 pins may affect the oscillation frequency.
- When the internal oscillator is selected, be sure to avoid using the pins to which OSC3 and OSC4 are assigned as input pins, as it may affect the oscillation frequency.

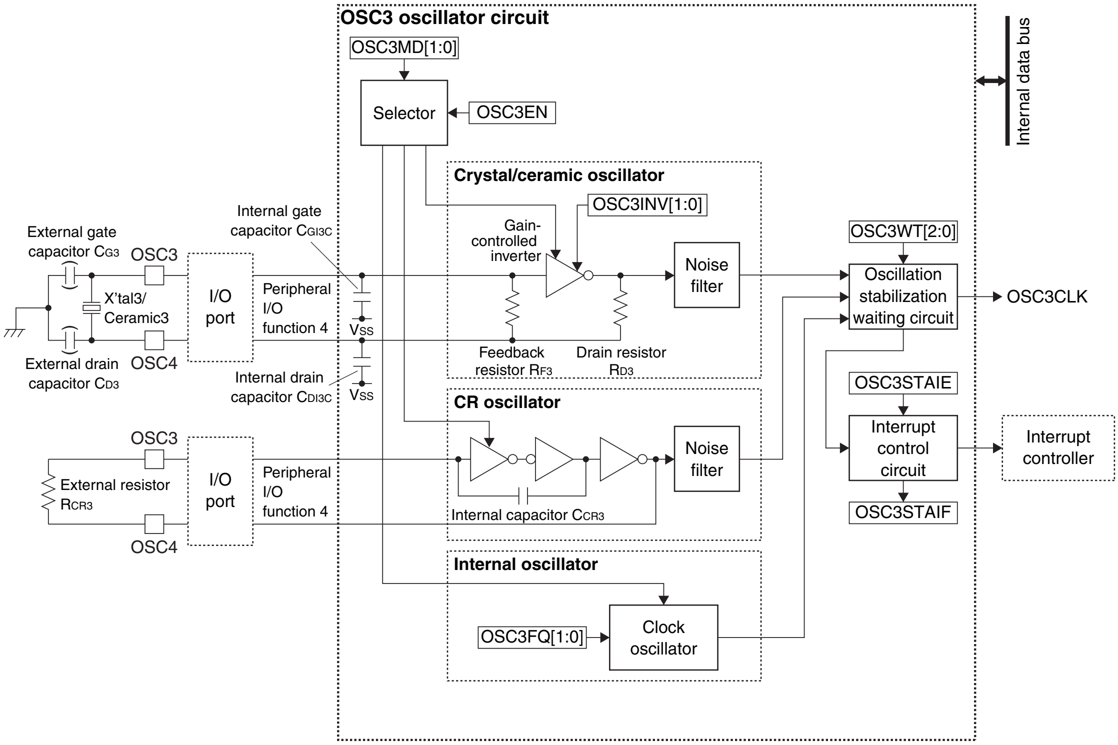


Figure 2.3.3.3 OSC3 Oscillator Circuit Configuration

For the recommended parts and the oscillation characteristics, refer to the "Basic External Connection Diagram" chapter and the "Electrical Characteristics" chapter, respectively.

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.

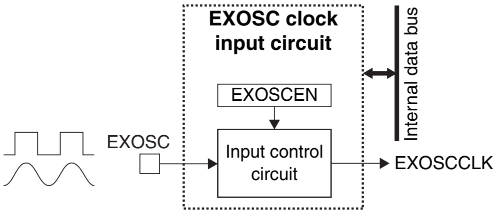


Figure 2.3.3.4 EXOSC Clock Input Circuit

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.

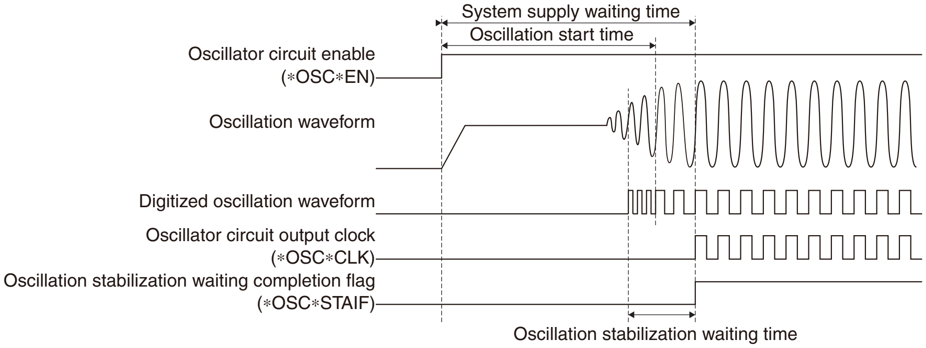


Figure 2.3.4.1 Oscillation Start Time and Oscillation Stabilization Waiting Time

The oscillation stabilization waiting times for the OSC1 and OSC3 oscillator circuits can be set using the CLGOSC1.OSC1WT[1:0] bits and CLGOSC3.OSC3WT[2:0] bits, respectively. 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 IOSC oscillator circuit is fixed at 16 IOSCCLK clocks. The oscillation stabilization waiting time for the OSC1 oscillator circuit should be set to 16,384 OSC1CLK clocks or more. The oscillation stabilization waiting time for the OSC3 oscillator circuit should be set to 1,024 OSC3CLK clocks or more when crystal/ceramic oscillator is selected, or four OSC3CLK clocks or more when CR oscillator or internal oscillator is selected.
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.

By default, the oscillation startup control circuit in the OSC1 oscillator circuit is enabled with the CLGOSC1.OSC1BUP bit set to 1. This reduces oscillation start time by using the high-gain oscillation inverter for a set period of time (startup boosting operation) after the oscillator circuit is enabled (by setting the CLGOSC.OSC1EN bit to 1). Note, however, that the oscillation operation may become unstable if there is a large gain differential between normal operation and startup boosting operation. Furthermore, the oscillation start time being actually reduced depends on the characteristics of the resonator used. Figure 2.3.4.2 shows an operation example when the oscillation startup control circuit is used.

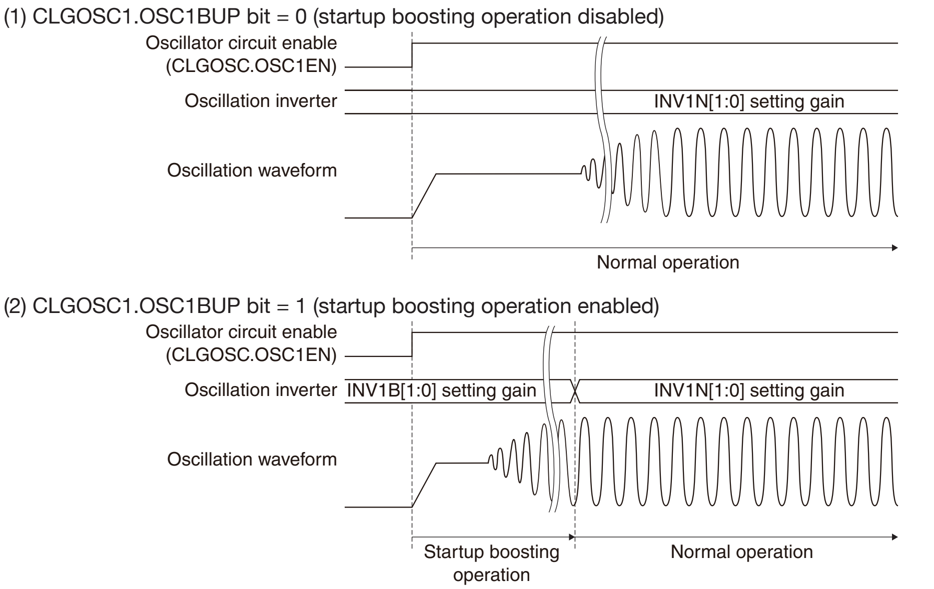


Figure 2.3.4.2 Operation Example when the Oscillation Startup Control Circuit is Used

Oscillation start procedure for the IOSC oscillator circuit

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

1. Write 1 to the CLGINTF.IOSCSTAIF bit. (Clear interrupt flag)
2. Write 1 to the CLGINTE.IOSCSTAIE bit. (Enable interrupt)
3. Write 1 to the CLGOSC.IOSCEN bit. (Start oscillation)
4. IOSCCLK can be used if the CLGINTF.IOSCSTAIF bit = 1 after an interrupt occurs.

Oscillation start procedure for the OSC1 oscillator circuit

Follow the procedure shown below to start oscillation of the OSC1 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.INV1B[1:0] bits (Set oscillation inverter gain for startup boosting period)
   - CLGOSC1.INV1N[1:0] bits (Set oscillation inverter gain)
   - CLGOSC1.CGI1[2:0] bits (Set internal gate capacitor)
   - CLGOSC1.OSC1WT[1:0] bits (Set oscillation stabilization waiting time)

   When the oscillation startup control circuit is not used, configure the following bit (see Figure 2.3.4.2):
   - Set the CLGOSC1.OSC1BUP bit to 0. (Disable oscillation startup control circuit)
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 values of the CLGOSC1.INV1N[1:0], CLGOSC1.CGI1[2:0], CLGOSC1.OSC1WT[1:0], and CLGOSC1.INV1B[1:0] bits should be determined after performing evaluation using the populated circuit board.

Oscillation start procedure for the OSC3 oscillator circuit

Follow the procedure shown below to start oscillation of the OSC3 oscillator circuit.

1. Write 1 to the CLGINTF.OSC3STAIF bit. (Clear interrupt flag)
2. Write 1 to the CLGINTE.OSC3STAIE bit. (Enable interrupt)
3. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGOSC3 register bits.
   - CLGOSC3.OSC3MD[1:0] bits (Select oscillator type)
   - CLGOSC3.OSC3WT[2:0] bits (Set oscillation stabilization waiting time)

   In addition to the above, configure the following bits when using the crystal/ceramic oscillator:
   - CLGOSC3.OSC3INV[1:0] bits (Set oscillation inverter gain)

   Configure the following bits when using the internal oscillator:
   - CLGOSC3.OSC3FQ[1:0] bits (Select oscillation frequency)
5. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
6. When using the crystal/ceramic or CR oscillator, assign the OSC3 oscillator input/output functions to the ports. (Refer to the “I/O Ports” chapter.)
7. Write 1 to the CLGOSC.OSC3EN bit. (Start oscillation)
8. OSC3CLK can be used if the CLGINTF.OSC3STAIF bit = 1 after an interrupt occurs.

The setting values of the CLGOSC3.OSC3INV[1:0] and CLGOSC3.OSC3WT[2:0] bits should be determined after performing evaluation using the populated circuit board.

Note: Make sure the CLGOSC.OSC3EN bit is set to 0 (while the OSC3 oscillation is halted) when switching the oscillator within three types.

System clock switching

The CPU boots using IOSCCLK 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.IOSCSLPC, CLGOSC.OSC1SLPC, CLGOSC.OSC3SLPC, and CLGOSC.EXOSCSLPC bits are used for this control. Figure 2.3.4.3 shows a control example.

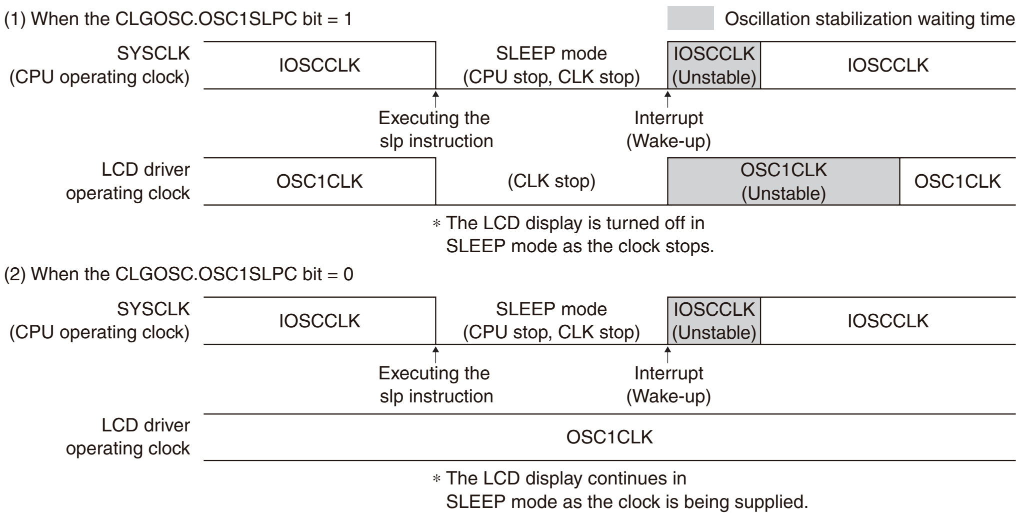


Figure 2.3.4.3 Clock Control Example in SLEEP Mode

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.

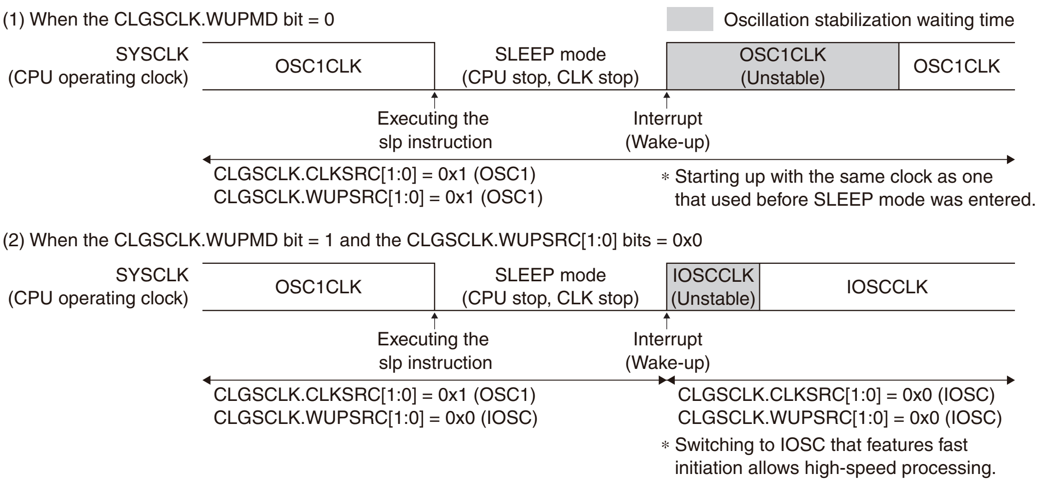


Figure 2.3.4.4 Clock Control Example at SLEEP Cancelation

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)

IOSC oscillation auto-trimming function

The auto-trimming function adjusts the IOSCCLK clock frequency by trimming the clock with reference to the high precision OSC1CLK clock generated by the OSC1 oscillator circuit. Follow the procedure shown below to enable the auto-trimming function.

1. After enabling the OSC1 oscillation, check if the stabilized clock is supplied (CLGINTF.OSC1STAIF bit = 1).
2. After enabling the IOSC oscillation, check if the stabilized clock is supplied (CLGINTF.IOSCSTAIF bit = 1).
3. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
4. If the SYSCLK clock source is IOSC, set the CLGSCLK.CLKSRC[1:0] bits to a value other than 0x0 (IOSC).
5. Write 1 to the CLGINTF.IOSCTEDIF bit. (Clear interrupt flag)
6. Write 1 to the CLGINTE.IOSCTEDIE bit. (Enable interrupt)
7. Write 1 to the CLGIOSC.IOSCSTM bit. (Enable IOSC oscillation auto-trimming)
8. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
9. The trimmed IOSCCLK can be used if the CLGINTF.IOSCTEDIF bit = 1 after an interrupt occurs.

After the trimming operation has completed, the CLGIOSC.IOSCSTM bit automatically reverts to 0. Although the triming time depends on the temperature, an average of several 10 ms is required. When IOSCCLK is being used as the system clock or a peripheral circuit clock, do not use the auto-trimming function.

OSC1 oscillation stop detection function

The oscillation stop detection function restarts the OSC1 oscillator circuit when it detects oscillation stop under adverse environments that may stop the oscillation. Follow the procedure shown below to enable the oscillation stop detection function.

1. After enabling the OSC1 oscillation, check if the stabilized clock is supplied (CLGINTF.OSC1STAIF bit = 1).
2. Write 1 to the CLGINTF.OSC1STPIF bit. (Clear interrupt flag)
3. Write 1 to the CLGINTE.OSC1STPIE bit. (Enable interrupt)
4. Write 0x0096 to the MSCPROT.PROT[15:0] bits. (Remove system protection)
5. Set the following CLGOSC1 register bits:
   - Set the CLGOSC1.OSDRB bit to 1. (Enable OSC1 restart function)
   - Set the CLGOSC1.OSDEN bit to 1. (Enable oscillation stop detection function)
6. Write a value other than 0x0096 to the MSCPROT.PROT[15:0] bits. (Set system protection)
7. The OSC1 oscillation stops if the CLGINTF.OSC1STPIF bit = 1 after an interrupt occurs.
   If the CLGOSC1.OSDRB bit = 1, the hardware restarts the OSC1 oscillator circuit.

Note: Enabling the oscillation stop detection function increase the oscillation stop detector current (I_{OSD1}).

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.

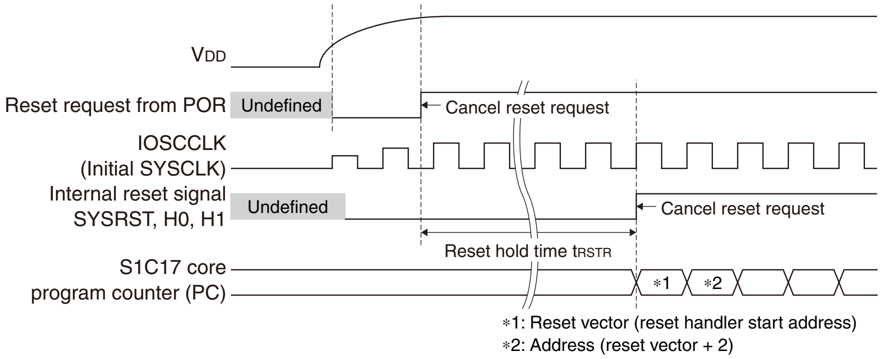


Figure 2.4.1.1 Initial Boot Sequence

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 "IOSC RUN," "OSC1 RUN," "OSC3 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 "IOSC HALT," "OSC1 HALT," "OSC3 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.IOSCSLPC/OSC1SLPC/OSC3SLPC/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.
The RAM retains data even in SLEEP mode.

Note: The current consumption when a clock source is active in SLEEP mode by setting the CLGOSC. IOSCSLPC/OSC1SLPC/OSC3SLPC/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_{HALT2}, and I_{HALT3}" 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.

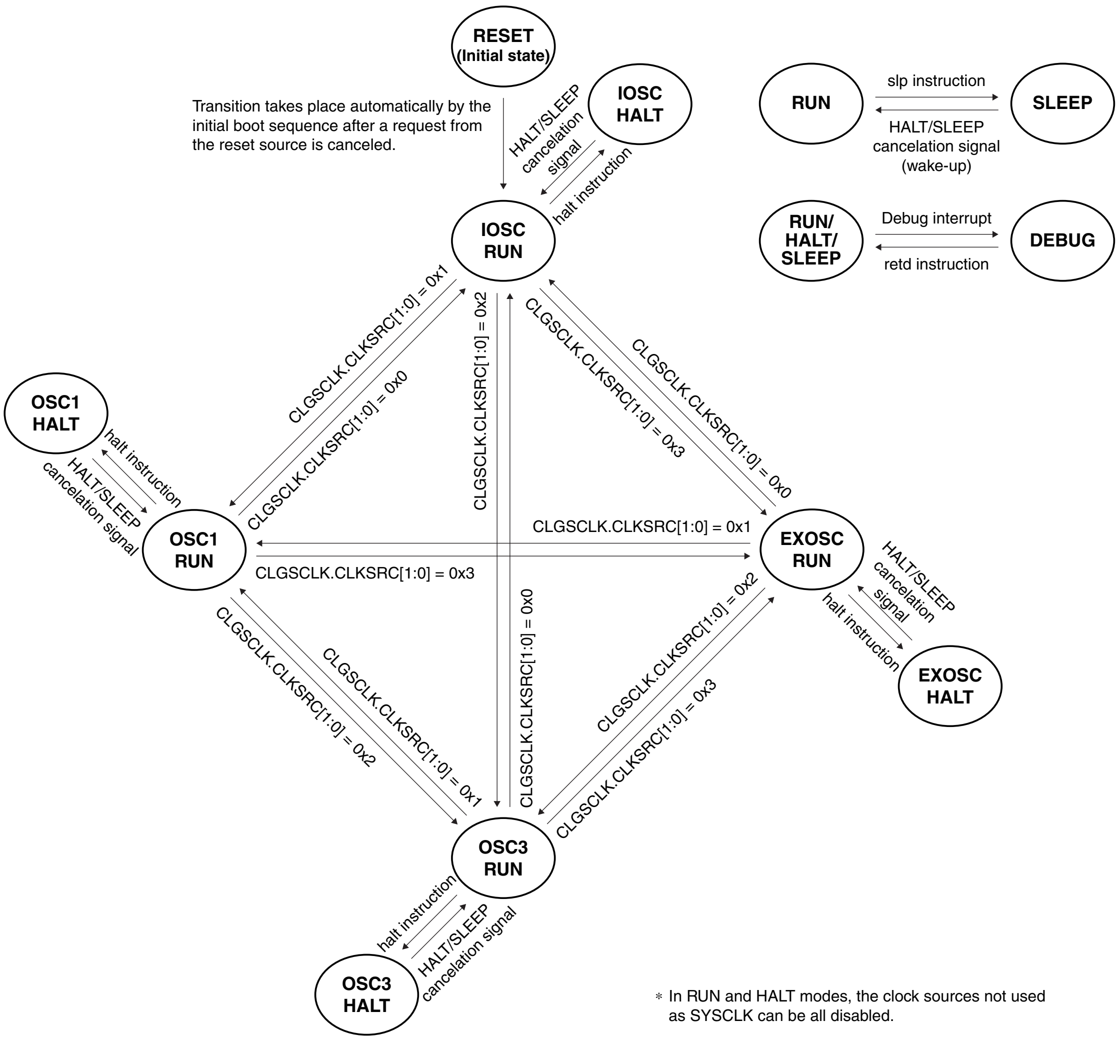


Figure 2.4.2.1 Operating Mode-to-Mode State Transition Diagram

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

PWG2 and CLG have a function to generate the interrupts shown in Table 2.5.1.

Table 2.5.1 PWG2 and CLG Interrupt Functions

Interrupt	Interrupt flag	Set condition	Clear condition
PWG2 mode transition completion	PWGINTF.MODCMPIF	When the transition from super economy mode to another mode has completed, or when the transition from normal mode to economy mode has completed in automatic mode (See Notes below.)	Writing 1
IOSC oscillation stabilization waiting completion	CLGINTF.IOSCSTAIF	When the IOSC 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
OSC3 oscillation stabilization waiting completion	CLGINTF.OSC3STAIF	When the OSC3 oscillation stabilization waiting operation has completed after the oscillation starts	Writing 1
OSC1 oscillation stop	CLGINTF.OSC1STPIF	When OSC1CLK is stopped, or when the CLGOSC.OSC1EN or CLGOSC1.OSDEN bit setting is altered from 1 to 0.	Writing 1
IOSC oscillation auto-trimming completion	CLGINTF.IOSCTEDIF	When the IOSC oscillation auto-trimming operation has completed	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.

Notes:
- The PWGINTF.MODCMPIF bit is set to 1 if a condition shown above is met only when the OSC1 oscillator circuit is operating regardless of RUN or SLEEP mode.
- When a transition, from RUN mode in which the system runs with a high-speed clock to SLEEP mode in which the OSC1 oscillator circuit only operates (high-speed clocks are halted), has occurred in automatic mode, the PWGINTF.MODCMPIF bit is set to 1 after a lapse of 1 ms from entering SLEEP mode. If the PWGINTE.MODCMPIE bit = 1 at this point, an interrupt occurs and the CPU wakes up from SLEEP mode. When putting the CPU to SLEEP mode with the OSC1 oscillator circuit activated, set the PWGINTE.MODCMPIE bit to 0.

2.6 Control Registers

PWG2 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PWGCTL	15–8	–	0x00	–	R	–
	7–3	–	0x00	–	R	
	2–0	PWGMOD[2:0]	0x0	H0	R/WP	

Bits 15–3 Reserved

Bits 2–0 PWGMOD[2:0]

These bits control the PWG2 operating mode.

Table 2.6.1 PWG2 Operating Mode

PWGCTL.PWGMOD[2:0] bits	Operating mode
0x7–0x6	Reserved
0x5	Super economy mode
0x4	Reserved
0x3	Economy mode
0x2	Normal mode
0x1	Reserved
0x0	Automatic mode

Note: The PWGCTL.PWGMOD[2:0] bits are set to 0x0 when 0x7, 0x6, 0x4, or 0x1 is written.

PWG2 Timing Control Register

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

Bits 15–2 Reserved

Bits 1–0 DCCCLK[1:0]

These bits set the charge pump operating clock (select an OSC1 clock division ratio).

Table 2.6.2 Charge Pump Operating Clock Setting

PWGTIM.DCCCLK[1:0] bits	OSC1 division ratio
0x3	1/256
0x2	1/128
0x1	1/64
0x0	1/32

PWG2 Interrupt Flag Register

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

Bits 15–1 Reserved

Bit 0 MODCMPIF

This bit indicates the PWG2 mode transition completion interrupt cause occurrence status.

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

PWG2 Interrupt Enable Register

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

Bits 15–1 Reserved

Bit 0 MODCMPIE

These bits enable the PWG2 mode transition completion interrupt.

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

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.OSC1EN, CLGOSC.OSC3EN, CLGOSC.IOSCEN) 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 2.6.3 SYSCLK Clock Source and Division Ratio Settings at Wake-up

CLGSCLK. WUPDIV[1:0] bits	CLGSCLK.WUPSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSCCLK	OSC1CLK	OSC3CLK	EXOSCCLK
0x3	1/8	Reserved	1/16	Reserved
0x2	1/4	Reserved	1/8	Reserved
0x1	1/2	1/2	1/2	Reserved
0x0	1/1	1/1	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 2.6.4 SYSCLK Clock Source and Division Ratio Settings

CLGSCLK. CLKDIV[1:0] bits	CLGSCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSCCLK	OSC1CLK	OSC3CLK	EXOSCCLK
0x3	1/8	Reserved	1/16	Reserved
0x2	1/4	Reserved	1/8	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	OSC3SLPC	1	H0	R/W	
	9	OSC1SLPC	1	H0	R/W	
	8	IOSCSLPC	1	H0	R/W	
	7–4	–	0x0	–	R	
	3	EXOSCEN	0	H0	R/W	
	2	OSC3EN	0	H0	R/W	
	1	OSC1EN	0	H0	R/W	
	0	IOSCEN	1	H0	R/W	

Bits 15–12 Reserved

Bit 11 EXOSCSLPC
Bit 10 OSC3SLPC
Bit 9 OSC1SLPC
Bit 8 IOSCSLPC

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.OSC3SLPC bit: OSC3 oscillator circuit
CLGOSC.OSC1SLPC bit: OSC1 oscillator circuit
CLGOSC.IOSCSLPC bit: IOSC oscillator circuit

Bits 7–4 Reserved

Bit 3 EXOSCEN
Bit 2 OSC3EN
Bit 1 OSC1EN
Bit 0 IOSCEN

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.OSC3EN bit: OSC3 oscillator circuit
CLGOSC.OSC1EN bit: OSC1 oscillator circuit
CLGOSC.IOSCEN bit: IOSC oscillator circuit

CLG IOSC Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGIOSC	15–8	–	0x00	–	R	–
	7–5	–	0x0	–	R	
	4	IOSCSTM	0	H0	R/WP	
	3–0	–	0x0	–	R	

Bits 15–5 Reserved

Bit 4 IOSCSTM

This bit controls the IOSCCLK auto-trimming function.

1 (WP): Start trimming
0 (WP): Stop trimming
1 (R): Trimming is executing.
0 (R): Trimming has finished. (Trimming operation inactivated.)

This bit is automatically cleared to 0 when trimming has finished.

Notes:
- Do not use IOSCCLK as the system clock or peripheral circuit clocks while the CLGIOSC.IOSCSTM bit = 1.
- The auto-trimming function does not work if the OSC1 oscillator circuit is stopped. Make sure the CLGINTF.OSC1STAIF bit is set to 1 before starting the trimming operation.

Bits 3–0 Reserved

CLG OSC1 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGOSC1	15	–	0	–	R	–
	14	OSDRB	1	H0	R/WP	
	13	OSDEN	0	H0	R/WP	
	12	OSC1BUP	1	H0	R/WP	
	11	–	0	–	R	
	10–8	CGI1[2:0]	0x0	H0	R/WP	
	7–6	INV1B[1:0]	0x2	H0	R/WP	
	5–4	INV1N[1:0]	0x1	H0	R/WP	
	3–2	–	0x0	–	R	
	1–0	OSC1WT[1:0]	0x2	H0	R/WP	

Bit 15 Reserved

Bit 14 OSDRB

This bit enables the OSC1 oscillator circuit restart function by the oscillation stop detector when OSC1 oscillation stop is detected.

1 (R/WP): Enable (Restart the OSC1 oscillator circuit when oscillation stop is detected.)
0 (R/WP): Disable

Bit 13 OSDEN

This bit controls the oscillation stop detector in the OSC1 oscillator circuit.

1 (R/WP): OSC1 oscillation stop detector on
0 (R/WP): OSC1 oscillation stop detector off

Note: Do not write 1 to the CLGOSC1.OSDEN bit before stabilized OSC1CLK is supplied. Furthermore, the CLGOSC1.OSDEN bit should be set to 0 when the CLGOSC.OSC1EN bit is set to 0.

Bit 12 OSC1BUP

This bit enables the oscillation startup control circuit in the OSC1 oscillator circuit.

1 (R/WP): Enable (Activate booster operation at startup.)
0 (R/WP): Disable

Bit 11 Reserved

Bits 10–8 CGI1[2:0]

These bits set the internal gate capacitance in the OSC1 oscillator circuit.

Table 2.6.5 OSC1 Internal Gate Capacitance Setting

CLGOSC1.CGI1[2:0] bits	Capacitance
0x7	Max.
0x6	↑
0x5	
0x4	
0x3	
0x2	
0x1	↓
0x0	Min.

For more information, refer to "OSC1 oscillator circuit characteristics, Internal gate capacitance C_{GI1}" in the "Electrical Characteristics" chapter.

Bits 7–6 INV1B[1:0]

These bits set the oscillation inverter gain that will be applied at boost startup of the OSC1 oscillator circuit.

Table 2.6.6 Setting Oscillation Inverter Gain at OSC1 Boost Startup

CLGOSC1.INV1B[1:0] bits	Inverter gain
0x3	Max.
0x2	↑
0x1	↓
0x0	Min.

Note: The CLGOSC1.INV1B[1:0] bits must be set to a value equal to or larger than the CLGOSC1.INV1N[1:0] bits.

Bits 5–4 INV1N[1:0]

These bits set the oscillation inverter gain applied at normal operation of the OSC1 oscillator circuit.

Table 2.6.7 Setting Oscillation Inverter Gain at OSC1 Normal Operation

CLGOSC1.INV1N[1:0] bits	Inverter gain
0x3	Max.
0x2	↑
0x1	↓
0x0	Min.

Bits 3–2 Reserved

Bits 1–0 OSC1WT[1:0]

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

Table 2.6.8 OSC1 Oscillation Stabilization Waiting Time Setting

CLGOSC1.OSC1WT[1:0] bits	Oscillation stabilization waiting time
0x3	65,536 clocks
0x2	16,384 clocks
0x1	4,096 clocks
0x0	Reserved

CLG OSC3 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGOSC3	15–12	–	0x0	–	R	–
	11–10	OSC3FQ[1:0]	0x3	H0	R/WP	
	9–8	OSC3MD[1:0]	0x0	H0	R/WP	
	7–6	–	0x0	–	R	
	5–4	OSC3INV[1:0]	0x1	H0	R/WP	
	3	–	0	–	R	
	2–0	OSC3WT[2:0]	0x0	H0	R/WP	

Bits 15–12 Reserved

Bits 11–10 OSC3FQ[1:0]

These bits set the oscillation frequency when the internal oscillator is selected as the OSC3 oscillator.

Table 2.6.9 OSC3 Internal Oscillator Frequency Setting

CLGOSC3.OSC3FQ[1:0] bits	OSC3 internal oscillator frequency
0x3	4 MHz
0x2	2 MHz
0x1	1 MHz
0x0	500 kHz

Bits 9–8 OSC3MD[1:0]

These bits select an oscillator type of the OSC3 oscillator circuit.

Table 2.6.10 OSC3 Oscillator Type Selection

CLGOSC3.OSC3MD[1:0] bits	OSC3 oscillator type
0x3	Reserved
0x2	Crystal/ceramic oscillator
0x1	CR oscillator
0x0	Internal oscillator

Bits 7–6 Reserved

Bits 5–4 OSC3INV[1:0]

These bits set the oscillation inverter gain when crystal/ceramic oscillator is selected as the OSC3 oscillator type.

Table 2.6.11 OSC3 Oscillation Inverter Gain Setting

CLGOSC3.OSC3INV[1:0] bits	Inverter gain
0x3	Max.
0x2	↑
0x1	↓
0x0	Min.

Bit 3 Reserved

Bits 2–0 OSC3WT[2:0]

These bits set the oscillation stabilization waiting time for the OSC3 oscillator circuit.

Table 2.6.12 OSC3 Oscillation Stabilization Waiting Time Setting

CLGOSC3.OSC3WT[2:0] bits	Oscillation stabilization waiting time
0x7	65,536 clocks
0x6	16,384 clocks
0x5	4,096 clocks
0x4	1,024 clocks
0x3	256 clocks
0x2	64 clocks
0x1	16 clocks
0x0	4 clocks

CLG Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGINTF	15–8	–	0x00	–	R	–
	7	–	0x0	–	R	
	6	(reserved)	0	H0	R	
	5	OSC1STPIF	0	H0	R/W	Cleared by writing 1.
	4	IOSCTEDIF	0	H0	R/W	
	3	–	0	–	R	–
	2	OSC3STAIF	0	H0	R/W	Cleared by writing 1.
	1	OSC1STAIF	0	H0	R/W	
	0	IOSCSTAIF	0	H0	R/W	

Bits 15–6 Reserved

Bit 5 OSC1STPIF
Bit 4 IOSCTEDIF

These bits indicate the OSC1 oscillation stop and IOSC oscillation auto-trimming completion interrupt cause occurrence statuses.

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 interrupt as follows:
CLGINTF.OSC1STPIF bit: OSC1 oscillation stop interrupt
CLGINTF.IOSCTEDIF bit: IOSC oscillation auto-trimming completion interrupt

Bit 3 Reserved

Bit 2 OSC3STAIF
Bit 1 OSC1STAIF
Bit 0 IOSCSTAIF

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.OSC3STAIF bit: OSC3 oscillator circuit
CLGINTF.OSC1STAIF bit: OSC1 oscillator circuit
CLGINTF.IOSCSTAIF bit: IOSC oscillator circuit

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

CLG Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGINTE	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6	(reserved)	0	H0	R	
	5	OSC1STPIE	0	H0	R/W	
	4	IOSCTEDIE	0	H0	R/W	
	3	–	0	–	R	
	2	OSC3STAIE	0	H0	R/W	
	1	OSC1STAIE	0	H0	R/W	
	0	IOSCSTAIE	0	H0	R/W	

Bits 15–6 Reserved

Bit 5 OSC1STPIE
Bit 4 IOSCTEDIE

These bits enable the OSC1 oscillation stop and IOSC oscillation auto-trimming completion interrupts.

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

Each bit corresponds to the interrupt as follows:
CLGINTE.OSC1STPIE bit: OSC1 oscillation stop interrupt
CLGINTE.IOSCTEDIE bit: IOSC oscillation auto-trimming completion interrupt

Bit 3 Reserved

Bit 2 OSC3STAIE
Bit 1 OSC1STAIE
Bit 0 IOSCSTAIE

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.OSC3STAIE bit: OSC3 oscillator circuit
CLGINTE.OSC1STAIE bit: OSC1 oscillator circuit
CLGINTE.IOSCSTAIE bit: IOSC 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 2.6.13 FOUT Clock Source and Division Ratio Settings

CLGFOUT. FOUTDIV[2:0] bits	CLGFOUT.FOUTSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSCCLK	OSC1CLK	OSC3CLK	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.

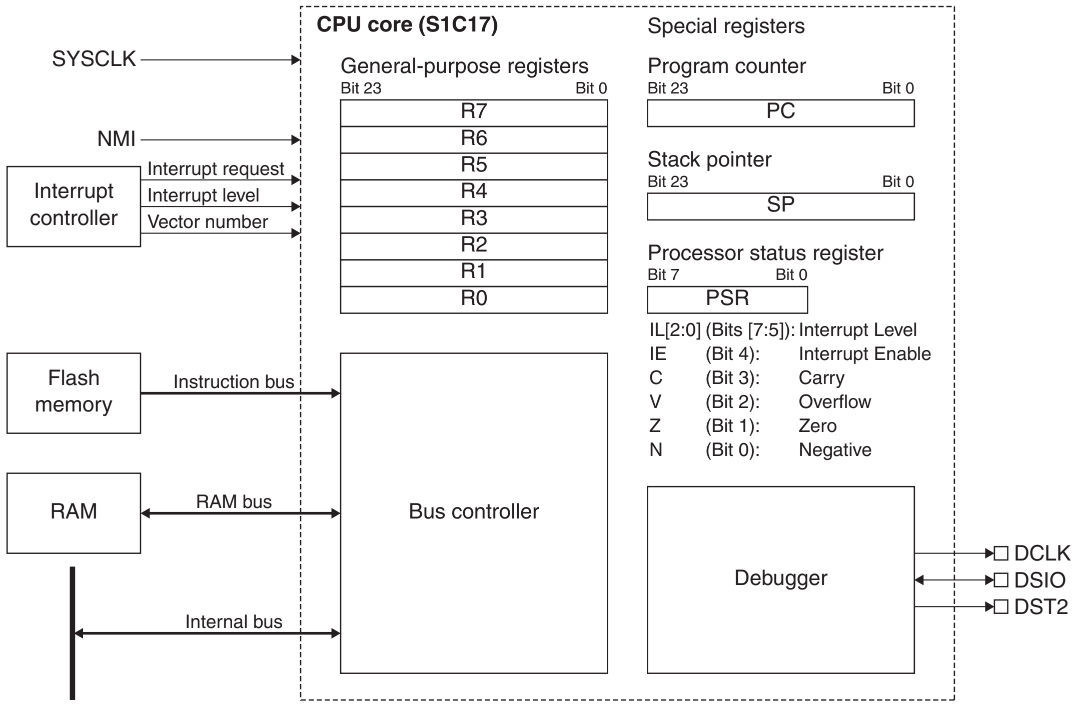


Figure 3.1.1 S1C17 Configuration

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.

Notes:
- Do not drive the DCLK pin with a high level from outside (e.g. pulling up with a resistor). Also, do not connect (short-circuit) between the DCLK pin and another GPIO port. In the both cases, the IC may not start up normally due to unstable pin input/output status at power on.
- Do not drive the DSIO pin with a low level from outside, as it generates a debug interrupt that puts the CPU into DEBUG mode.

3.3.4 External Connection

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

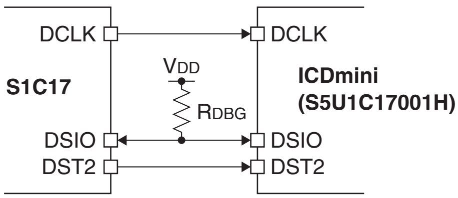


Figure 3.3.4.1 External Connection

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

3.3.5 Flash Security Function

This IC provides a security function to protect the internal Flash memory and RAM from unauthorized reading and tampering by using the debugger through ICDmini. Figure 3.3.5.1 shows a Flash security function setting flow.

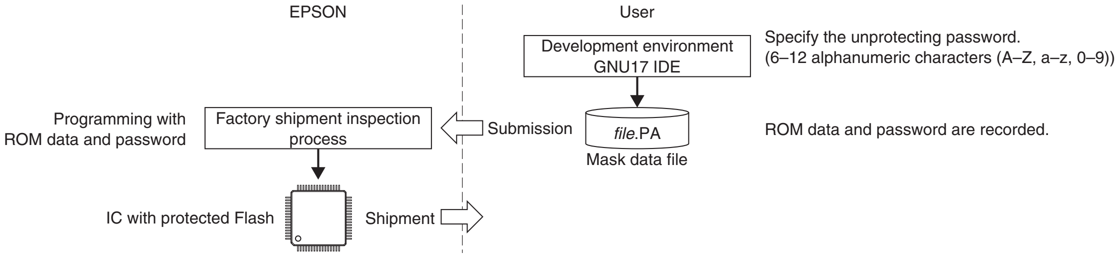


Figure 3.3.5.1 Shipment of IC with ROM Data Programmed and Flash Security Function Setting Flow

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

- The Flash memory and RAM data are undefined if they are 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.

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
- All memory and control registers are accessible in 16-bit width and one cycle.
- Write-protect function to protect system control registers

Figure 4.1.1 shows the memory map.

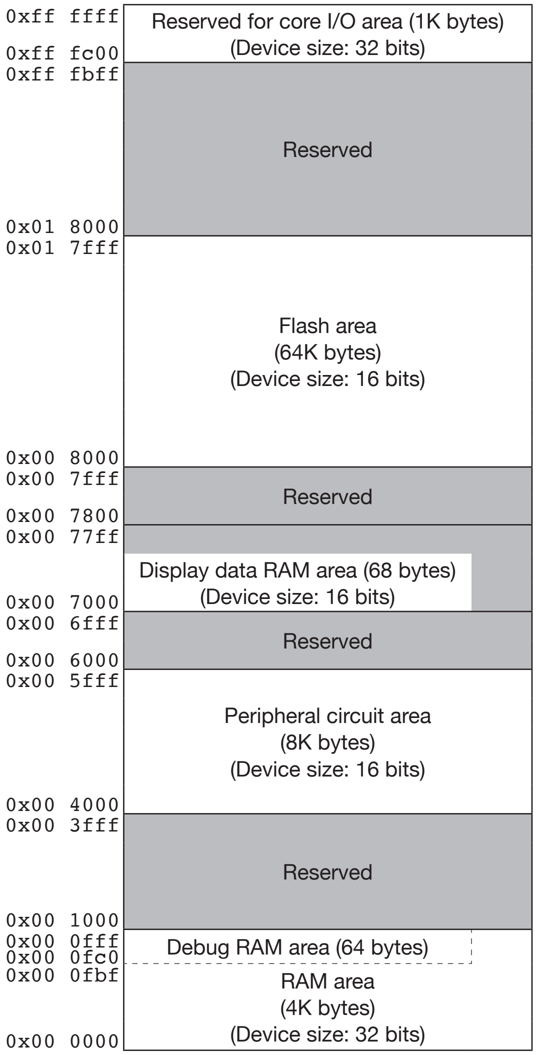


Figure 4.1.1 Memory Map

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

Table 4.3.1.1 shows the Flash memory pin.

Table 4.3.1.1 Flash Memory Pin

Pin name	I/O	Initial status	Function
V_{PP}	P	–	Flash programming power supply

For the V_{PP} voltage, refer to "Recommended Operating Conditions, Flash programming voltage V_{PP}" in the "Electrical Characteristics" chapter.

Note: Always leave the V_{PP} pin open except when programming the Flash memory.

4.3.2 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.

4.3.3 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 4.3.3.1 shows a connection diagram for on-board programming.

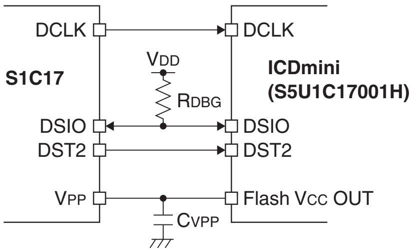


Figure 4.3.3.1 External Connection

The V_{PP} pin must be left open except when programming the Flash memory. However, it is not necessary to disconnect the wire when using ICDmini to supply the V_{PP} power, as ICDmini controls the power supply so that it will be supplied during Flash programming only. When supplying the V_{PP} power source, be sure to connect C_{VPP} for stabilizing the V_{PP} voltage.

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.

Note: The Flash programming requires a 1.8 V or higher V_{DD} voltage.

4.4 RAM

The RAM 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.

Note: The 64 bytes at the end of the RAM 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 RAM 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 RAM area results in the same operation (undefined value is read out) as when a reserved area is accessed.

4.5 Display Data RAM

The embedded display data RAM is used to store display data for the LCD driver. Areas unused for display data in the display data RAM can be used as a general-purpose RAM. For specific information on the display data RAM, refer to "Display Data RAM" in the "LCD Driver" chapter.

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 (PWG2)	0x4020	PWGCTL	PWG2 Control Register
	0x4022	PWGTIM	PWG2 Timing Control Register
	0x4024	PWGINTF	PWG2 Interrupt Flag Register
	0x4026	PWGINTE	PWG2 Interrupt Enable Register
Clock generator (CLG)	0x4040	CLGSCLK	CLG System Clock Control Register
	0x4042	CLGOSC	CLG Oscillation Control Register
	0x4044	CLGIOSC	CLG IOSC Control Register
	0x4046	CLGOSC1	CLG OSC1 Control Register
	0x4048	CLGOSC3	CLG OSC3 Control Register
	0x404c	CLGINTF	CLG Interrupt Flag Register
	0x404e	CLGINTE	CLG Interrupt Enable Register
	0x4050	CLGFOUT	CLG FOUT 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 (RTCA)	0x40c0	RTCCTL	RTC Control Register
	0x40c2	RTCALM1	RTC Second Alarm Register
	0x40c4	RTCALM2	RTC Hour/Minute Alarm Register
	0x40c6	RTCSWCTL	RTC Stopwatch Control Register
	0x40c8	RTCSEC	RTC Second/1Hz Register
	0x40ca	RTCHUR	RTC Hour/Minute Register
	0x40cc	RTCMON	RTC Month/Day Register
	0x40ce	RTCYAR	RTC Year/Week Register
	0x40d0	RTCINTF	RTC Interrupt Flag Register
	0x40d2	RTCINTE	RTC Interrupt Enable 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

Peripheral circuit	Address	Register name	
I/O ports (PPORT)	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
	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
	0x4226	P2INTF	P2 Port Interrupt Flag Register
	0x4228	P2INTCTL	P2 Port Interrupt 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
	0x4236	P3INTF	P3 Port Interrupt Flag Register
	0x4238	P3INTCTL	P3 Port Interrupt Control Register
	0x423c	P3MODSEL	P3 Port Mode 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
Universal port multiplexer (UPMUX)	0x4300	P0UPMUX0	P00–01 Universal Port Multiplexer Setting Register
	0x4302	P0UPMUX1	P02–03 Universal Port Multiplexer Setting Register
	0x4304	P0UPMUX2	P04–05 Universal Port Multiplexer Setting Register
	0x4306	P0UPMUX3	P06 Universal Port Multiplexer Setting Register
	0x4308	P1UPMUX0	P10–11 Universal Port Multiplexer Setting Register
	0x430a	P1UPMUX1	P12–13 Universal Port Multiplexer Setting Register
	0x430c	P1UPMUX2	P14–15 Universal Port Multiplexer Setting Register
	0x430e	P1UPMUX3	P16–17 Universal Port Multiplexer Setting Register
	0x4310	P2UPMUX0	P20–21 Universal Port Multiplexer Setting Register
	0x4312	P2UPMUX1	P22–23 Universal Port Multiplexer Setting Register
	0x4314	P2UPMUX2	P24–25 Universal Port Multiplexer Setting Register
	0x4316	P2UPMUX3	P26–27 Universal Port Multiplexer Setting Register
UART (UART) Ch.0	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
Synchronous serial interface (SPIA)	0x43b0	SPI0MOD	SPIA Ch.0 Mode Register
	0x43b2	SPI0CTL	SPIA Ch.0 Control Register
	0x43b4	SPI0TXD	SPIA Ch.0 Transmit Data Register
	0x43b6	SPI0RXD	SPIA Ch.0 Receive Data Register

Peripheral circuit	Address	Register name	
Synchronous serial interface (SPIA)	0x43b8	SPI0INTF	SPIA Ch.0 Interrupt Flag Register
	0x43ba	SPI0INTE	SPIA Ch.0 Interrupt Enable Register
I²C (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 (T16B) Ch.0	0x5000	T16B0CLK	T16B Ch.0 Clock Control Register
	0x5002	T16B0CTL	T16B Ch.0 Counter Control Register
	0x5004	T16B0MC	T16B Ch.0 Max Counter Data Register
	0x5006	T16B0TC	T16B Ch.0 Timer Counter Data Register
	0x5008	T16B0CS	T16B Ch.0 Counter Status Register
	0x500a	T16B0INTF	T16B Ch.0 Interrupt Flag Register
	0x500c	T16B0INTE	T16B Ch.0 Interrupt Enable Register
	0x5010	T16B0CCCTL0	T16B Ch.0 Compare/Capture 0 Control Register
	0x5012	T16B0CCR0	T16B Ch.0 Compare/Capture 0 Data Register
	0x5018	T16B0CCCTL1	T16B Ch.0 Compare/Capture 1 Control Register
	0x501a	T16B0CCR1	T16B Ch.0 Compare/Capture 1 Data Register
16-bit PWM timer (T16B) Ch.1	0x5040	T16B1CLK	T16B Ch.1 Clock Control Register
	0x5042	T16B1CTL	T16B Ch.1 Counter Control Register
	0x5044	T16B1MC	T16B Ch.1 Max Counter Data Register
	0x5046	T16B1TC	T16B Ch.1 Timer Counter Data Register
	0x5048	T16B1CS	T16B Ch.1 Counter Status Register
	0x504a	T16B1INTF	T16B Ch.1 Interrupt Flag Register
	0x504c	T16B1INTE	T16B Ch.1 Interrupt Enable Register
	0x5050	T16B1CCCTL0	T16B Ch.1 Compare/Capture 0 Control Register
	0x5052	T16B1CCR0	T16B Ch.1 Compare/Capture 0 Data Register
	0x5058	T16B1CCCTL1	T16B Ch.1 Compare/Capture 1 Control Register
	0x505a	T16B1CCR1	T16B Ch.1 Compare/Capture 1 Data Register
UART(UART) Ch.1	0x5200	UA1CLK	UART Ch.1 Clock Control Register
	0x5202	UA1MOD	UART Ch.1 Mode Register
	0x5204	UA1BR	UART Ch.1 Baud-Rate Register
	0x5206	UA1CTL	UART Ch.1 Control Register
	0x5208	UA1TXD	UART Ch.1 Transmit Data Register
	0x520a	UA1RXD	UART Ch.1 Receive Data Register
	0x520c	UA1INTF	UART Ch.1 Status and Interrupt Flag Register
	0x520e	UA1INTE	UART Ch.1 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
Sound generator (SNDA)	0x5300	SNDCLK	SNDA Clock Control Register
	0x5302	SNDSEL	SNDA Select Register
	0x5304	SNDCTL	SNDA Control Register
	0x5306	SNDDAT	SNDA Data Register
	0x5308	SNDINTF	SNDA Interrupt Flag Register
	0x530a	SNDINTE	SNDA Interrupt Enable Register
LCD driver (LCD8B)	0x5400	LCD8CLK	LCD8B Clock Control Register
	0x5402	LCD8CTL	LCD8B Control Register
	0x5404	LCD8TIM1	LCD8B Timing Control Register 1
	0x5406	LCD8TIM2	LCD8B Timing Control Register 2
	0x5408	LCD8PWR	LCD8B Power Control Register
	0x540a	LCD8DSP	LCD8B Display Control Register

Peripheral circuit	Address	Register name	
LCD driver (LCD8B)	0x540c	LCD8COMC0	LCD8B COM Pin Control Register 0
	0x5410	LCD8INTF	LCD8B Interrupt Flag Register
	0x5412	LCD8INTE	LCD8B 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
	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
R/F converter (RFC) Ch.2	0x5480	RFC2CLK	RFC Ch.2 Clock Control Register
	0x5482	RFC2CTL	RFC Ch.2 Control Register
	0x5484	RFC2TRG	RFC Ch.2 Oscillation Trigger Register
	0x5486	RFC2MCL	RFC Ch.2 Measurement Counter Low Register
	0x5488	RFC2MCH	RFC Ch.2 Measurement Counter High Register
	0x548a	RFC2TCL	RFC Ch.2 Time Base Counter Low Register
	0x548c	RFC2TCH	RFC Ch.2 Time Base Counter High Register
	0x548e	RFC2INTF	RFC Ch.2 Interrupt Flag Register
	0x5490	RFC2INTE	RFC Ch.2 Interrupt Enable Register
R/F converter (RFC) Ch.3	0x54a0	RFC3CLK	RFC Ch.3 Clock Control Register
	0x54a2	RFC3CTL	RFC Ch.3 Control Register
	0x54a4	RFC3TRG	RFC Ch.3 Oscillation Trigger Register
	0x54a6	RFC3MCL	RFC Ch.3 Measurement Counter Low Register
	0x54a8	RFC3MCH	RFC Ch.3 Measurement Counter High Register
	0x54aa	RFC3TCL	RFC Ch.3 Time Base Counter Low Register
	0x54ac	RFC3TCH	RFC Ch.3 Time Base Counter High Register
	0x54ae	RFC3INTF	RFC Ch.3 Interrupt Flag Register
	0x54b0	RFC3INTE	RFC Ch.3 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.
	7–3	–	0x06	–	R	–
	2–0	IRAMSZ[2:0]	0x3	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
0x7–0x4	Reserved
0x3	4KB
0x2	2KB
0x1	1KB
0x0	512B

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]	0x1	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

When V_{DD} = 1.2 to 1.6 V

FLASHCWAIT.RDWAIT[1:0] bits	Number of bus Access cycles	System clock frequency
0x3	4	1.1 MHz (max.)
0x2	3	1.1 MHz (max.)
0x1	2	1.1 MHz (max.)
0x0	1	800 kHz (max.)

When V_{DD} = 1.6 to 3.6 V

FLASHCWAIT.RDWAIT[1:0] bits	Number of bus Access cycles	System clock frequency
0x3	4	4.2 MHz (max.)
0x2	3	4.2 MHz (max.)
0x1	2	4.2 MHz (max.)
0x0	1	2.1 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.

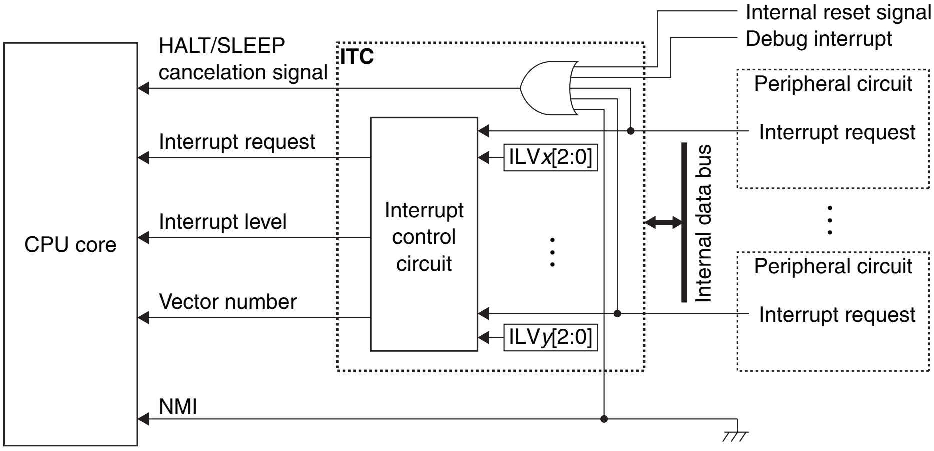


Figure 5.1.1 ITC Configuration

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 • Key reset • Watchdog timer overflow • Supply voltage detector reset	1
1 (0x01)	TTBR + 0x04	Address misaligned interrupt	Memory access instruction	2
–	(0xfffc00)	Debug 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	Power generator interrupt	PWG2 mode transition completion	
7 (0x07)	TTBR + 0x1c	Clock generator interrupt	• IOSC oscillation stabilization waiting completion • OSC1 oscillation stabilization waiting completion • OSC3 oscillation stabilization waiting completion • OSC1 oscillation stop • IOSC oscillation auto-trimming completion	
8 (0x08)	TTBR + 0x20	Real-time clock interrupt	• 1-day, 1-hour, 1-minute, and 1-second • 1/32-second, 1/8-second, 1/4-second, and 1/2-second • Stopwatch 1 Hz, 10 Hz, and 100 Hz • Alarm • Theoretical regulation completion	
9 (0x09)	TTBR + 0x24	16-bit timer Ch.0 interrupt	Underflow	
10 (0x0a)	TTBR + 0x28	UART Ch.0 interrupt	• End of transmission • Framing error • Parity error • Overrun error • Receive buffer two bytes full • Receive buffer one byte full • Transmit buffer empty	
11 (0x0b)	TTBR + 0x2c	16-bit timer Ch.1 interrupt	Underflow	
12 (0x0c)	TTBR + 0x30	Synchronous serial interface Ch.0 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty • Overrun error	
13 (0x0d)	TTBR + 0x34	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	
14 (0x0e)	TTBR + 0x38	16-bit PWM timer Ch.0 interrupt	• Capture overwrite • Compare/capture • Counter MAX • Counter zero	
15 (0x0f)	TTBR + 0x3c	16-bit PWM timer Ch.1 interrupt	• Capture overwrite • Compare/capture • Counter MAX • Counter zero	
16 (0x10)	TTBR + 0x40	UART Ch.1 interrupt	• End of transmission • Framing error • Parity error • Overrun error • Receive buffer two bytes full • Receive buffer one byte full • Transmit buffer empty	
17 (0x11)	TTBR + 0x44	16-bit timer Ch.2 interrupt	Underflow	
18 (0x12)	TTBR + 0x48	Sound generator interrupt	• Sound buffer empty • Sound output completion	
19 (0x13)	TTBR + 0x4c	LCD driver interrupt	Frame	
20 (0x14)	TTBR + 0x50	R/F converter Ch.0 interrupt	• Reference oscillation completion • Sensor A oscillation completion • Sensor B oscillation completion • Measurement counter overflow error • Time base counter overflow error	
21 (0x15)	TTBR + 0x54	R/F converter Ch.1 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
22 (0x16)	TTBR + 0x58	R/F converter Ch.2 interrupt	• Reference oscillation completion • Sensor A oscillation completion • Sensor B oscillation completion • Measurement counter overflow error • Time base counter overflow error	
23 (0x17)	TTBR + 0x5c	R/F converter Ch.3 interrupt	• Reference oscillation completion • Sensor A oscillation completion • Sensor B oscillation completion • Measurement counter overflow error • Time base counter overflow error	
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	Clock generator interrupt (ILVCLG)
	7–3	–	0x00	–	R	–
	2–0	ILV2[2:0]	0x0	H0	R/W	Power generator interrupt (ILVPWG2)
ITCLV2 (ITC Interrupt Level Setup Register 2)	15–11	–	0x00	–	R	–
	10–8	ILV5[2:0]	0x0	H0	R/W	16-bit timer Ch.0 interrupt (ILVT16_0)
	7–3	–	0x00	–	R	–
	2–0	ILV4[2:0]	0x0	H0	R/W	Real-time clock interrupt (ILVRTCA_0)
ITCLV3 (ITC Interrupt Level Setup Register 3)	15–11	–	0x00	–	R	–
	10–8	ILV7[2:0]	0x0	H0	R/W	16-bit timer Ch.1 interrupt (ILVT16_1)
	7–3	–	0x00	–	R	–
	2–0	ILV6[2:0]	0x0	H0	R/W	UART Ch.0 interrupt (ILVUART_0)
ITCLV4 (ITC Interrupt Level Setup Register 4)	15–11	–	0x00	–	R	–
	10–8	ILV9[2:0]	0x0	H0	R/W	I^2C interrupt (ILVI2C_0)
	7–3	–	0x00	–	R	–
	2–0	ILV8[2:0]	0x0	H0	R/W	Synchronous serial interface Ch.0 interrupt (ILVSPIA_0)
ITCLV5 (ITC Interrupt Level Setup Register 5)	15–11	–	0x00	–	R	–
	10–8	ILV11[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.1 interrupt (ILVT16B_1)
	7–3	–	0x00	–	R	–
	2–0	ILV10[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.0 interrupt (ILVT16B_0)
ITCLV6 (ITC Interrupt Level Setup Register 6)	15–11	–	0x00	–	R	–
	10–8	ILV13[2:0]	0x0	H0	R/W	16-bit timer Ch.2 interrupt (ILVT16_2)
	7–3	–	0x00	–	R	–
	2–0	ILV12[2:0]	0x0	H0	R/W	UART Ch.1 interrupt (ILVUART_1)
ITCLV7 (ITC Interrupt Level Setup Register 7)	15–11	–	0x00	–	R	–
	10–8	ILV15[2:0]	0x0	H0	R/W	LCD driver interrupt (ILVLCD8B)
	7–3	–	0x00	–	R	–
	2–0	ILV14[2:0]	0x0	H0	R/W	Sound generator interrupt (ILVSNDA_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)

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ITCLV9 (ITC Interrupt Level Setup Register 9)	15–11	–	0x00	–	R	–
	10–8	ILV19[2:0]	0x0	H0	R/W	R/F converter Ch.3 interrupt (ILVRFC_3)
	7–3	–	0x00	–	R	–
	2–0	ILV18[2:0]	0x0	H0	R/W	R/F converter Ch.2 interrupt (ILVRFC_2)

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.

Table 6.1.1 Port Configuration of S1C17W15

Item	S1C17W15 100-pin package/chip	S1C17W15 80-pin package	S1C17W15 64-pin package
Port groups included	P0[6:0], P1[7:0], P2[7:0], P3[7:0], Pd[4:0]	P0[6:0], P1[7:0], P2[7:0], P3[4:0], Pd[4:0]	P0[6:0], P1[7:0], P2[7:0], Pd[4:0]
Ports with general-purpose I/O function (GPIO)	P0[6:0], P1[7:0], P2[7:0], P3[7:0], Pd[4:0] (Pd2: output only)	P0[6:0], P1[7:0], P2[7:0], P3[4:0], Pd[4:0] (Pd2: output only)	P0[6:0], P1[7:0], P2[7:0], Pd[4:0] (Pd2: output only)
Ports with chattering filter function	P0[6:0], P1[7:0]	P0[6:0], P1[7:0]	P0[6:0], P1[7:0]
Ports with interrupt function	P0[6:0], P1[7:0], P2[7:0], P3[7:0]	P0[6:0], P1[7:0], P2[7:0], P3[4:0]	P0[6:0], P1[7:0], P2[7:0]
Ports for debug function	Pd[2:0]		
Key-entry reset function	Supported (P0[3:0])		

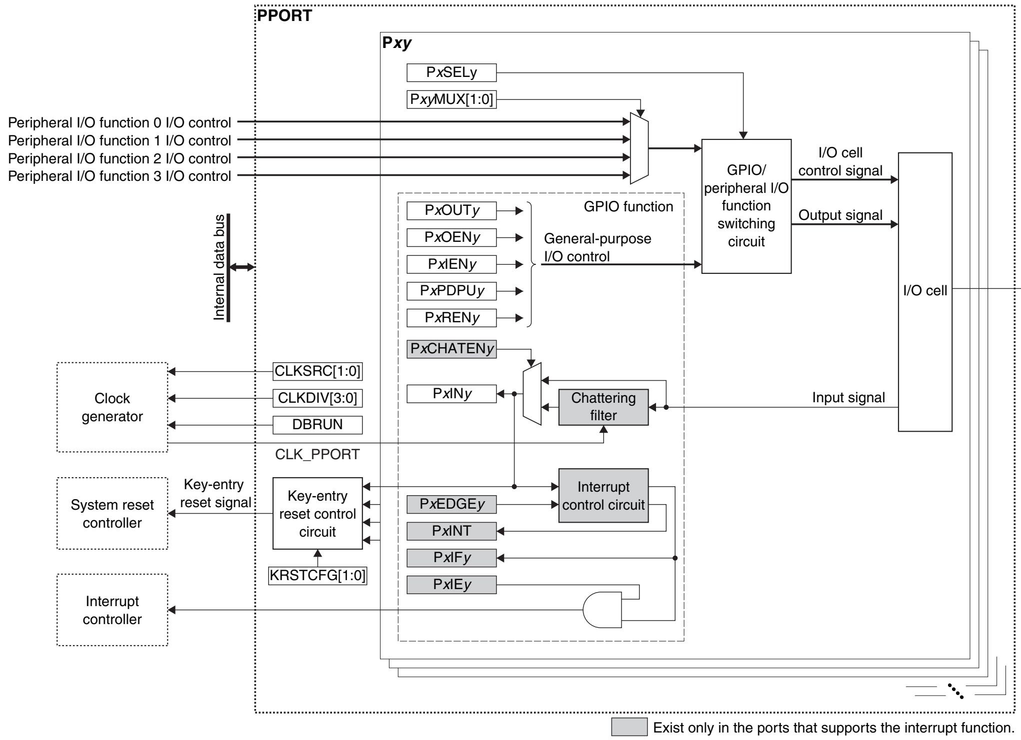


Figure 6.1.1 PPORT Configuration

6.2 I/O Cell Structure and Functions

Figure 6.2.1 shows the I/O cell Configuration.

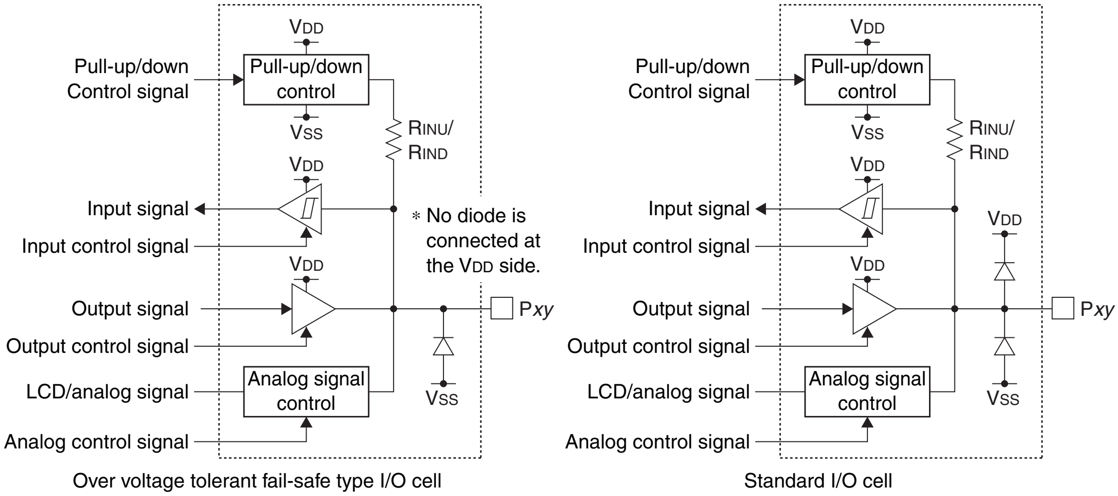


Figure 6.2.1 I/O Cell Configuration

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 not 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

P*x*IOEN. P*x*IEN*y* bit	P*x*IOEN. P*x*OEN*y* bit	P*x*RCTL. P*x*REN*y* bit	P*x*RCTL. P*x*PDPU*y* 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 P*x*MODSEL.P*x*SEL*y* 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 P*x*DAT.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 P*x*DAT.P*x*IN*y* bit.

Note: The P*x*DAT.P*x*IN*y* bit retains the input port status at 1 clock before being read from the CPU.

Chattering filter function

Some ports have a chattering filter function and it can be controlled in each port. This function is enabled by setting the P*x*CHATEN.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
P*x*IOEN	15–8	P*x*IEN[7:0]	0x00	H0	R/W	–
	7–0	P*x*OEN[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
P*x*RCTL	15–8	P*x*PDPU[7:0]	0x00	H0	R/W	–
	7–0	P*x*REN[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
	IOSC	OSC1	OSC3	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 P*x*INT

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 and interrupt 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	–	0	–	R	–
	14–8	P0OUT[6:0]	0x00	H0	R/W	
	7	–	0	–	R	
	6–0	P0IN[6:0]	0x00	H0	R	
P0IOEN (P0 Port Enable Register)	15	–	0	–	R	–
	14–8	P0IEN[6:0]	0x00	H0	R/W	
	7	–	0	–	R	
	6–0	P0OEN[6:0]	0x00	H0	R/W	
P0RCTL (P0 Port Pull-up/down Control Register)	15	–	0	–	R	–
	14–8	P0PDPU[6:0]	0x00	H0	R/W	
	7	–	0	–	R	
	6–0	P0REN[6:0]	0x00	H0	R/W	
P0INTF (P0 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6–0	P0IF[6:0]	0x00	H0	R/W	Cleared by writing 1.
P0INTCTL (P0 Port Interrupt Control Register)	15	–	0	–	R	–
	14–8	P0EDGE[6:0]	0x00	H0	R/W	
	7	–	0	–	R	
	6–0	P0IE[6:0]	0x00	H0	R/W	
P0CHATEN (P0 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6–0	P0CHATEN[6:0]	0x00	H0	R/W	
P0MODSEL (P0 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6–0	P0SEL[6:0]	0x00	H0	R/W	
P0FNCSEL (P0 Port Function Select Register)	15–14	–	0x0	–	R	–
	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	RFC Ch.0	SENB0	UPMUX	∗1	–	–	–	–
P01	P01	RFC Ch.0	SENA0	UPMUX	∗1	–	–	–	–
P02	P02	RFC Ch.0	REF0	UPMUX	∗1	–	–	–	–
P03	P03	RFC Ch.0	RFIN0	UPMUX	∗1	–	–	–	–
P04	P04	RTCA	RTC1S	UPMUX	∗1	–	–	–	–
P05	P05	T16B Ch.0	EXCL01	UPMUX	∗1	–	–	–	–
P06	P06	T16B Ch.1	EXCL10	UPMUX	∗1	–	–	–	–

∗1: Refer to the “Universal Port Multiplexer” chapter.

6.7.2 P1 Port Group

The P1 port group supports the GPIO and interrupt 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]	0x00	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	–
	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.1	SENB1	UPMUX	*1	–	–	–	–
P11	P11	RFC Ch.1	SENA1	UPMUX	*1	–	–	–	–
P12	P12	RFC Ch.1	REF1	UPMUX	*1	–	–	–	–
P13	P13	RFC Ch.1	RFIN1	UPMUX	*1	–	–	–	–
P14	P14	SNDA	#BZOUT	UPMUX	*1	–	–	–	–
P15	P15	SNDA	BZOUT	UPMUX	*1	–	–	–	–
P16	P16	CLG	FOUT	UPMUX	*1	–	–	–	–
P17	P17	T16B Ch.1	EXCL11	UPMUX	*1	–	–	–	–

*1: Refer to the "Universal Port Multiplexer" chapter.

6.7.3 P2 Port Group

The P2 port group supports the GPIO and interrupt functions.

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]	0x00	H0	R	

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P2IOEN (P2 Port Enable Register)	15–8	P2IEN[7:0]	0x00	H0	R/W	–
	7–0	P2OEN[7:0]	0x00	H0	R/W	
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 (P2 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
	7–0	P2IF[7:0]	0x00	H0	R/W	Cleared by writing 1.
P2INTCTL (P2 Port Interrupt Control Register)	15–8	P2EDGE[7:0]	0x00	H0	R/W	–
	7–0	P2IE[7:0]	0x00	H0	R/W	
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/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	

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	RFC Ch.2	SENB2	UPMUX	*1	–	–	LCD8B	SEG23
P21	P21	RFC Ch.2	SENA2	UPMUX	*1	–	–	LCD8B	SEG22
P22	P22	RFC Ch.2	REF2	UPMUX	*1	–	–	LCD8B	SEG21
P23	P23	RFC Ch.2	RFIN2	UPMUX	*1	–	–	LCD8B	SEG20
P24	P24	RFC Ch.3	SENB3	UPMUX	*1	–	–	LCD8B	SEG19
P25	P25	RFC Ch.3	SENA3	UPMUX	*1	–	–	LCD8B	SEG18
P26	P26	RFC Ch.3	REF3	UPMUX	*1	–	–	LCD8B	SEG17
P27	P27	RFC Ch.3	RFIN3	UPMUX	*1	–	–	LCD8B	SEG16

*1: Refer to the “Universal Port Multiplexer” chapter.

6.7.4 P3 Port Group

The P3 port group supports the GPIO and interrupt functions.

Note: The P3 port group does not exist in the 64-pin package.

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	P3OUT[7:5] and P3IN[7:5] are reserved bits in the 80-pin package.
	7–0	P3IN[7:0]	0x00	H0	R	
P3IOEN (P3 Port Enable Register)	15–8	P3IEN[7:0]	0x00	H0	R/W	P3IEN[7:5] and P3OEN[7:5] are reserved bits in the 80-pin package.
	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	P3PDPU[7:5] and P3REN[7:5] are reserved bits in the 80-pin package.
	7–0	P3REN[7:0]	0x00	H0	R/W	
P3INTF (P3 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
	7–0	P3IF[7:0]	0x00	H0	R/W	Cleared by writing 1. P3IF[7:5] are reserved bits in the 80-pin package.

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P3INTCTL (P3 Port Interrupt Control Register)	15–8	P3EDGE[7:0]	0x00	H0	R/W	P3EDGE[7:5] and P3IE[7:5] are reserved bits in the 80-pin package.
	7–0	P3IE[7:0]	0x00	H0	R/W	
P3CHATEN	15–0	–	0x0000	–	R	–
P3MODSEL (P3 Port Mode Select Register)	15–8	–	0x00	–	R	–
	7–0	P3SEL[7:0]	0x00	H0	R/W	P3SEL[7:5] are reserved bits in the 80-pin package.
P3FNCSEL	15–0	–	0xaaaa	–	R	–

Table 6.7.4.2 P3 Port Group Function Assignment

Port name	P3SELy = 0	P3SELy = 1							
	GPIO	P3yMUX = 0x0 (Function 0)		P3yMUX = 0x1 (Function 1)		P3yMUX = 0x2 (Function 2)		P3yMUX = 0x3 (Function 3)	
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin
P30	P30	–	–	–	–	–	–	–	–
P31	P31	–	–	–	–	–	–	–	–
P32	P32	–	–	–	–	–	–	–	–
P33	P33	–	–	–	–	–	–	–	–
P34	P34	–	–	–	–	–	–	–	–
P35 *	P35	–	–	–	–	–	–	–	–
P36 *	P36	–	–	–	–	–	–	–	–
P37 *	P37	–	–	–	–	–	–	–	–

* These ports are available only in the 100-pin package/chip.

6.7.5 Pd Port Group

The Pd port group consists of five ports Pd0–Pd4 and three ports Pd0–Pd2 are configured as a debugging function port at initialization. These five 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 6.7.5.1 Control Registers for Pd Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PDDAT (Pd Port Data Register)	15–13	–	0x0	–	R	–
	12–8	PDOUT[4:0]	0x00	H0	R/W	
	7–5	–	0x0	–	R	
	4–3	PDIN[4:3]	x	H0	R	
	2	–	0	–	R	
	1–0	PDIN[1:0]	x	H0	R	
PDIOEN (Pd Port Enable Register)	15–13	–	0x0	–	R	–
	12–11	PDIEN[4:3]	0x0	H0	R/W	
	10	(reserved)	0	H0	R/W	
	9–8	PDIEN[1:0]	0x0	H0	R/W	
	7–5	–	0x0	–	R	
	4–0	PDOEN[4:0]	0x00	H0	R/W	
PDRCTL (Pd Port Pull-up/down Control Register)	15–13	–	0x0	–	R	–
	12–11	PDPDPU[4:3]	0x0	H0	R/W	
	10	(reserved)	0	H0	R/W	
	9–8	PDPDPU[1:0]	0x0	H0	R/W	
	7–5	–	0x0	–	R	
	4–3	PDREN[4:3]	0x0	H0	R/W	
	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	–
	7–5	–	0x0	–	R	
	4–0	PDSEL[4:0]	0x07	H0	R/W	

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PDFNCSEL (Pd Port Function Select Register)	15–10	–	0x00	–	R	–
	9–8	PD4MUX[1:0]	0x0	H0	R/W	
	7–6	PD3MUX[1:0]	0x0	H0	R/W	
	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	

Table 6.7.5.2 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	–	–	–	–	–	–
Pd3	Pd3	CLG	EXOSC	T16B Ch.0	EXCL00	CLG	OSC3	–	–
Pd4	Pd4	–	–	–	–	CLG	OSC4	–	–

6.7.6 Common Registers between Port Groups

Table 6.7.6.1 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	–
	7–4	–	0x0	–	R	
	3	P3INT	0	H0	R	Reserved bit in the 64-pin package
	2	P2INT	0	H0	R	–
	1	P1INT	0	H0	R	
	0	P0INT	0	H0	R	

7 Universal Port Multiplexer (UPMUX)

7.1 Overview

UPMUX is a multiplexer that allows software to assign the desired peripheral I/O function to an I/O port. The main features are outlined below.

- Allows programmable assignment of the synchronous serial interface, I^{2}C, UART, and 16-bit PWM timer peripheral I/O functions to the P0[6:0], P1[7:0], and P2[7:0] ports.
- The peripheral I/O function assigned via UPMUX is enabled by setting the P*x*FNCSEL.P*xy*MUX[1:0] bits to 0x1.

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) and '*y*' refers to a port number (*y* = 0, 1, 2, ··· , 7).

Figure 7.1.1 shows the configuration of UPMUX.

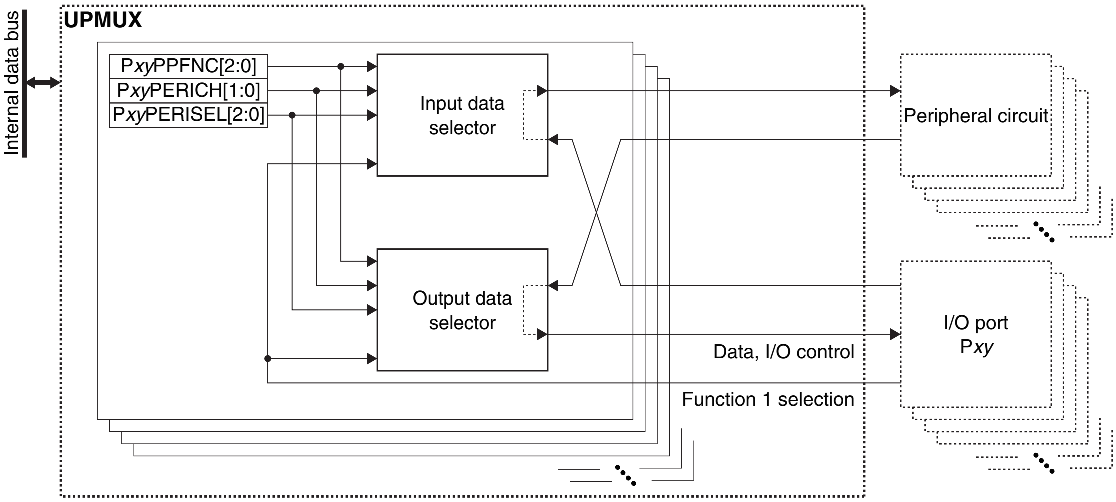


Figure 7.1.1 UPMUX Configuration

7.2 Peripheral Circuit I/O Function Assignment

An I/O function of a peripheral circuit supported may be assigned to peripheral I/O function 1 of an I/O port listed above. The following shows the procedure to assign a peripheral I/O function and enable it in the I/O port:

1. Configure the P*x*IOEN register of the I/O port.
   - 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 of the I/O port to 0. (Disable peripheral I/O function)
3. Set the following P*x*UPMUX*n* register bits (*n* = 0 to 3).
   - P*x*UPMUX*n*.P*xy*PERISEL[2:0] bits (Select peripheral circuit)
   - P*x*UPMUX*n*.P*xy*PERICH[1:0] bits (Select peripheral circuit channel)
   - P*x*UPMUX*n*.P*xy*PPFNC[2:0] bits (Select function to assign)
4. Initialize the peripheral circuit.
5. Set the P*x*FNCSEL.P*xy*MUX[1:0] bits of the I/O port to 0x1. (Select peripheral I/O function 1)
6. Set the P*x*MODSEL.P*x*SEL*y* bit of the I/O port to 1. (Enable peripheral I/O function)

7.3 Control Registers

P*xy*–*xz* Universal Port Multiplexer Setting Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
P*x*UPMUX*n*	15–13	P*xz*PPFNC[2:0]	0x0	H0	R/W	–
	12–11	P*xz*PERICH[1:0]	0x0	H0	R/W	
	10–8	P*xz*PERISEL[2:0]	0x0	H0	R/W	
	7–5	P*xy*PPFNC[2:0]	0x0	H0	R/W	
	4–3	P*xy*PERICH[1:0]	0x0	H0	R/W	
	2–0	P*xy*PERISEL[2:0]	0x0	H0	R/W	

∗1: ‘*x*’ in the register name refers to a port group number and ‘*n*’ refers to a register number (0–2).
∗2: ‘*x*’ in the bit name refers to a port group number, ‘*y*’ refers to an even port number (0, 2, 4, 6), and ‘*z*’ refers to an odd port number (*z* = *y* + 1).

Bits 15–13 P*xz*PPFNC[2:0]
Bits 7–5 P*xy*PPFNC[2:0]

These bits specify the peripheral I/O function to be assigned to the port. (See Table 7.3.1.)

Bits 12–11 P*xz*PERICH[1:0]
Bits 4–3 P*xy*PERICH[1:0]

These bits specify a peripheral circuit channel number. (See Table 7.3.1.)

Bits 10–8 P*xz*PERISEL[2:0]
Bits 2–0 P*xy*PERISEL[2:0]

These bits specify a peripheral circuit. (See Table 7.3.1.)

Table 7.3.1 Peripheral I/O Function Selections

<table>
<tr><th rowspan="6">PxUPMUXn.
PxyPPFNC[2:0]
bits
(Peripheral I/O
function)</th><th colspan="8">PxUPMUXn.PxyPERISEL[2:0] bits (Peripheral circuit)</th></tr>
<tr><td>0x0</td><td>0x1</td><td>0x2</td><td>0x3</td><td>0x4</td><td>0x5</td><td>0x6</td><td>0x7</td></tr>
<tr><td>None ∗</td><td>I2C</td><td>SPIA</td><td>UART</td><td>T16B</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><th colspan="8">PxUPMUXn.PxyPERICH[1:0] bits (Peripheral circuit channel)</th></tr>
<tr><td>–</td><td>0x0</td><td>0x0</td><td>0x0, 0x1</td><td>0x0, 0x1</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>–</td><td>Ch.0</td><td>Ch.0</td><td>Ch.0, 1</td><td>Ch.0, 1</td><td>–</td><td>–</td><td>–</td></tr>
<tr><td>0x0</td><td>None ∗</td><td>None ∗</td><td>None ∗</td><td>None ∗</td><td>None ∗</td><td>None ∗</td><td>None ∗</td><td>None ∗</td></tr>
<tr><td>0x1</td><td rowspan="7">Reserved</td><td>SCLn</td><td>SDIn</td><td>USINn</td><td>TOUTn0/
CAPn0</td><td rowspan="7">Reserved</td><td rowspan="7">Reserved</td><td rowspan="7">Reserved</td></tr>
<tr><td>0x2</td><td>SDAn</td><td>SDOn</td><td>USOUTn</td><td>TOUTn1/
CAPn1</td></tr>
<tr><td>0x3</td><td rowspan="5">Reserved</td><td>SPICLKn</td><td rowspan="5">Reserved</td><td rowspan="5">Reserved</td></tr>
<tr><td>0x4</td><td>#SPISSn</td></tr>
<tr><td>0x5</td><td rowspan="3">Reserved</td></tr>
<tr><td>0x6</td></tr>
<tr><td>0x7</td></tr>
</table>

∗ “None” means no assignment. Selecting this will put the P*xy* pin into Hi-Z status when peripheral I/O function 1 is selected and enabled in the I/O port.

Note: Do not assign a peripheral input function to two or more I/O ports. Although the I/O ports output the same waveforms when an output function is assigned to two or more I/O port, a skew occurs due to the internal delay.

8 Watchdog Timer (WDT)

8.1 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 8.1.1 shows the configuration of WDT.

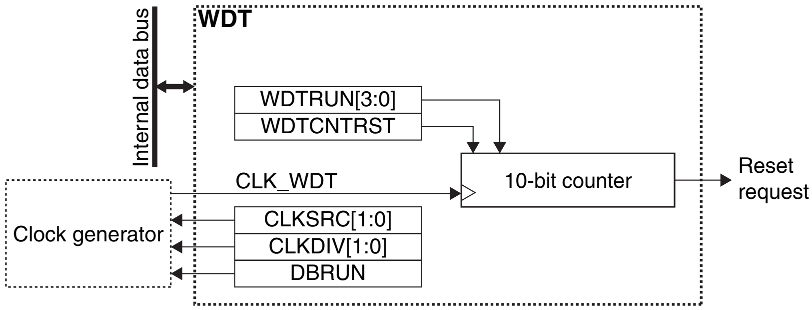


Figure 8.1.1 WDT Configuration

8.2 Clock Settings

8.2.1 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).

$$t_{WDT} = \frac{1{,}024}{CLK\_WDT} \quad \text{(Eq. 8.1)}$$

Where

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

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

8.2.2 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.

8.3 Operations

8.3.1 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.

8.3.2 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.

8.4 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 8.4.1 Clock Source and Division Ratio Settings

WDTCLK. CLKDIV[1:0] bits	WDTCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x3	1/16,384	1/128	1/16,384	1/1
0x2	1/8,192		1/8,192	
0x1	1/4,096		1/4,096	
0x0	1/2,048		1/2,048	

(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 (WP):	Stop
Values other than 0xa (WP):	Run
0xa (R):	Idle
0x0 (R):	Running

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.

9 Real-Time Clock (RTCA)

9.1 Overview

RTCA is a real-time clock with a perpetual calendar function. The main features of RTCA are outlined below.

- Includes a BCD real-time clock counter to implement a time-of-day clock (second, minute, and hour) and calendar (day, day of the week, month, and year with leap year supported).
- Provides a hold function for reading correct counter values by suspending the real-time clock counter operation.
- 24-hour or 12-hour mode is selectable.
- Capable of controlling the starting and stopping of the time-of-day clock.
- Provides a 30-second correction function to adjust time using a time signal.
- Includes a 1 Hz counter to count 128 to 1 Hz.
- Includes a BCD stopwatch counter with 1/100-second counting supported.
- Provides a theoretical regulation function to correct clock error due to frequency tolerance with no external parts required.

Figure 9.1.1 shows the configuration of RTCA.

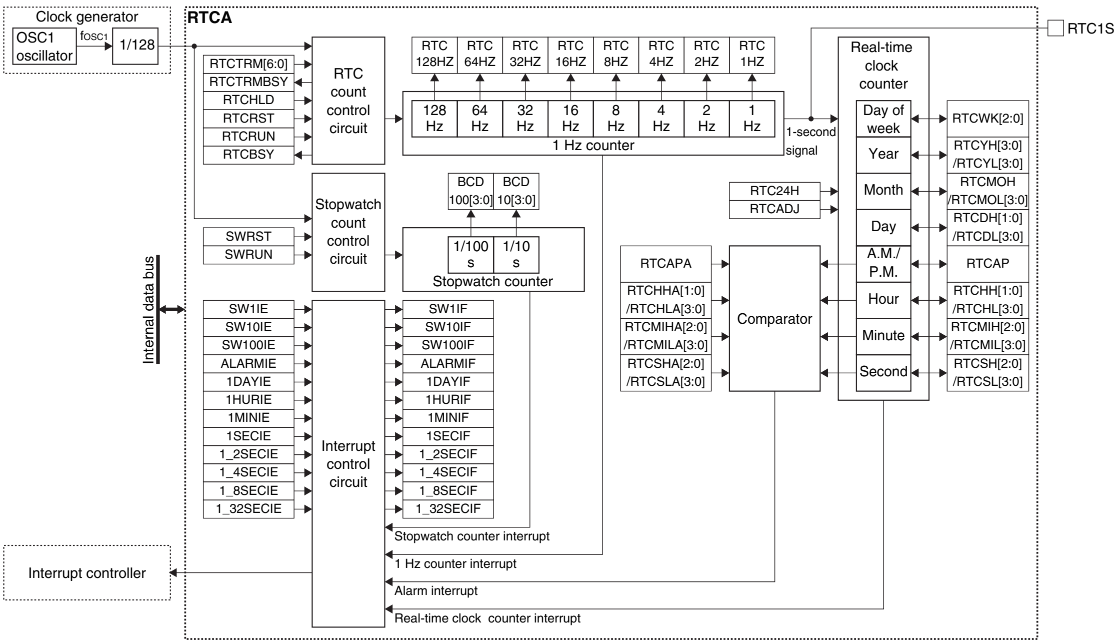


Figure 9.1.1 RTCA Configuration

9.2 Output Pin and External Connection

9.2.1 Output Pin

Table 9.2.1.1 shows the RTCA pin.

Table 9.2.1.1 RTCA Pin

Pin name	I/O*	Initial status*	Function
RTC1S	O	O (L)	1-second signal monitor output pin

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

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

9.3 Clock Settings

9.3.1 RTCA Operating Clock

RTCA uses CLK_RTCA, which is generated by the clock generator from OSC1 as the clock source, as its operating clock. RTCA is operable when OSC1 is enabled.
To continue the RTCA operation during SLEEP mode with OSC1 being activated, the CLGOSC.OSC1SLPC bit must be set to 0.

9.3.2 Theoretical Regulation Function

The time-of-day clock loses accuracy if the OSC1 frequency f_{OSC1} has a frequency tolerance from 32.768 kHz. To correct this error without changing any external part, RTCA provides a theoretical regulation function. Follow the procedure below to perform theoretical regulation.

1. Measure f_{OSC1} and calculate the frequency tolerance correction value "m [ppm] = -{(f_{OSC1} - 32,768 [Hz]) / 32,768 [Hz]} × 10^6."
2. Determine the theoretical regulation execution cycle time "n seconds."
3. Determine the value to be written to the RTCCTL.RTCTRM[6:0] bits from the results in Steps 1 and 2.
4. Write the value determined in Step 3 to the RTCCTL.RTCTRM[6:0] bits periodically in n-second cycles using an RTCA alarm or second interrupt.
5. Monitor the RTC1S signal to check that every n-second cycle has no error included.

The correction value for theoretical regulation can be specified within the range from -64 to +63 and it should be written to the RTCCTL.RTCTRM[6:0] bits as a two's-complement number. Use Eq. 9.1 to calculate the correction value.

$$\text{RTCTRM[6:0]} = \frac{m}{10^6} \times 256 \times n \quad \text{(However, RTCTRM[6:0] is an integer after rounding off to -64 to +63.)} \qquad \text{(Eq. 9.1)}$$

Where
n: Theoretical regulation execution cycle time [second] (time interval to write the correct value to the RTCCTL.RTCTRM[6:0] bits periodically via software)
m: OSC1 frequency tolerance correction value [ppm]

Figure 9.3.2.1 shows the RTC1S signal waveform.

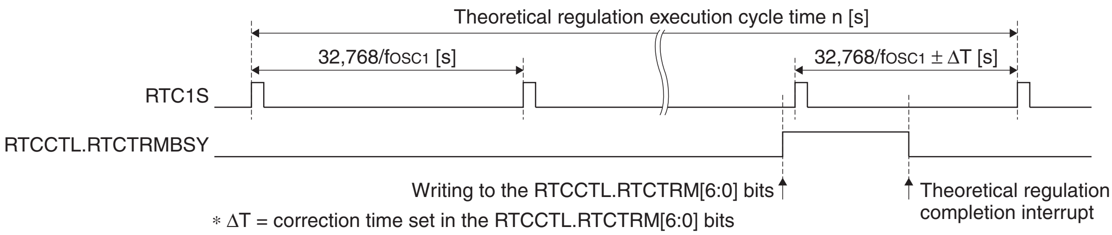


Figure 9.3.2.1 RTC1S Signal Waveform

Table 9.3.2.1 lists the frequency tolerance correction rates when the theoretical regulation execution cycle time n is 4,096 seconds as an example.

Table 9.3.2.1 Correction Rates when Theoretical Regulation Execution Cycle Time n = 4,096 Seconds

RTCCTL.RTCTRM[6:0] bits (two's-complement)	Correction value (decimal)	Correction rate [ppm]	RTCCTL.RTCTRM[6:0] bits (two's-complement)	Correction value (decimal)	Correction rate [ppm]
0x00	0	0.0	0x40	-64	-61.0
0x01	1	1.0	0x41	-63	-60.1
0x02	2	1.9	0x42	-62	-59.1
0x03	3	2.9	0x43	-61	-58.2
...	...	...	...	...	...
0x3e	62	59.1	0x7e	-2	-1.9
0x3f	63	60.1	0x7f	-1	-1.0

Minimum resolution: 1 ppm, Correction rate range: -61.0 to 60.1 ppm

Notes: • The theoretical regulation affects only the real-time clock counter and 1 Hz counter. It does not affect the stopwatch counter.

• After a value is written to the RTCCTL.RTCTRM[6:0] bits, the theoretical regulation correction takes effect on the 1 Hz counter value at the same timing as when the 1 Hz counter changes to 0x7f. Also an interrupt occurs depending on the counter value at this time.

9.4 Operations

9.4.1 RTCA Control

Follow the sequences shown below to set time to RTCA, to read the current time and to set alarm.

Time setting

1. Set RTCA to 12H or 24H mode using the RTCCTL.RTC24H bit.
2. Write 1 to the RTCCTL.RTCRUN bit to enable for the real-time clock counter to start counting up.
3. Check to see if the RTCCTL.RTCBSY bit = 0 that indicates the counter is ready to rewrite. If the RTCCTL.RTCBSY bit = 1, wait until it is set to 0.
4. Write the current date and time in BCD code to the control bits listed below.
   RTCSEC.RTCSH[2:0]/RTCSL[3:0] bits (second)
   RTCHUR.RTCMIH[2:0]/RTCMIL[3:0] bits (minute)
   RTCHUR.RTCHH[1:0]/RTCHL[3:0] bits (hour)
   RTCHUR.RTCAP bit (AM/PM) (effective when RTCCTL.RTC24H bit = 0)
   RTCMON.RTCDH[1:0]/RTCDL[3:0] bits (day)
   RTCMON.RTCMOH/RTCMOL[3:0] bits (month)
   RTCYAR.RTCYH[3:0]/RTCYL[3:0] bits (year)
   RTCYAR.RTCWK[2:0] bits (day of the week)
5 Write 1 to the RTCCTL.RTCADJ bit (execute 30-second correction) using a time signal to adjust the time. (For more information on the 30-second correction, refer to “Real-Time Clock Counter Operations.”)
6. Write 1 to the real-time clock counter interrupt flags in the RTCINTF register to clear them.
7. Write 1 to the interrupt enable bits in the RTCINTE register to enable real-time clock counter interrupts.

Time read

1. Check to see if the RTCCTL.RTCBSY bit = 0. If the RTCCTL.RTCBSY bit = 1, wait until it is set to 0.
2. Write 1 to the RTCCTL.RTCHLD bit to suspend count-up operation of the real-time clock counter.
3. Read the date and time from the control bits listed in “Time setting, Step 4” above.
4. Write 0 to the RTCCTL.RTCHLD bit to resume count-up operation of the real-time clock counter. If a second count-up timing has occurred in the count hold state, the hardware corrects the second counter for +1 second (for more information on the +1 second correction, refer to “Real-Time Clock Counter Operations”).

Alarm setting

1. Write 0 to the RTCINTE.ALARMIE bit to disable alarm interrupts.
2. Write the alarm time in BCD code to the control bits listed below (a time within 24 hours from the current time can be specified).
   RTCALM1.RTCSHA[2:0]/RTCSLA[3:0] bits (second)
   RTCALM2.RTCMIHA[2:0]/RTCMILA[3:0] bits (minute)
   RTCALM2.RTCHHA[1:0]/RTCHLA[3:0] bits (hour)
   RTCALM2.RTCAPA bit (AM/PM) (effective when RTCCTL.RTC24H bit = 0)
3. Write 1 to the RTCINTF.ALARMIF bit to clear the alarm interrupt flag.
4. Write 1 to the RTCINTE.ALARMIE bit to enable alarm interrupts.
   When the real-time clock counter reaches the alarm time set in Step 2, an alarm interrupt occurs.

9.4.2 Real-Time Clock Counter Operations

The real-time clock counter consists of second, minute, hour, AM/PM, day, month, year, and day of the week counters and it performs counting up using the RTC1S signal. It has the following functions as well.

Recognizing leap years

The leap year recognizing algorithm used in RTCA is effective only for Christian Era years. Years within 0 to 99 that can be divided by four without a remainder are recognized as leap years. If the year counter = 0x00, RTCA assumes it as a common year. If a leap year is recognized, the count range of the day counter changes when the month counter is set to February.

Corrective operation when a value out of the effective range is set

When a value out of the effective range is set to the year, day of the week, or hour (in 24H mode) counter, the counter will be cleared to 0 at the next count-up timing. When a such value is set to the month, day, or hour (in 12H mode) counter, the counter will be set to 1 at the next count-up timing.

Note: Do not set the RTCMON.RTCMOL[3:0] bits to 0x0 if the RTCMON.RTCMOH bit = 0.

30-second correction

This function is provided to set the time-of-day clock by the time signal. Writing 1 to the RTCCTL.RTCADJ bit clears the second counter and adds 1 to the minute counter if the second counter represents 30 to 59 seconds, or clears the second counter with the minute counter left unchanged if the second counter represents 0 to 29 seconds.

+1 second correction

If a second count-up timing occurred while the RTCCTL.RTCHLD bit = 1 (count hold state), the real-time clock counter counts up by +1 second (performs +1 second correction) after the counting has resumed by writing 0 to the RTCCTL.RTCHLD bit.

Note: If two or more second count-up timings occurred while the RTCCTL.RTCHLD bit = 1, the counter is always corrected for +1 second only.

9.4.3 Stopwatch Control

Follow the sequences shown below to start counting of the stopwatch and to read the counter.

Count start

1. Write 1 to the RTCSWCTL.SWRST bit to reset the stopwatch counter.
2. Write 1 to the stopwatch interrupt flags in the RTCINTF register to clear them.
3. Write 1 to the interrupt enable bits in the RTCINTE register to enable stopwatch interrupts.
4. Write 1 to the RTCSWCTL.SWRUN bit to start stopwatch count up operation.

Counter read

1. Read the count value from the RTCSWCTL.BCD10[3:0] and BCD100[3:0] 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.

9.4.4 Stopwatch Count-up Pattern

The stopwatch consists of 1/100-second and 1/10-second counters and these counters perform counting up in increments of approximate 1/100 and 1/10 seconds with the count-up patterns shown in Figure 9.4.4.1.

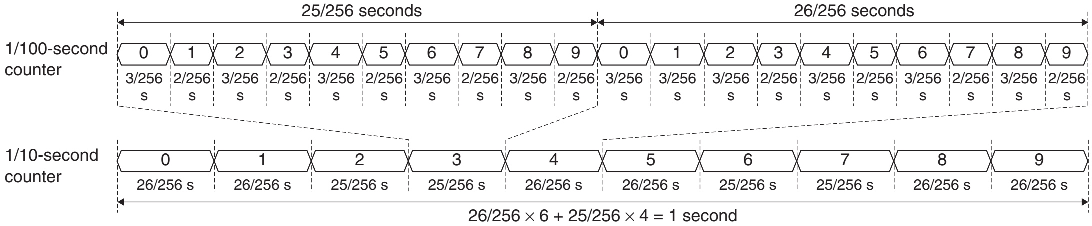


Figure 9.4.4.1 Stopwatch Count-Up Patterns

9.5 Interrupts

RTCA has a function to generate the interrupts shown in Table 9.5.1.

Table 9.5.1 RTCA Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Alarm	RTCINTF.ALARMIF	Matching between the RTCALM1–2 register contents and the real-time clock counter contents	Writing 1
1-day	RTCINTF.1DAYIF	Day counter count up	Writing 1
1-hour	RTCINTF.1HURIF	Hour counter count up	Writing 1
1-minute	RTCINTF.1MINIF	Minute counter count up	Writing 1
1-second	RTCINTF.1SECIF	Second counter count up	Writing 1
1/2-second	RTCINTF.1_2SECIF	See Figure 9.5.1.	Writing 1
1/4-second	RTCINTF.1_4SECIF	See Figure 9.5.1.	Writing 1
1/8-second	RTCINTF.1_8SECIF	See Figure 9.5.1.	Writing 1
1/32-second	RTCINTF.1_32SECIF	See Figure 9.5.1.	Writing 1
Stopwatch 1 Hz	RTCINTF.SW1IF	1/10-second counter overflow	Writing 1
Stopwatch 10 Hz	RTCINTF.SW10IF	1/10-second counter count up	Writing 1
Stopwatch 100 Hz	RTCINTF.SW100IF	1/100-second counter count up	Writing 1
Theoretical regulation completion	RTCINTF.RTCTRMIF	At the end of theoretical regulation operation	Writing 1

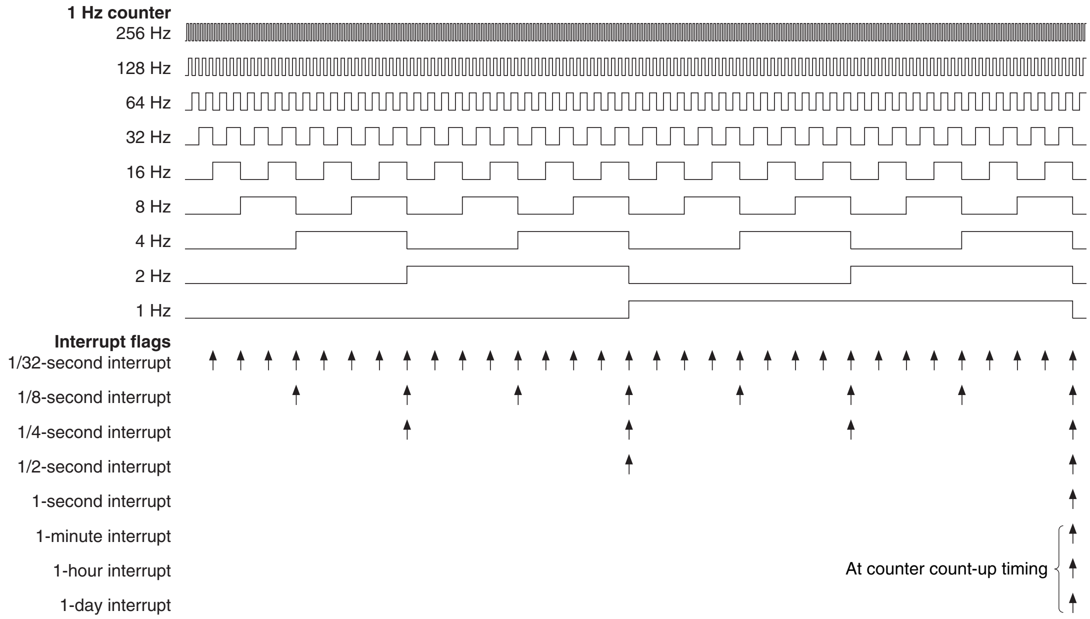


Figure 9.5.1 RTCA Interrupt Timings

Notes:
- 1-second to 1/32-second interrupts occur after a lapse of 1/256 second from change of the 1 Hz counter value.
- An alarm interrupt occurs after a lapse of 1/256 second from matching between the AM/PM (in 12H mode), hour, minute, and second counter value and the alarm setting value.

RTCA 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

RTC Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCCTL	15	RTCTRMBSY	0	H0	R	–
	14–8	RTCTRM[6:0]	0x00	H0	W	Read as 0x00.
	7	–	0	–	R	–
	6	RTCBSY	0	H0	R	
	5	RTCHLD	0	H0	R/W	Cleared by setting the RTCCTL.RTCRST bit to 1.
	4	RTC24H	0	H0	R/W	–
	3	–	0	–	R	
	2	RTCADJ	0	H0	R/W	Cleared by setting the RTCCTL.RTCRST bit to 1.
	1	RTCRST	0	H0	R/W	–
	0	RTCRUN	0	H0	R/W	

Bit 15 RTCTRMBSY

This bit indicates whether the theoretical regulation is currently executed or not.

1 (R): Theoretical regulation is executing.

0 (R): Theoretical regulation has finished (or not executed).

This bit goes 1 when a value is written to the RTCCTL.RTCTRM[6:0] bits. The theoretical regulation takes up to 1 second for execution. This bit reverts to 0 automatically after the theoretical regulation has finished execution.

Bits 14–8 RTCTRM[6:0]

Write the correction value for adjusting the 1 Hz frequency to these bits to execute theoretical regulation. For a calculation method of correction value, refer to “Theoretical Regulation Function.”

Notes:
- When the RTCCTL.RTCTRMBSY bit = 1, the RTCCTL.RTCTRM[6:0] bits cannot be rewritten.
- Writing 0x00 to the RTCCTL.RTCTRM[6:0] bits sets the RTCCTL.RTCTRMBSY bit to 1 as well. However, no correcting operation is performed.

Bit 7 Reserved

Bit 6 RTCBSY

This bit indicates whether the counter is performing count-up operation or not.

1 (R): In count-up operation

0 (R): Idle (ready to rewrite real-time clock counter)

This bit goes 1 when performing 1-second count-up, +1 second correction, or 30-second correction. It retains 1 for 1/256 second and then reverts to 0.

Bit 5 RTCHLD

This bit halts the count-up operation of the real-time clock counter.

1 (R/W): Halt real-time clock counter count-up operation

0 (R/W): Normal operation

Writing 1 to this bit halts the count-up operation of the real-time clock counter, this makes it possible to read the counter value correctly without changing the counter. Write 0 to this bit to resume count-up operation immediately after the counter has been read. Depending on these operation timings, the +1 second correction may be executed after the count-up operation resumes. For more information on the +1 second correction, refer to “Real-Time Clock Counter Operations.”

Note: When the RTCCTL.RTCTRMBSY bit = 1, the RTCCTL.RTCHLD bit cannot be rewritten to 1 (as fixed at 0).

Bit 4 RTC24H

This bit sets the hour counter to 24H mode or 12H mode.
1 (R/W): 24H mode
0 (R/W): 12H 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.

Bit 3 Reserved

Bit 2 RTCADJ

This bit executes the 30-second correction time adjustment function.
1 (W): Execute 30-second correction
0 (W): Ineffective
1 (R): 30-second correction is executing.
0 (R): 30-second correction has finished. (Normal operation)

Writing 1 to this bit executes 30-second correction and an enabled interrupt occurs even if the RTCCTL.RTCRUN bit = 0. The correction takes up to 2/256 seconds. The RTCCTL.RTCADJ bit is automatically cleared to 0 when the correction has finished. For more information on the 30-second correction, refer to "Real-Time Clock Counter Operations."

Notes:
- Be sure to avoid writing to this bit when the RTCCTL.RTCBSY bit = 1.
- Do not write 1 to this bit again while the RTCCTL.RTCADJ bit = 1.

Bit 1 RTCRST

This bit resets the 1 Hz counter, the RTCCTL.RTCADJ bit, and the RTCCTL.RTCHLD bit.
1 (W): Reset
0 (W): Ineffective
1 (R): Reset is being executed.
0 (R): Reset has finished. (Normal operation)

This bit is automatically cleared to 0 after reset has finished.

Bit 0 RTCRUN

This bit starts/stops the real-time clock counter.
1 (R/W): Running/start control
0 (R/W): Idle/stop control

When the real-time clock counter 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.

RTC Second Alarm Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCALM1	15	–	0	–	R	–
	14–12	RTCSHA[2:0]	0x0	H0	R/W	
	11–8	RTCSLA[3:0]	0x0	H0	R/W	
	7–0	–	0x00	–	R	

Bit 15 Reserved

Bits 14–12 RTCSHA[2:0]
Bits 11–8 RTCSLA[3:0]

The RTCALM1.RTCSHA[2:0] bits and the RTCALM1.RTCSLA[3:0] bits set the 10-second digit and 1-second digit of the alarm time, respectively. A value within 0 to 59 seconds can be set in BCD code as shown in Table 9.6.1.

Table 9.6.1 Setting Examples in BCD Code

Setting value in BCD code		Alarm (second) setting
RTCALM1.RTCSHA[2:0] bits	RTCALM1.RTCSLA[3:0] bits	
0x0	0x0	00 seconds
0x0	0x1	01 second
···	···	···
0x0	0x9	09 seconds
0x1	0x0	10 seconds
···	···	···
0x5	0x9	59 seconds

Bits 7–0 Reserved

RTC Hour/Minute Alarm Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCALM2	15	–	0	–	R	–
	14	RTCAPA	0	H0	R/W	
	13–12	RTCHHA[1:0]	0x0	H0	R/W	
	11–8	RTCHLA[3:0]	0x0	H0	R/W	
	7	–	0	–	R	
	6–4	RTCMIHA[2:0]	0x0	H0	R/W	
	3–0	RTCMILA[3:0]	0x0	H0	R/W	

Bit 15 Reserved

Bit 14 RTCAPA

This bit sets A.M. or P.M. of the alarm time in 12H mode (RTCCTL.RTC24H bit = 0).

1 (R/W): P.M.

0 (R/W): A.M.

This setting is ineffective in 24H mode (RTCCTL.RTC24H bit = 1).

Bits 13–12 RTCHHA[1:0]

Bits 11–8 RTCHLA[3:0]

The RTCALM2.RTCHHA[1:0] bits and the RTCALM2.RTCHLA[3:0] bits set the 10-hour digit and 1-hour digit of the alarm time, respectively. A value within 1 to 12 o'clock in 12H mode or 0 to 23 in 24H mode can be set in BCD code.

Bit 7 Reserved

Bits 6–4 RTCMIHA[2:0]

Bits 3–0 RTCMILA[3:0]

The RTCALM2.RTCMIHA[2:0] bits and the RTCALM2.RTCMILA[3:0] bits set the 10-minute digit and 1-minute digit of the alarm time, respectively. A value within 0 to 59 minutes can be set in BCD code.

RTC Stopwatch Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCSWCTL	15–12	BCD10[3:0]	0x0	H0	R	–
	11–8	BCD100[3:0]	0x0	H0	R	
	7–5	–	0x0	–	R	
	4	SWRST	0	H0	W	Read as 0.
	3–1	–	0x0	–	R	–
	0	SWRUN	0	H0	R/W	

Bits 15–12 BCD10[3:0]

Bits 11–8 BCD100[3:0]

The 1/10-second and 1/100-second digits of the stopwatch counter can be read as a BCD code from the RTCSWCTL.BCD10[3:0] bits and the RTCSWCTL.BCD100[3:0] bits, respectively.

Note: The counter value may not be read correctly while the stopwatch counter is running. The RTCSWCTL.BCD10[3:0]/BCD100[3:0] bits must be read twice and assume the counter value was read successfully if the two read results are the same.

Bits 7–5 Reserved

Bit 4 SWRST

This bit resets the stopwatch counter to 0x00.

1 (W): Reset
0 (W): Ineffective
0 (R): Always 0 when being read

When the stopwatch counter in running status is reset, it continues counting from count 0x00. The stopwatch counter retains 0x00 if it is reset in idle status.

Bits 3–1 Reserved

Bit 0 SWRUN

This bit starts/stops the stopwatch counter.

1 (R/W): Running/start control
0 (R/W): Idle/stop control

When the stopwatch counter 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.

Note: The stopwatch counter stops in sync with the stopwatch clock after 0 is written to the RTCSWCTL.SWRUN bit. Therefore, the counter value may be incremented (+1) from the value at writing 0.

RTC Second/1Hz Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCSEC	15	–	0	–	R	–
	14–12	RTCSH[2:0]	0x0	H0	R/W	
	11–8	RTCSL[3:0]	0x0	H0	R/W	
	7	RTC1HZ	0	H0	R	Cleared by setting the RTCCTL.RTCRST bit to 1.
	6	RTC2HZ	0	H0	R	
	5	RTC4HZ	0	H0	R	
	4	RTC8HZ	0	H0	R	
	3	RTC16HZ	0	H0	R	
	2	RTC32HZ	0	H0	R	
	1	RTC64HZ	0	H0	R	
	0	RTC128HZ	0	H0	R	

Bit 15 Reserved

Bits 14–12 RTCSH[2:0]
Bits 11–8 RTCSL[3:0]

The RTCSEC.RTCSH[2:0] bits and the RTCSEC.RTCSL[3:0] bits are used to set and read the 10-second digit and the 1-second digit of the second counter, respectively. The setting/read values are a BCD code within the range from 0 to 59.

Note: Be sure to avoid writing to the RTCSEC.RTCSH[2:0]/RTCSL[3:0] bits while the RTCCTL.RTCBSY bit = 1.

Bit 7 RTC1HZ
Bit 6 RTC2HZ
Bit 5 RTC4HZ
Bit 4 RTC8HZ
Bit 3 RTC16HZ
Bit 2 RTC32HZ
Bit 1 RTC64HZ
Bit 0 RTC128HZ

1 Hz counter data can be read from these bits.
The following shows the correspondence between the bit and frequency:
RTCSEC.RTC1HZ bit: 1 Hz
RTCSEC.RTC2HZ bit: 2 Hz
RTCSEC.RTC4HZ bit: 4 Hz
RTCSEC.RTC8HZ bit: 8 Hz
RTCSEC.RTC16HZ bit: 16 Hz
RTCSEC.RTC32HZ bit: 32 Hz
RTCSEC.RTC64HZ bit: 64 Hz
RTCSEC.RTC128HZ bit: 128 Hz

Note: The counter value may not be read correctly while the 1 Hz counter is running. These bits must be read twice and assume the counter value was read successfully if the two read results are the same.

RTC Hour/Minute Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCHUR	15	–	0	–	R	–
	14	RTCAP	0	H0	R/W	
	13–12	RTCHH[1:0]	0x1	H0	R/W	
	11–8	RTCHL[3:0]	0x2	H0	R/W	
	7	–	0	–	R	
	6–4	RTCMIH[2:0]	0x0	H0	R/W	
	3–0	RTCMIL[3:0]	0x0	H0	R/W	

Bit 15 Reserved

Bit 14 RTCAP

This bit is used to set and read A.M. or P.M. data in 12H mode (RTCCTL.RTC24H bit = 0).
1 (R/W): P.M.
0 (R/W): A.M.

In 24H mode (RTCCTL.RTC24H bit = 1), this bit is fixed at 0 and writing 1 is ignored. However, if the RTCHUR.RTCAP bit = 1 when changed to 24H mode, it goes 0 at the next count-up timing of the hour counter.

Bits 13–12 RTCHH[1:0]
Bits 11–8 RTCHL[3:0]

The RTCHUR.RTCHH[1:0] bits and the RTCHUR.RTCHL[3:0] bits are used to set and read the 10-hour digit and the 1-hour digit of the hour counter, respectively. The setting/read values are a BCD code within the range from 1 to 12 in 12H mode or 0 to 23 in 24H mode.

Note: Be sure to avoid writing to the RTCHUR.RTCHH[1:0]/RTCHL[3:0] bits while the RTCCTL.RTCBSY bit = 1.

Bit 7 Reserved

Bits 6–4 RTCMIH[2:0]
Bits 3–0 RTCMIL[3:0]

The RTCHUR.RTCMIH[2:0] bits and the RTCHUR.RTCMIL[3:0] bits are used to set and read the 10-minute digit and the 1-minute digit of the minute counter, respectively. The setting/read values are a BCD code within the range from 0 to 59.

Note: Be sure to avoid writing to the RTCHUR.RTCMIH[2:0]/RTCMIL[3:0] bits while the RTCCTL.RTCBSY bit = 1.

RTC Month/Day Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCMON	15–13	–	0x0	–	R	–
	12	RTCMOH	0	H0	R/W	
	11–8	RTCMOL[3:0]	0x1	H0	R/W	
	7–6	–	0x0	–	R	
	5–4	RTCDH[1:0]	0x0	H0	R/W	
	3–0	RTCDL[3:0]	0x1	H0	R/W	

Bits 15–13 Reserved

Bit 12 RTCMOH
Bits 11–8 RTCMOL[3:0]

The RTCMON.RTCMOH bit and the RTCMON.RTCMOL[3:0] bits are used to set and read the 10-month digit and the 1-month digit of the month counter, respectively. The setting/read values are a BCD code within the range from 1 to 12.

Notes:
- Be sure to avoid writing to the RTCMON.RTCMOH/RTCMOL[3:0] bits while the RTCCTL.RTCBSY bit = 1.
- Be sure to avoid setting the RTCMON.RTCMOH/RTCMOL[3:0] bits to 0x00.

Bits 7–6 Reserved

Bits 5–4 RTCDH[1:0]
Bits 3–0 RTCDL[3:0]

The RTCMON.RTCDH[1:0] bits and the RTCMON.RTCDL[3:0] bits are used to set and read the 10-day digit and the 1-day digit of the day counter, respectively. The setting/read values are a BCD code within the range from 1 to 31 (to 28 for February in a common year, to 29 for February in a leap year, or to 30 for April/June/September/November).

Note: Be sure to avoid writing to the RTCMON.RTCDH[1:0]/RTCDL[3:0] bits while the RTCCTL.RTCBSY bit = 1.

RTC Year/Week Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCYAR	15–11	–	0x00	–	R	–
	10–8	RTCWK[2:0]	0x0	H0	R/W	
	7–4	RTCYH[3:0]	0x0	H0	R/W	
	3–0	RTCYL[3:0]	0x0	H0	R/W	

Bits 15–11 Reserved

Bits 10–8 RTCWK[2:0]

These bits are used to set and read day of the week.
The day of the week counter is a base-7 counter and the setting/read values are 0x0 to 0x6. Table 9.6.2 lists the correspondence between the count value and day of the week.

Table 9.6.2 Correspondence between the count value and day of the week

RTCYAR.RTCWK[2:0] bits	Day of the week
0x6	Saturday
0x5	Friday
0x4	Thursday
0x3	Wednesday
0x2	Tuesday
0x1	Monday
0x0	Sunday

Note: Be sure to avoid writing to the RTCYAR.RTCWK[2:0] bits while the RTCCTL.RTCBSY bit = 1.

Bits 7–4 RTCYH[3:0]
Bits 3–0 RTCYL[3:0]

The RTCYAR.RTCYH[3:0] bits and the RTCYAR.RTCYL[3:0] bits are used to set and read the 10-year digit and the 1-year digit of the year counter, respectively. The setting/read values are a BCD code within the range from 0 to 99.

Note: Be sure to avoid writing to the RTCYAR.RTCYH[3:0]/RTCYL[3:0] bits while the RTCCTL. RTCBSY bit = 1.

RTC Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCINTF	15	RTCTRMIF	0	H0	R/W	Cleared by writing 1.
	14	SW1IF	0	H0	R/W	
	13	SW10IF	0	H0	R/W	
	12	SW100IF	0	H0	R/W	
	11–9	–	0x0	–	R	–
	8	ALARMIF	0	H0	R/W	Cleared by writing 1.
	7	1DAYIF	0	H0	R/W	
	6	1HURIF	0	H0	R/W	
	5	1MINIF	0	H0	R/W	
	4	1SECIF	0	H0	R/W	
	3	1_2SECIF	0	H0	R/W	
	2	1_4SECIF	0	H0	R/W	
	1	1_8SECIF	0	H0	R/W	
	0	1_32SECIF	0	H0	R/W	

Bit 15 RTCTRMIF
Bit 14 SW1IF
Bit 13 SW10IF
Bit 12 SW100IF

These bits indicate the real-time clock 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.RTCTRMIF bit: Theoretical regulation completion interrupt
RTCINTF.SW1IF bit: Stopwatch 1 Hz interrupt
RTCINTF.SW10IF bit: Stopwatch 10 Hz interrupt
RTCINTF.SW100IF bit: Stopwatch 100 Hz interrupt

Bits 11–9 Reserved

Bit 8 ALARMIF
Bit 7 1DAYIF
Bit 6 1HURIF
Bit 5 1MINIF
Bit 4 1SECIF
Bit 3 1_2SECIF
Bit 2 1_4SECIF
Bit 1 1_8SECIF
Bit 0 1_32SECIF

These bits indicate the real-time clock 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. ALARMIF bit: Alarm interrupt
RTCINTF.1DAYIF bit: 1-day interrupt
RTCINTF.1HURIF bit: 1-hour interrupt
RTCINTF.1MINIF bit: 1-minute interrupt
RTCINTF.1SECIF bit: 1-second interrupt
RTCINTF.1_2SECIF bit: 1/2-second interrupt
RTCINTF.1_4SECIF bit: 1/4-second interrupt
RTCINTF.1_8SECIF bit: 1/8-second interrupt
RTCINTF.1_32SECIF bit: 1/32-second interrupt

RTC Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCINTE	15	RTCTRMIE	0	H0	R/W	–
	14	SW1IE	0	H0	R/W	
	13	SW10IE	0	H0	R/W	
	12	SW100IE	0	H0	R/W	
	11–9	–	0x0	–	R	
	8	ALARMIE	0	H0	R/W	
	7	1DAYIE	0	H0	R/W	
	6	1HURIE	0	H0	R/W	
	5	1MINIE	0	H0	R/W	
	4	1SECIE	0	H0	R/W	
	3	1_2SECIE	0	H0	R/W	
	2	1_4SECIE	0	H0	R/W	
	1	1_8SECIE	0	H0	R/W	
	0	1_32SECIE	0	H0	R/W	

Bit 15 RTCTRMIE
Bit 14 SW1IE
Bit 13 SW10IE
Bit 12 SW100IE

These bits enable real-time clock interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
RTCINTE.RTCTRMIE bit: Theoretical regulation completion interrupt
RTCINTE.SW1IE bit: Stopwatch 1 Hz interrupt
RTCINTE.SW10IE bit: Stopwatch 10 Hz interrupt
RTCINTE.SW100IE bit: Stopwatch 100 Hz interrupt

Bits 11–9 Reserved

Bit 8 ALARMIE
Bit 7 1DAYIE
Bit 6 1HURIE
Bit 5 1MINIE
Bit 4 1SECIE
Bit 3 1_2SECIE
Bit 2 1_4SECIE
Bit 1 1_8SECIE
Bit 0 1_32SECIE

These bits enable real-time clock interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
RTCINTE.ALARMIE bit: Alarm interrupt
RTCINTE.1DAYIE bit: 1-day interrupt
RTCINTE.1HURIE bit: 1-hour interrupt
RTCINTE.1MINIE bit: 1-minute interrupt
RTCINTE.1SECIE bit: 1-second interrupt
RTCINTE.1_2SECIE bit: 1/2-second interrupt
RTCINTE.1_4SECIE bit: 1/4-second interrupt
RTCINTE.1_8SECIE bit: 1/8-second interrupt
RTCINTE.1_32SECIE bit: 1/32-second interrupt

10 Supply Voltage Detector (SVD)

10.1 Overview

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

- Power supply voltage to be detected: External power supply (EXSVD)
- Detectable voltage level: Selectable from among 30 levels (1.2 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 10.1.1 shows the configuration of SVD.

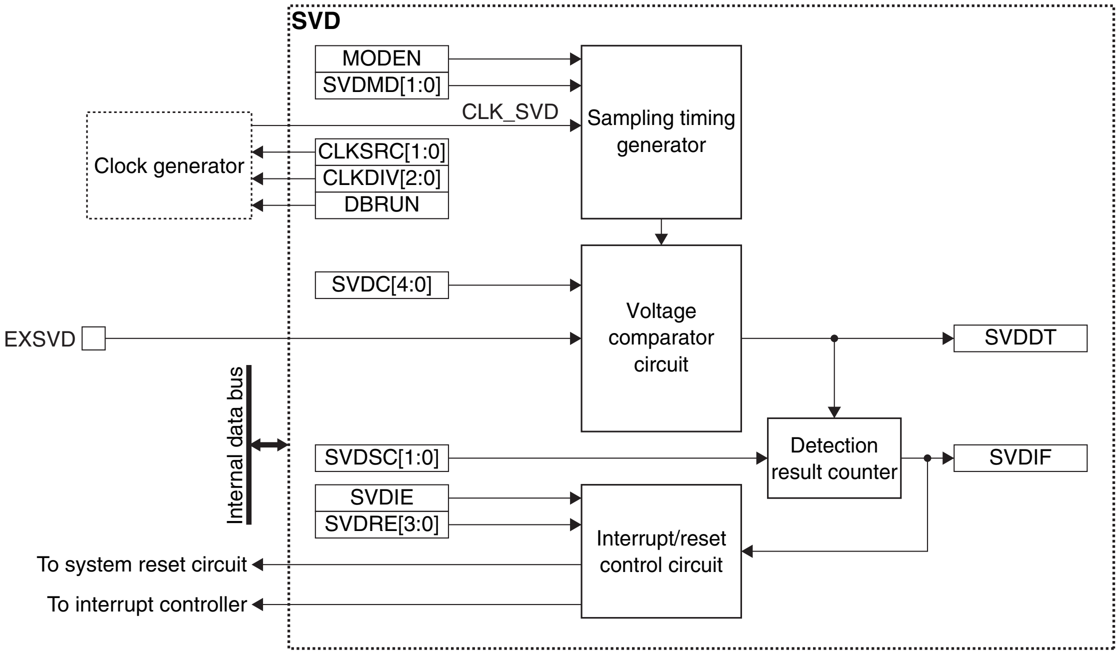


Figure 10.1.1 SVD Configuration

10.2 Input Pin and External Connection

10.2.1 Input Pin

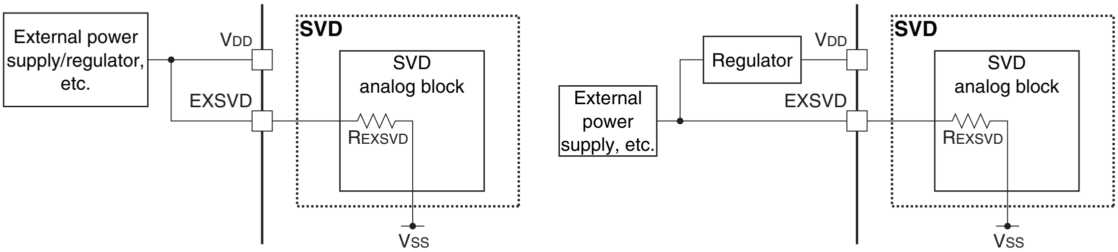
Table 10.2.1.1 shows the SVD input pin.

Table 10.2.1.1 SVD Input Pin

Pin name	I/O	Initial status	Function
EXSVD	A	I	External power supply voltage detection pin

Note: Do not leave the EXSVD pin open even if SVD is not used.

10.2.2 External Connection



(1) When detecting the voltage supplied to the V_{DD} pin (2) When detecting an external power supply voltage

Figure 10.2.2.1 Connection between EXSVD Pin and External Power Supply

For the EXSVD pin input voltage range and the EXSVD input impedance, refer to "Supply Voltage Detector Characteristics" in the "Electrical Characteristics" chapter.

10.3 Clock Settings

10.3.1 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.

10.3.2 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.

10.3.3 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.

10.4 Operations

10.4.1 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.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.SVDIE 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 (EXSVD) ≥ SVD detection voltage V_{SVD} when SVDINTF.SVDDT bit = 0
- Power supply voltage (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).

10.4.2 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.

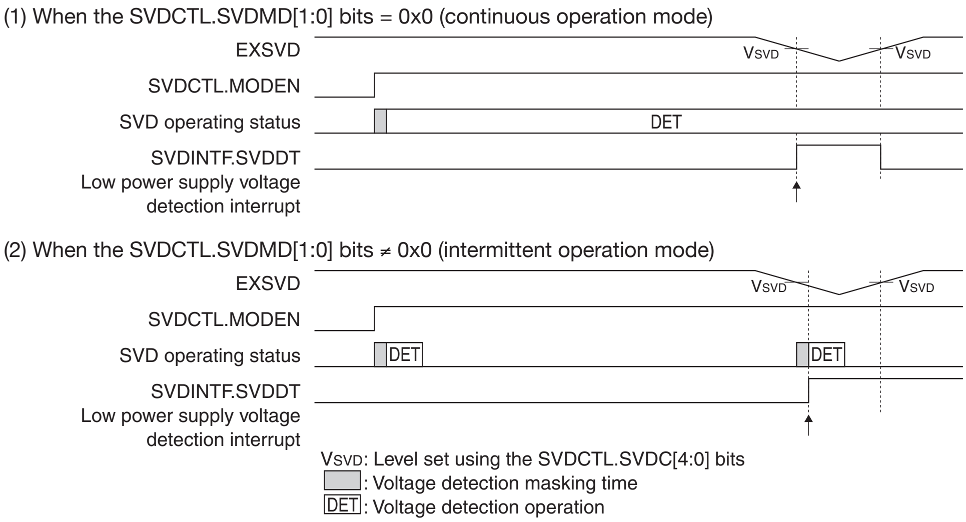


Figure 10.4.2.1 SVD Operations

10.5 SVD Interrupt and Reset

10.5.1 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 10.5.1.1 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 V_{SVD}. 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.

10.5.2 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.
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 10.5.2.1.

Table 10.5.2.1 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	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.

10.6 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 10.6.1 Clock Source and Division Ratio Settings

SVDCLK. CLKDIV[2:0] bits	SVDCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x6, 0x7	Reserved	1/1	Reserved	1/1
0x5	1/128		1/128	
0x4	1/64		1/64	
0x3	1/32		1/32	
0x2	1/16		1/16	
0x1	1/8		1/8	
0x0	1/4		1/4	

(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	–	0	–	R	–
	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]	0x1e	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 Reserved

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 10.6.2 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 from among 30 levels.

Table 10.6.3 Setting of SVD Detection Voltage V_{SVD}

SVDCTL.SVDC[4:0] bits	SVD detection voltage V_{SVD} [V]
0x1e	High
0x1d	↑
0x1c	
:	
0x02	↓
0x01	Low
0x00, 0x1f	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 10.6.4 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 (EXSVD) < SVD detection voltage V_{SVD}
0 (R): Power supply voltage (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.

11 16-bit Timers (T16)

11.1 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 11.1.1 shows the configuration of a T16 channel.

Table 11.1.1 T16 Channel Configuration of S1C17W15

Item	S1C17W15
Number of channels	3 channels (Ch.0–Ch.2)
Event counter function	Not supported (No EXCL*m* pins are provided.)
Peripheral clock output (Outputs the counter underflow signal.)	Ch.1 → Synchronous serial interface master clock

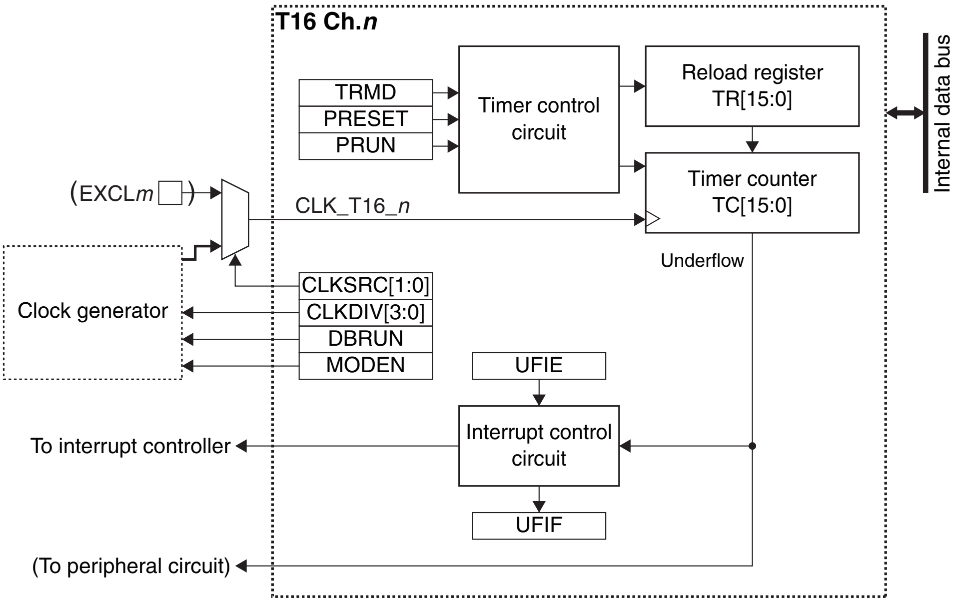


Figure 11.1.1 Configuration of a T16 Channel

11.2 Input Pin

Table 11.2.1 shows the T16 input pin.

Table 11.2.1 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.

11.3 Clock Settings

11.3.1 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)

11.3.2 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.

11.3.3 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.

11.3.4 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.

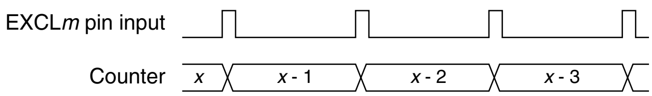


Figure 11.3.4.1 Count Down Timing

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

11.4 Operations

11.4.1 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)

11.4.2 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:

$$T = \frac{TR + 1}{f_{CLK\_T16\_n}} \qquad f_T = \frac{f_{CLK\_T16\_n}}{TR + 1} \qquad \text{(Eq. 11.1)}$$

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

11.4.3 Operations in Repeat Mode

T16 Ch.*n* enters repeat mode by setting the 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.

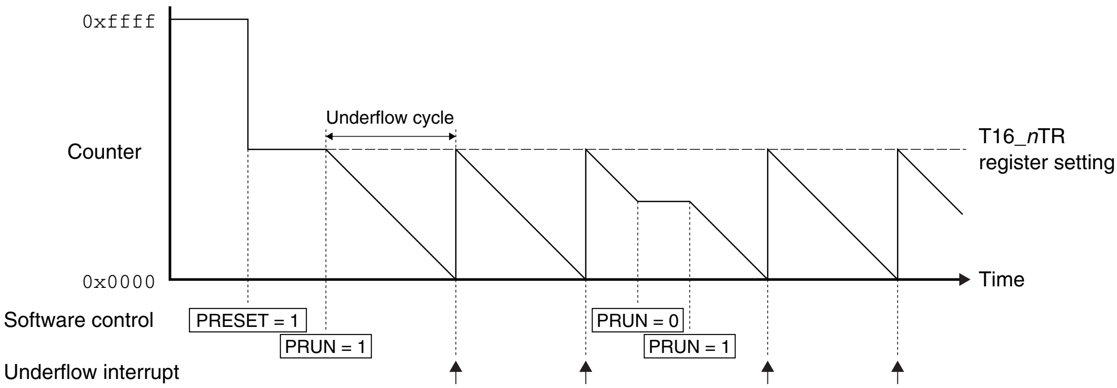


Figure 11.4.3.1 Count Operations in Repeat Mode

11.4.4 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.

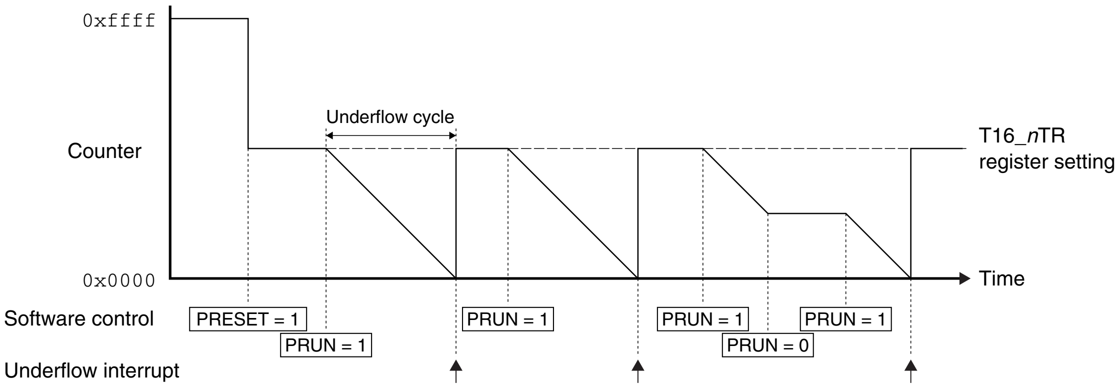


Figure 11.4.4.1 Count Operations in One-shot Mode

11.4.5 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.

11.5 Interrupt

Each T16 channel has a function to generate the interrupt shown in Table 11.5.1.

Table 11.5.1 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.

11.6 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 11.6.1 Clock Source and Division Ratio Settings

T16_*n*CLK. CLKDIV[3:0] bits	T16_*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	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.

12 UART (UART)

12.1 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 12.1.1 shows the UART configuration.

Table 12.1.1 UART Channel Configuration of S1C17W15

Item	S1C17W15
Number of channels	2 channels (Ch.0 and Ch.1)

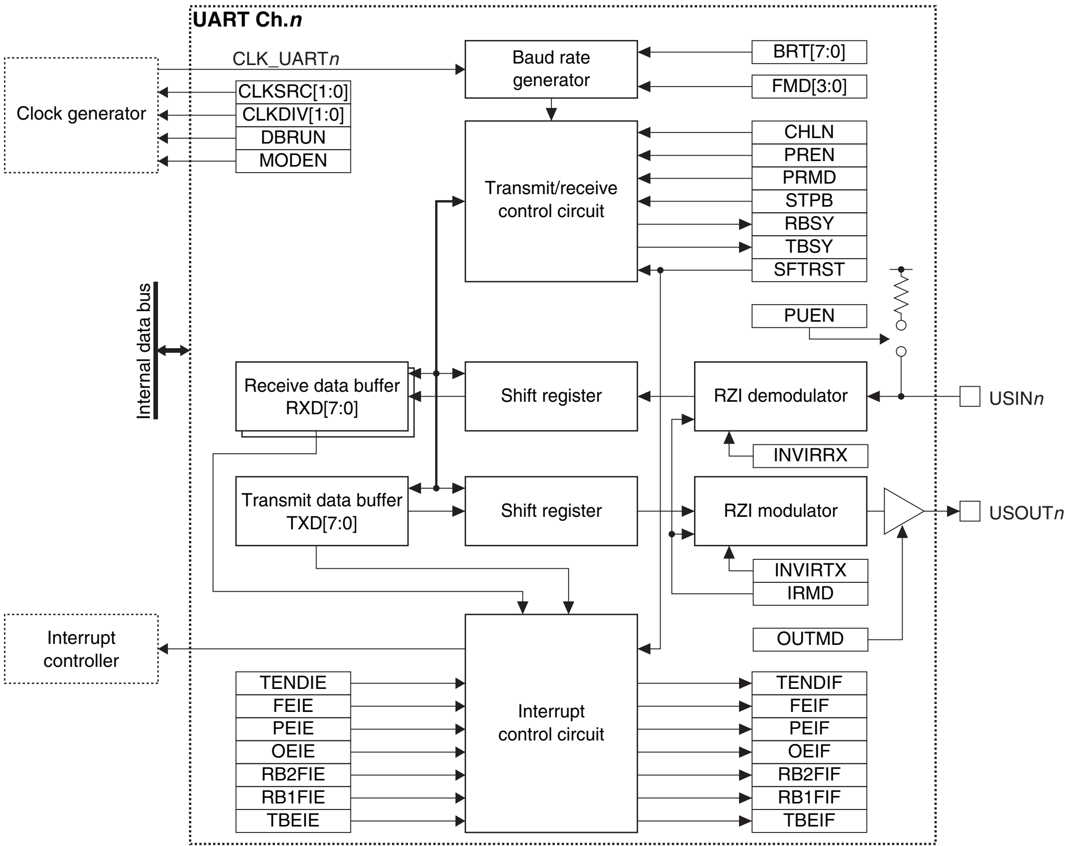


Figure 12.1.1 UART Configuration

12.2 Input/Output Pins and External Connections

12.2.1 List of Input/Output Pins

Table 12.2.1.1 lists the UART pins.

Table 12.2.1.1 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.

12.2.2 External Connections

Figure 12.2.2.1 shows a connection diagram between the UART in this IC and an external UART device.

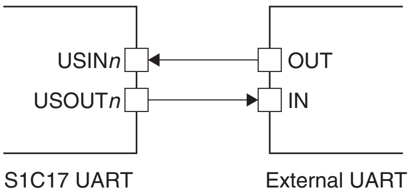


Figure 12.2.2.1 Connections between UART and an External UART Device

12.2.3 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.

12.2.4 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.

12.3 Clock Settings

12.3.1 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.

12.3.2 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.

12.3.3 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.

12.3.4 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.

$$bps = \frac{CLK\_UART}{\{(BRT + 1) \times 16 + FMD\}} \qquad BRT = \left(\frac{CLK\_UART}{bps} - FMD - 16\right) \div 16 \qquad \text{(Eq. 12.1)}$$

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.

12.4 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 12.4.1 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

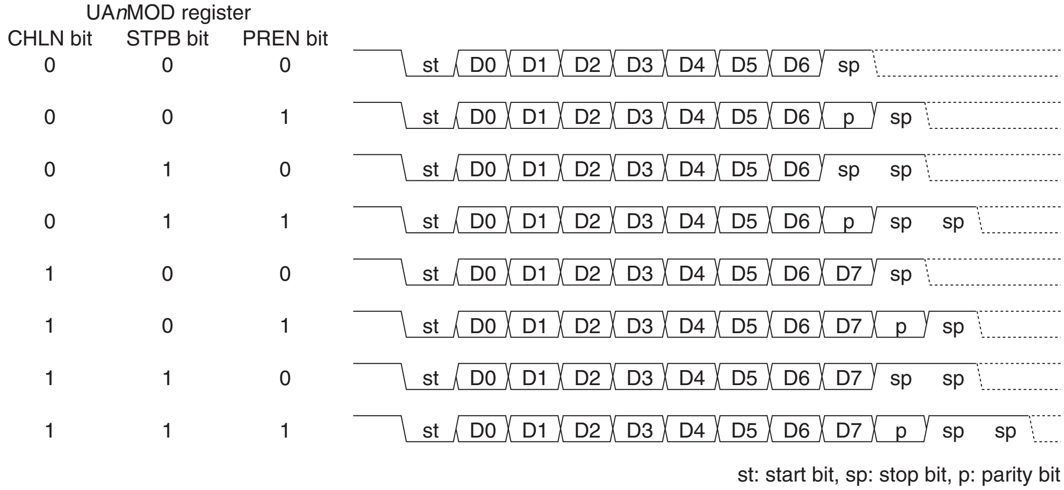


Figure 12.4.1 Data Format

12.5 Operations

12.5.1 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.

12.5.2 Data Transmission

A data sending procedure and the UART Ch.*n* operations are shown below. Figures 12.5.2.1 and 12.5.2.2 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).

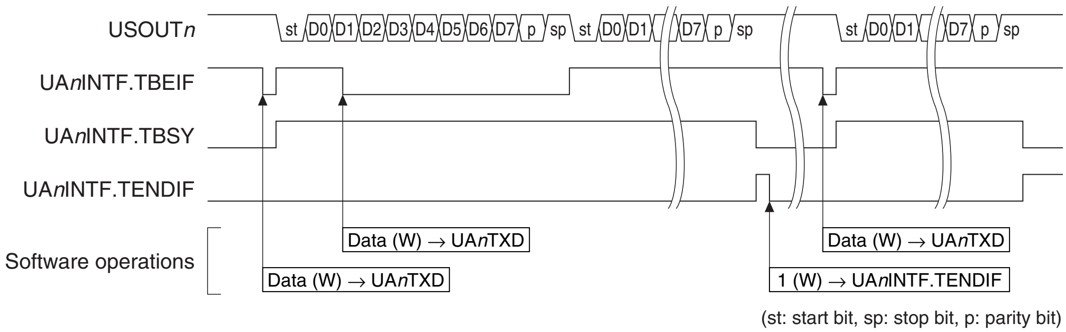


Figure 12.5.2.1 Example of Data Sending Operations

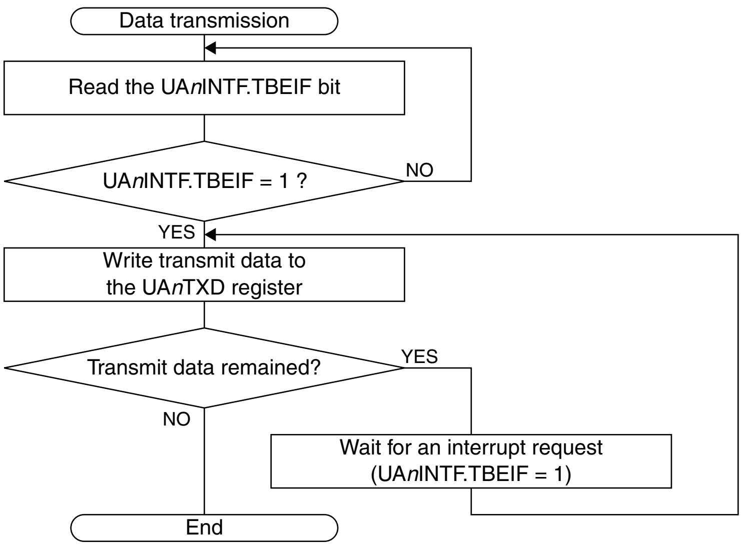


Figure 12.5.2.2 Data Transmission Flowchart

12.5.3 Data Reception

A data receiving procedure and the UART Ch.*n* operations are shown below. Figures 12.5.3.1 and 12.5.3.2 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).

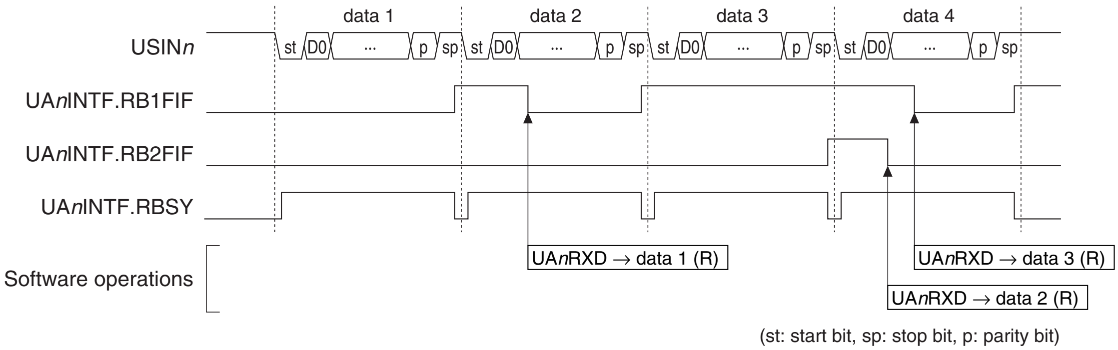


Figure 12.5.3.1 Example of Data Receiving Operations

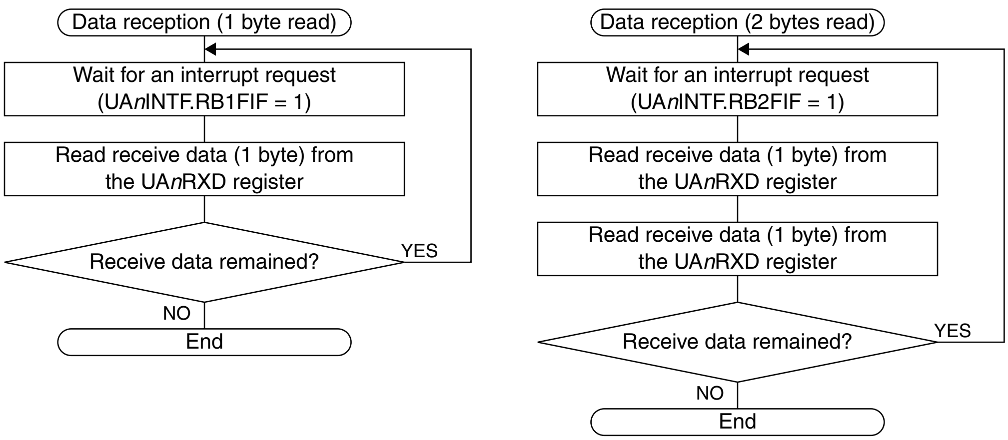


Figure 12.5.3.2 Data Reception Flowcharts

12.5.4 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.

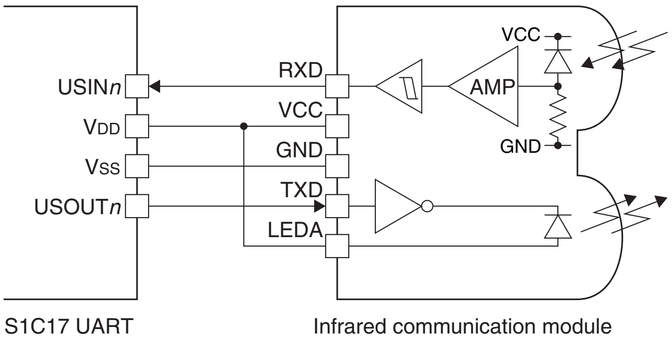


Figure 12.5.4.1 Example of Connections with an Infrared Communication Module

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.

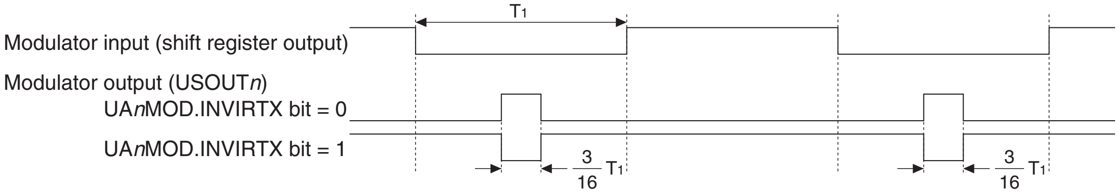


Figure 12.5.4.2 IrDA Transmission Signal Waveform

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.

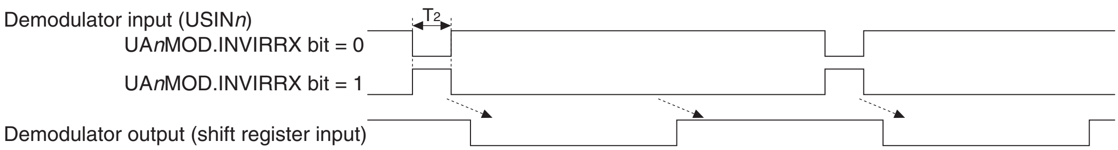


Figure 12.5.4.3 IrDA Receive Signal Waveform

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

12.6 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.

12.6.1 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.

12.6.2 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).

12.6.3 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.

12.7 Interrupts

The UART has a function to generate the interrupts shown in Table 12.7.1.

Table 12.7.1 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 the “Receive Errors.”	Writing 1, reading received data that encountered an error, or software reset
Parity error	UA*n*INTF.PEIF	Refer to the “Receive Errors.”	Writing 1, reading received data that encountered an error, or software reset
Overrun error	UA*n*INTF.OEIF	Refer to the “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.

12.8 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 12.8.1 Clock Source and Division Ratio Settings

UA*n*CLK. CLKDIV[1:0] bits	UA*n*CLK.CLKSRC[1:0] bits			
	0x0 IOSC	0x1 OSC1	0x2 OSC3	0x3 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 12.5.3.1.)

1 (R): During receiving
0 (R): Idle

Bit 8 TBSY

This bit indicates the sending status. (See Figure 12.5.2.1.)

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

13 Synchronous Serial Interface (SPIA)

13.1 Overview

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

- Supports both master and slave modes.
- Data length: 2 to 16 bits programmable
- 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, end of transmission, and overrun 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 SPIA interrupt.
- Input pins can be pulled up/down with an internal resistor.

Figure 13.1.1 shows the SPIA configuration.

Table 13.1.1 SPIA Channel Configuration of S1C17W15

Item	S1C17W15
Number of channels	1 channel (Ch.0)
Internal clock input	Ch.0 ← 16-bit timer Ch.1

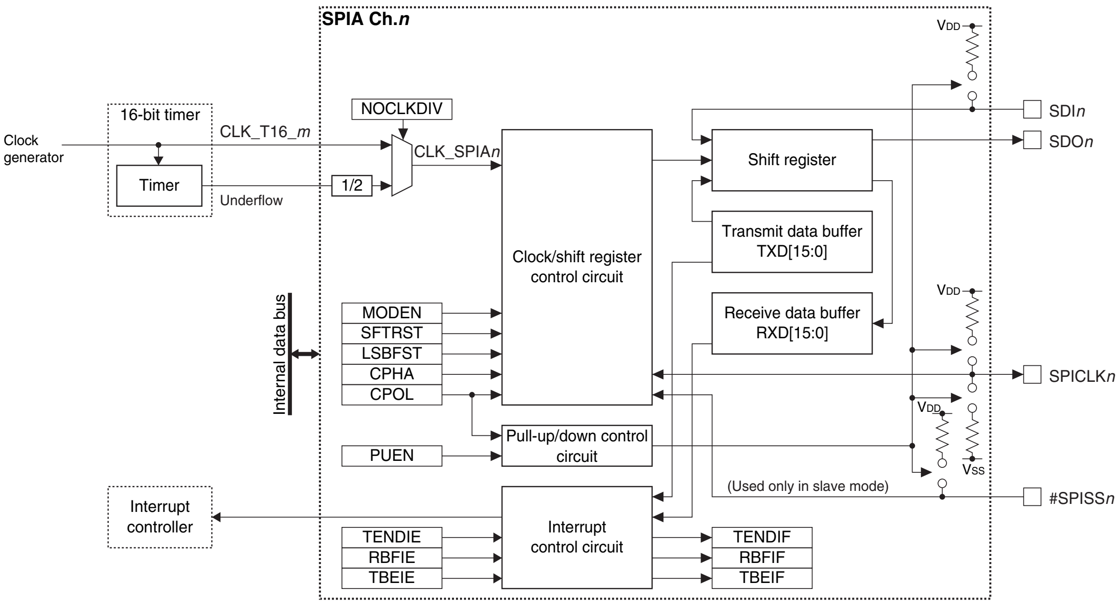


Figure 13.1.1 SPIA Configuration

13.2 Input/Output Pins and External Connections

13.2.1 List of Input/Output Pins

Table 13.2.1.1 lists the SPIA pins.

Table 13.2.1.1 List of SPIA Pins

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

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

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

13.2.2 External Connections

SPIA operates in master mode or slave mode. Figures 13.2.2.1 and 13.2.2.2 show connection diagrams between SPIA in each mode and external SPI devices.

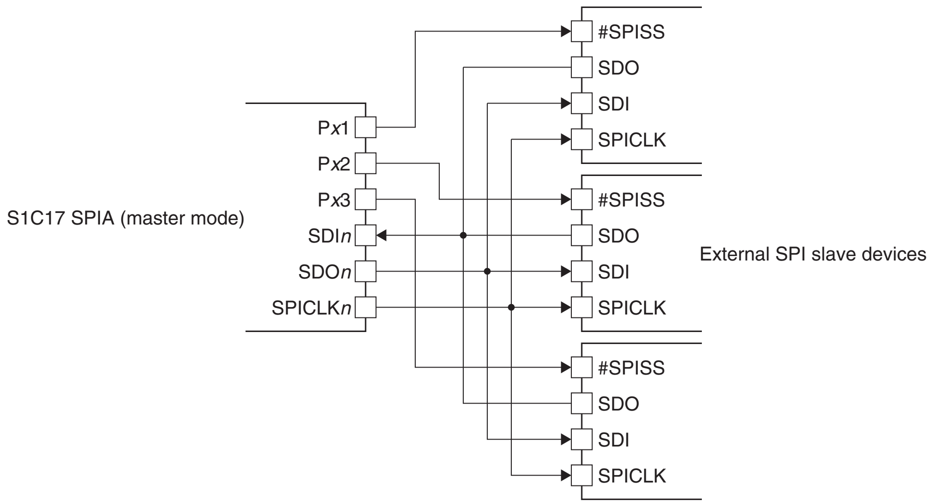


Figure 13.2.2.1 Connections between SPIA in Master Mode and External SPI Slave Devices

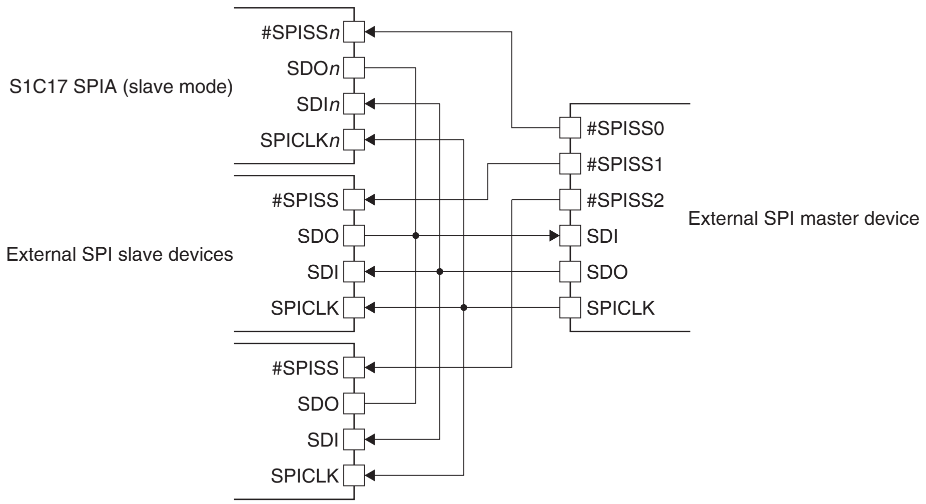


Figure 13.2.2.2 Connections between SPIA in Slave Mode and External SPI Master Device

13.2.3 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 13.2.3.1.

Table 13.2.3.1 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 SPIA to transmit/receive data. While a high level is applied to this pin, SPIA 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.

13.2.4 Input Pin Pull-Up/Pull-Down Function

The SPIA 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 13.2.4.1. This function is enabled by setting the SPI*n*MOD.PUEN bit to 1.

Table 13.2.4.1 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

13.3 Clock Settings

13.3.1 SPIA Operating Clock

Operating clock in master mode

In master mode, the SPIA 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 SPIA channel is input to SPIA as CLK_SPIA*n*. Since this clock is also used as the SPI clock SPICLK*n* without changing, the CLK_SPIA*n* frequency becomes the baud rate.

To supply CLK_SPIA*n* to SPIA, the 16-bit timer clock source must be enabled in the clock generator. It does not matter how the T16_*m*CTL.MODEN and T16_*m*CTL.PRUN bits of the corresponding 16-bit timer channel are 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, SPIA inputs 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.

$$f_{SPICLK} = \frac{f_{CLK\_SPIA}}{2 \times (RLD + 1)} \qquad RLD = \frac{f_{CLK\_SPIA}}{f_{SPICLK} \times 2} - 1 \qquad \text{(Eq. 13.1)}$$

Where
- f_{SPICLK}: SPICLK*n* frequency [Hz] (= baud rate [bps])
- f_{CLK_SPIA}: SPIA 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

SPIA 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 SPIA channel is not used. Furthermore, the SPI*n*MOD.NOCLKDIV bit setting becomes ineffective.

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

13.3.2 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 SPIA 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 SPIA 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 SPIA Ch.*n* will keep operating in DEBUG mode.

SPIA 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.

13.3.3 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 13.3.3.1 shows the clock waveform and data input/output timing in each setting.

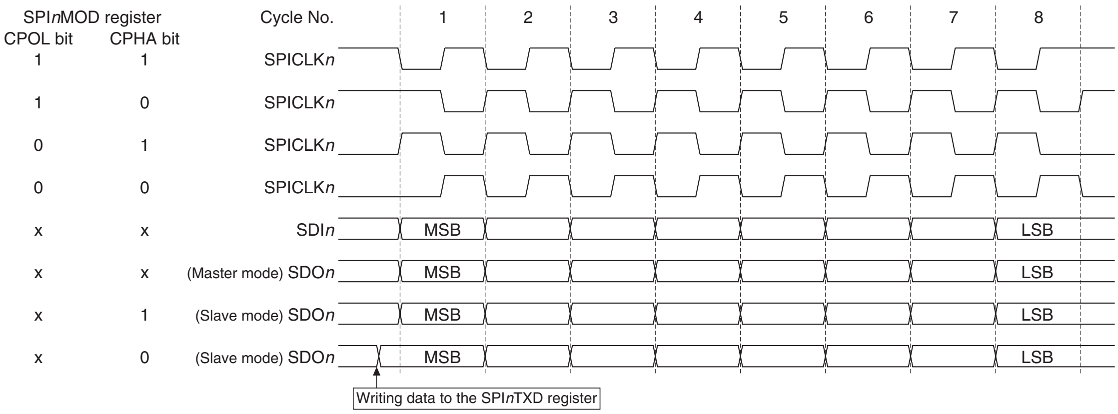


Figure 13.3.3.1 SPI Clock Phase and Polarity (SPI*n*MOD.LSBFST bit = 0, SPI*n*MOD.CHLN[3:0] bits = 0x7)

13.4 Data Format

The SPIA data length can be selected from 2 bits to 16 bits by setting the SPI*n*MOD.CHLN[3:0] bits. The input/output permutation is configurable to MSB first or LSB first using the SPI*n*MOD.LSBFST bit. Figure 13.4.1 shows a data format example when the SPI*n*MOD.CHLN[3:0] bits = 0x7, the SPI*n*MOD.CPOL bit = 0 and the SPI*n*MOD.CPHA bit = 0.

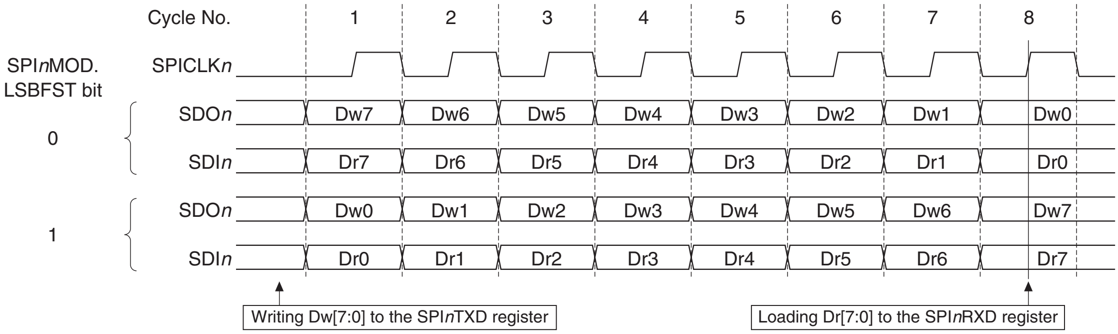


Figure 13.4.1 Data Format Selection Using the SPI*n*MOD.LSBFST Bit
(SPI*n*MOD.CHLN[3:0] bits = 0x7, SPI*n*MOD.CPOL bit = 0, SPI*n*MOD.CPHA bit = 0)

13.5 Operations

13.5.1 Initialization

SPIA 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 SPIA 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 SPIA 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 SPIA 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.

13.5.2 Data Transmission in Master Mode

A data sending procedure and operations in master mode are shown below. Figures 13.5.2.1 and 13.5.2.2 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 SPIA 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

SPIA 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 clocks of the number of the bits specified by the SPI*n*MOD.CHLN[3:0] bits and the transmit data bits are output in sequence from the SDO*n* pin in sync with these clocks.

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 last 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 SPIA issues an end-of-transmission interrupt request if the SPI*n*INTE.TENDIE bit = 1.

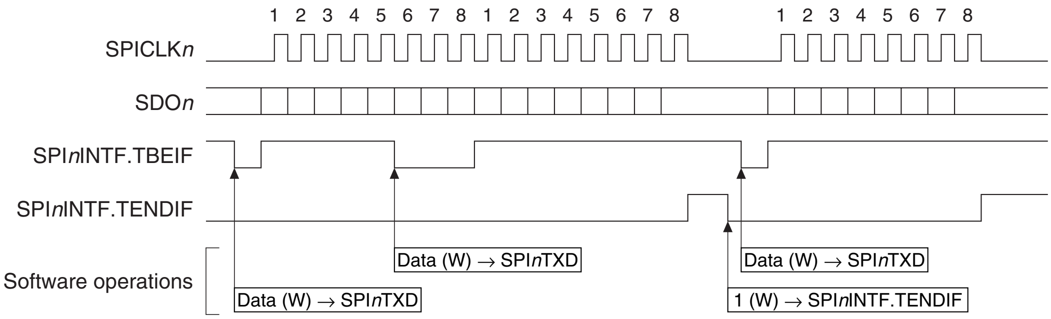


Figure 13.5.2.1 Example of Data Sending Operations in Master Mode (SPI*n*MOD.CHLN[3:0] bits = 0x7)

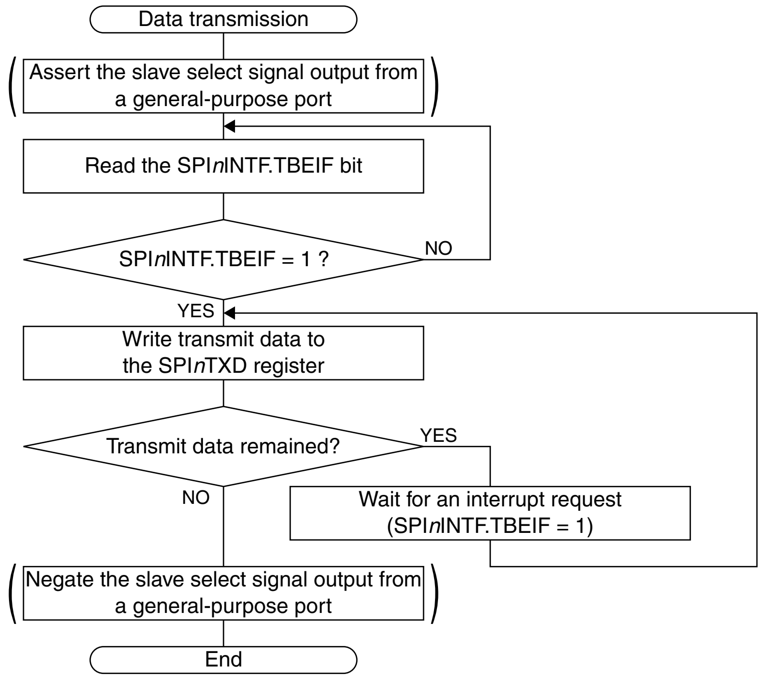


Figure 13.5.2.2 Data Transmission Flowchart in Master Mode

13.5.3 Data Reception in Master Mode

A data receiving procedure and operations in master mode are shown below. Figures 13.5.3.1 and 13.5.3.2 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 the SPICLK*n* cycles equivalent to "Data bit length - 1" after Step 6.

Data receiving operations

SPIA 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 clocks of the number of the bits specified by the SPI*n*MOD.CHLN[3:0] bits. The transmit data bits are output in sequence from the SDO*n* pin in sync with these clocks and the receive data bits input from the SDI*n* pin are shifted into the shift register.

When the last clock is output from the SPICLK*n* pin and receive data bits are all 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 SPIA 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 data of the number of the bits specified by the SPI*n*MOD.CHLN[3:0] bits is received when the SPI*n*INTF.RBFIF bit is set to 1, the SPI*n*RXD register is overwritten with the newly received data and the previously received data is lost. In this case, the SPI*n*INTF.OEIF bit is set.

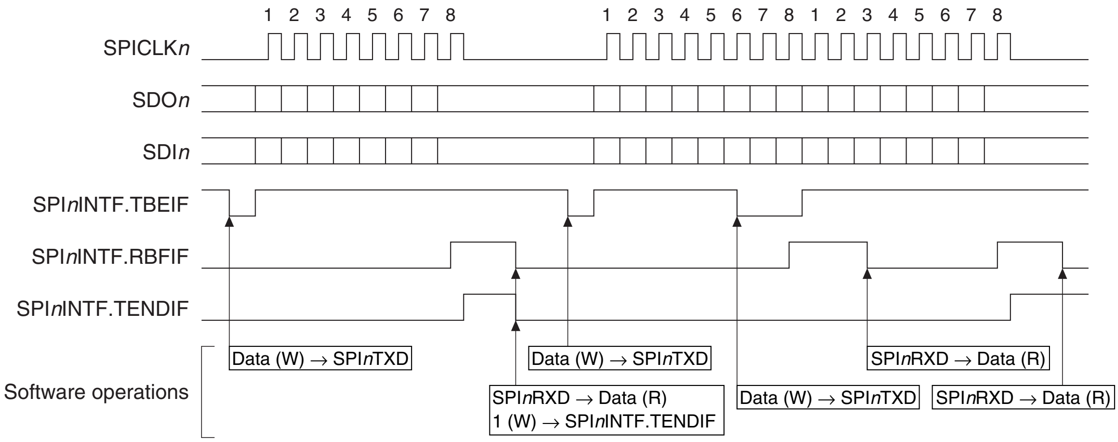
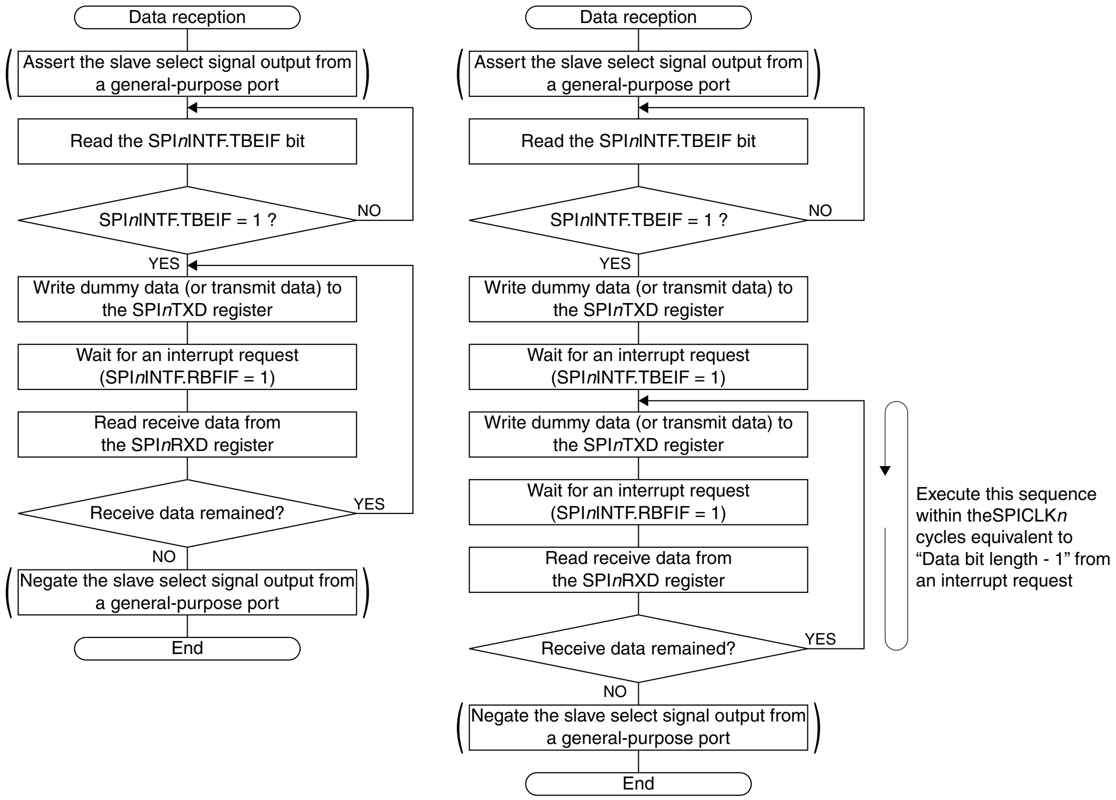


Figure 13.5.3.1 Example of Data Receiving Operations in Master Mode (SPI*n*MOD.CHLN[3:0] bits = 0x7)



(A) Intermittent data reception (B) Continuous data reception

Figure 13.5.3.2 Data Reception Flowcharts in Master Mode

13.5.4 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 SPIA Ch.*n* operations.
3. Stop the 16-bit timer to disable the clock supply to SPIA Ch.*n*.

13.5.5 Data Transfer in Slave Mode

A data sending/receiving procedure and operations in slave mode are shown below. Figures 13.5.5.1 and 13.5.5.2 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 by the time the sending SPI*n*TXD register data written is completed. 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.
- SPIA 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 SPIA interrupt.

Other operations are the same as master mode.

Notes: • If data of the number of bits specified by the SPI*n*MOD.CHLN[3:0] bits is received when the SPI*n*INTF.RBFIF bit is set to 1, the SPI*n*RXD register is overwritten with the newly received data and the previously received data is lost. In this case, the SPI*n*INTF.OEIF bit is set.

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

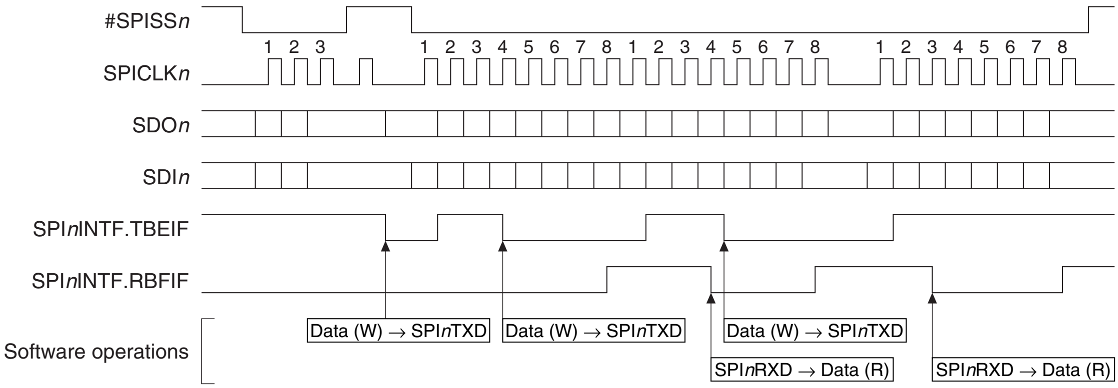


Figure 13.5.5.1 Example of Data Transfer Operations in Slave Mode (SPI*n*MOD.CHLN[3:0] bits = 0x7)

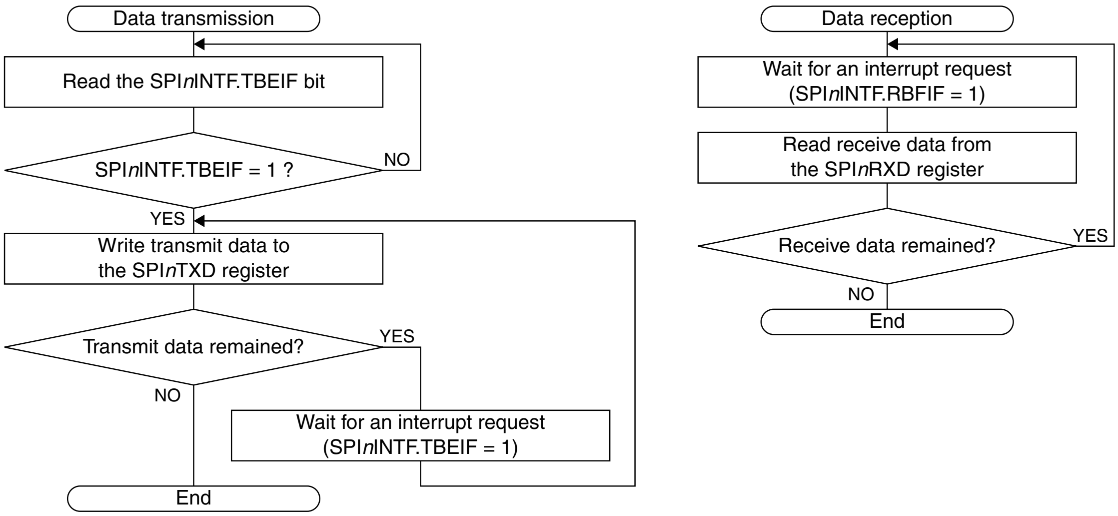


Figure 13.5.5.2 Data Transfer Flowcharts in Slave Mode

13.5.6 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 SPIA Ch.*n* operations.

13.6 Interrupts

SPIA has a function to generate the interrupts shown in Table 13.6.1.

Table 13.6.1 SPIA 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 data of the specified bit length (defined by the SPI*n*MOD. CHLN[3:0] bits) has been sent	Writing 1
Receive buffer full	SPI*n*INTF.RBFIF	When data of the specified bit length 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
Overrun error	SPI*n*INTF.OEIF	When the receive data buffer is full (when the received data has not been read) at the point that receiving data to the shift register has completed	Writing 1

SPIA 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 SPIA operating status.
Figure 13.6.1 shows the SPI*n*INTF.BSY and SPI*n*INTF.TENDIF bit set timings.

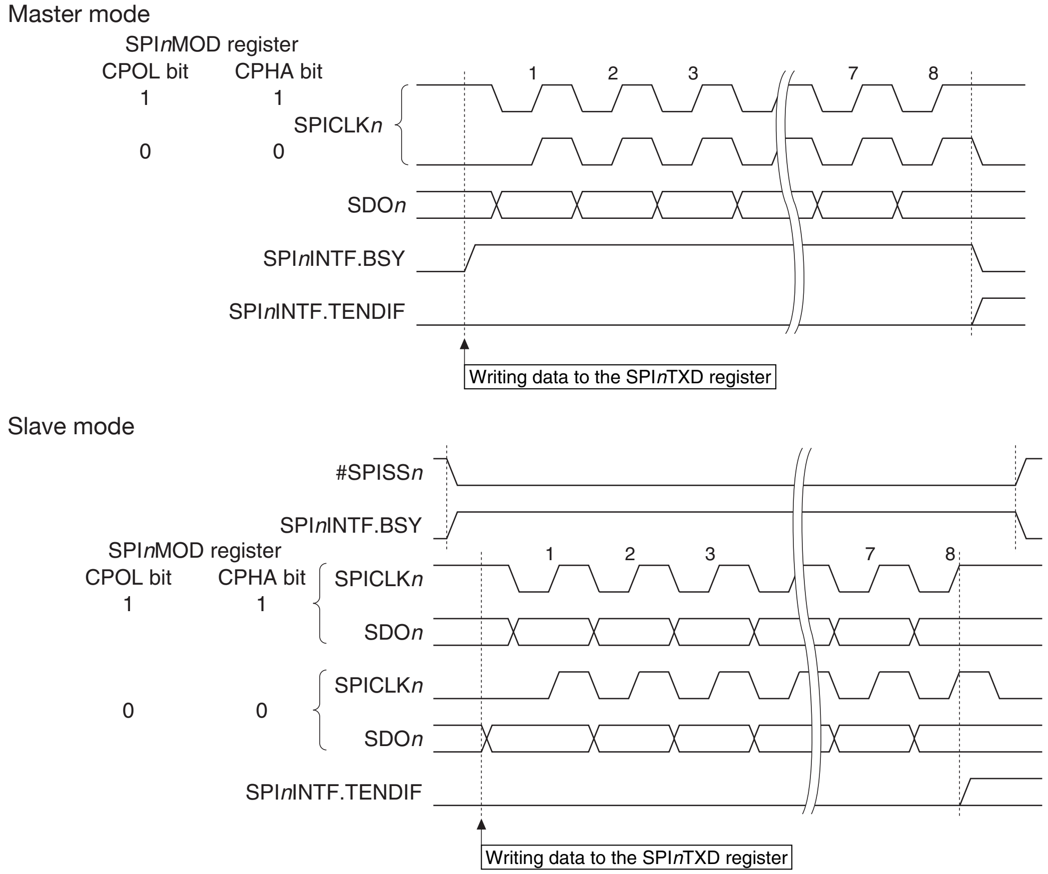


Figure 13.6.1 SPI*n*INTF.BSY and SPI*n*INTF.TENDIF Bit Set Timings (when SPI*n*MOD.CHLN[3:0] bits = 0x7)

13.7 Control Registers

SPIA Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*MOD	15–12	–	0x0	–	R	–
	11–8	CHLN[3:0]	0x7	H0	R/W	
	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 15–12 Reserved

Bits 11–8 CHLN[3:0]

These bits set the bit length of transfer data.

Table 13.7.1 Data Bit Length Settings

SPI*n*MOD.CHLN[3:0] bits	Data bit length
0xf	16 bits
0xe	15 bits
0xd	14 bits
0xc	13 bits
0xb	12 bits
0xa	11 bits
0x9	10 bits
0x8	9 bits
0x7	8 bits
0x6	7 bits
0x5	6 bits
0x4	5 bits
0x3	4 bits
0x2	3 bits
0x1	2 bits
0x0	Setting prohibited

Bits 7–6 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_SPIA*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 "SPIA 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 SPIA 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.

SPIA 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 SPIA.

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 SPIA shift register and transfer bit counter. This bit is automatically cleared after the reset processing has finished.

Bit 0 MODEN

This bit enables the SPIA operations.

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

SPIA Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*TXD	15–0	TXD[15:0]	0x0000	H0	R/W	–

Bits 15–0 TXD[15:0]

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 that the upper data bits that exceed the data bit length configured by the SPI*n*MOD.CHLN[3:0] bits will not be output from the SDO*n* pin.

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.

SPIA Ch.*n* Receive Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*RXD	15–0	RXD[15:0]	0x0000	H0	R	–

Bits 15–0 RXD[15:0]

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 that the upper bits that exceed the data bit length configured by the SPI*n*MOD.CHLN[3:0] bits become 0.

Note: The SPI*n*RXD.RXD[15:0] bits are cleared to 0x0000 when 1 is written to the SPI*n*CTL.MODEN bit or the SPI*n*CTL.SFTRST bit.

SPIA Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*INTF	15–8	–	0x00	–	R	–
	7	BSY	0	H0	R	
	6–4	–	0x0	–	R	
	3	OEIF	0	H0/S0	R/W	Cleared by writing 1.
	2	TENDIF	0	H0/S0	R/W	
	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 15–8 Reserved

Bit 7 BSY

This bit indicates the SPIA operating status.

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

0 (R): Idle

Bits 6–4 Reserved

Bit 3 OEIF

Bit 2 TENDIF

Bit 1 RBFIF

Bit 0 TBEIF

These bits indicate the SPIA interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag (OEIF, TENDIF)

0 (W): Ineffective

The following shows the correspondence between the bit and interrupt:

SPI*n*INTF.OEIF bit: Overrun error 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

SPIA Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPI*n*INTE	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3	OEIE	0	H0	R/W	
	2	TENDIE	0	H0	R/W	
	1	RBFIE	0	H0	R/W	
	0	TBEIE	0	H0	R/W	

Bits 15–4 Reserved

Bit 3 OEIE

Bit 2 TENDIE

Bit 1 RBFIE

Bit 0 TBEIE

These bits enable SPIA interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:

SPI*n*INTE.OEIE bit: Overrun error 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

14 I^{2}C (I2C)

14.1 Overview

The I2C is a subset of the I^{2}C bus interface. The features of the I2C are listed below.

- Functions as an I^{2}C 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 I^{2}C 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.
- The input filter for the SDA and SCL inputs does not comply with the standard for removing noise spikes less than 50 ns.

Figure 14.1.1 shows the I2C configuration.

Table 14.1.1 I2C Channel Configuration of S1C17W15

Item	S1C17W15
Number of channels	1 channel (Ch.0)

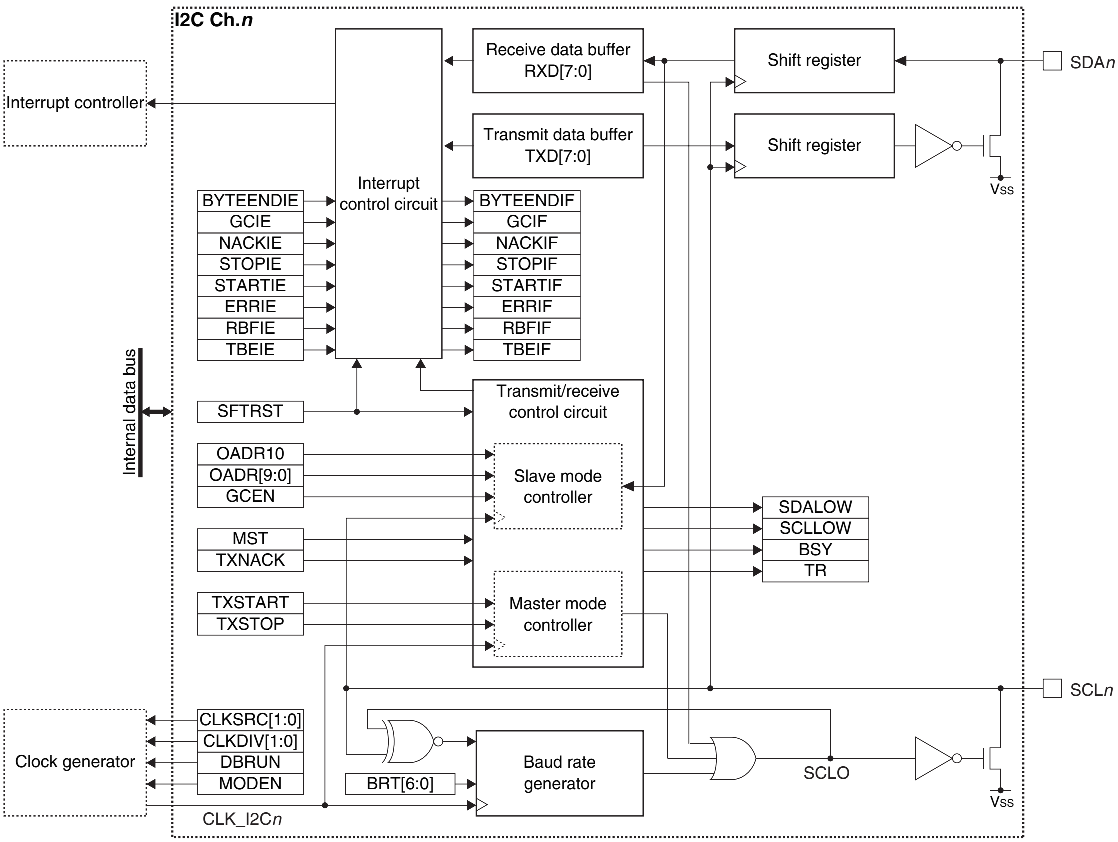


Figure 14.1.1 I2C Configuration

14.2 Input/Output Pins and External Connections

14.2.1 List of Input/Output Pins

Table 14.2.1.1 lists the I2C pins.

Table 14.2.1.1 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.

14.2.2 External Connections

Figure 14.2.2.1 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.

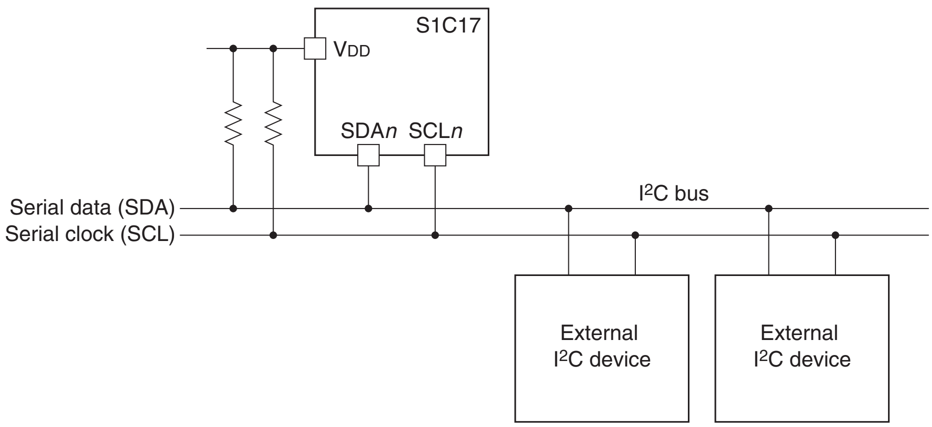


Figure 14.2.2.1 Connections between I2C and External I²C Devices

Notes:
- The SDA and SCL lines must be pulled up to a V_{DD} 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 V_{DD} 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.

14.3 Clock Settings

14.3.1 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.

14.3.2 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.

14.3.3 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.

$$bps = \frac{f_{CLK\_I2Cn}}{(BRT + 3) \times 2} \qquad BRT = \frac{f_{CLK\_I2Cn}}{bps \times 2} - 3 \qquad \text{(Eq. 14.1)}$$

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 14.3.3.1).

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

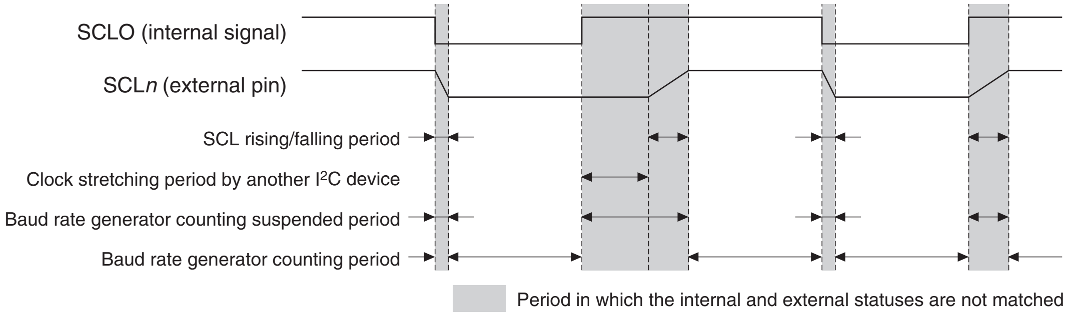


Figure 14.3.3.1 Baud Rate Generator Output Clock and SCL*n* Output Waveform

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.

14.4 Operations

14.4.1 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)

14.4.2 Data Transmission in Master Mode

A data sending procedure in master mode and the I2C Ch.*n* operations are shown below. Figures 14.4.2.1 and 14.4.2.2 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.

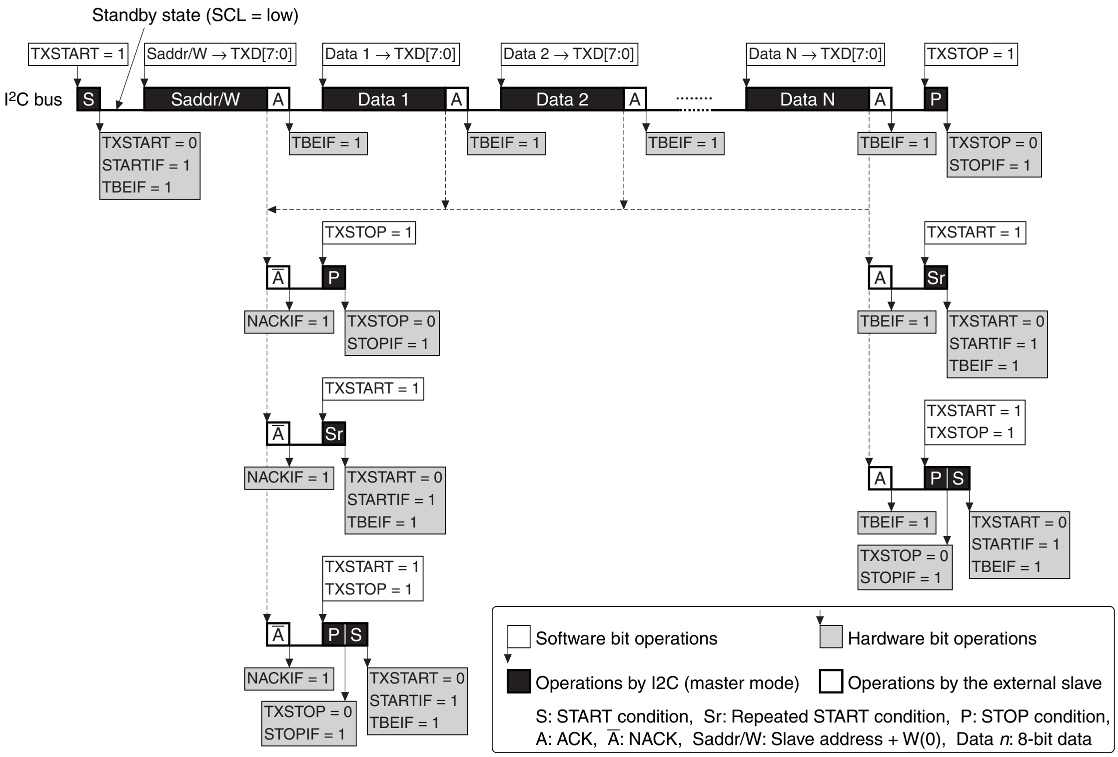


Figure 14.4.2.1 Example of Data Sending Operations in Master Mode

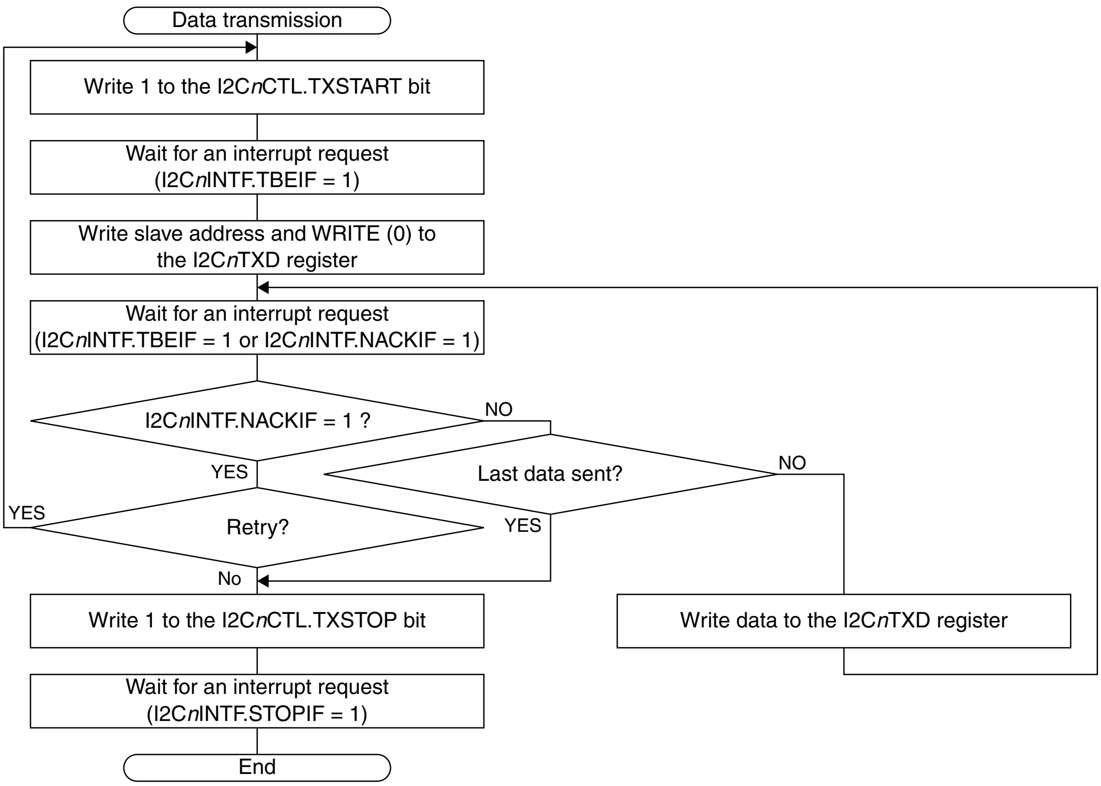


Figure 14.4.2.2 Master Mode Data Transmission Flowchart

14.4.3 Data Reception in Master Mode

A data receiving procedure in master mode and the I2C Ch.*n* operations are shown below. Figures 14.4.3.1 and 14.4.3.2 show an operation example and a flowchart, respectively.

Data receiving procedure

1. When receiving one-byte data, write 1 to the I2C*n*CTL.TXNACK bit.
2. Issue a START condition by setting the I2C*n*CTL.TXSTART bit to 1.
3. 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.
4. 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.
5. 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 6 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 9 or Step 2 if making a retry.
6. 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 7.
   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 9.
7. Read the received data from the I2C*n*RXD register.
8. Repeat Steps 5 to 7 until the end of data reception.
9. 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.

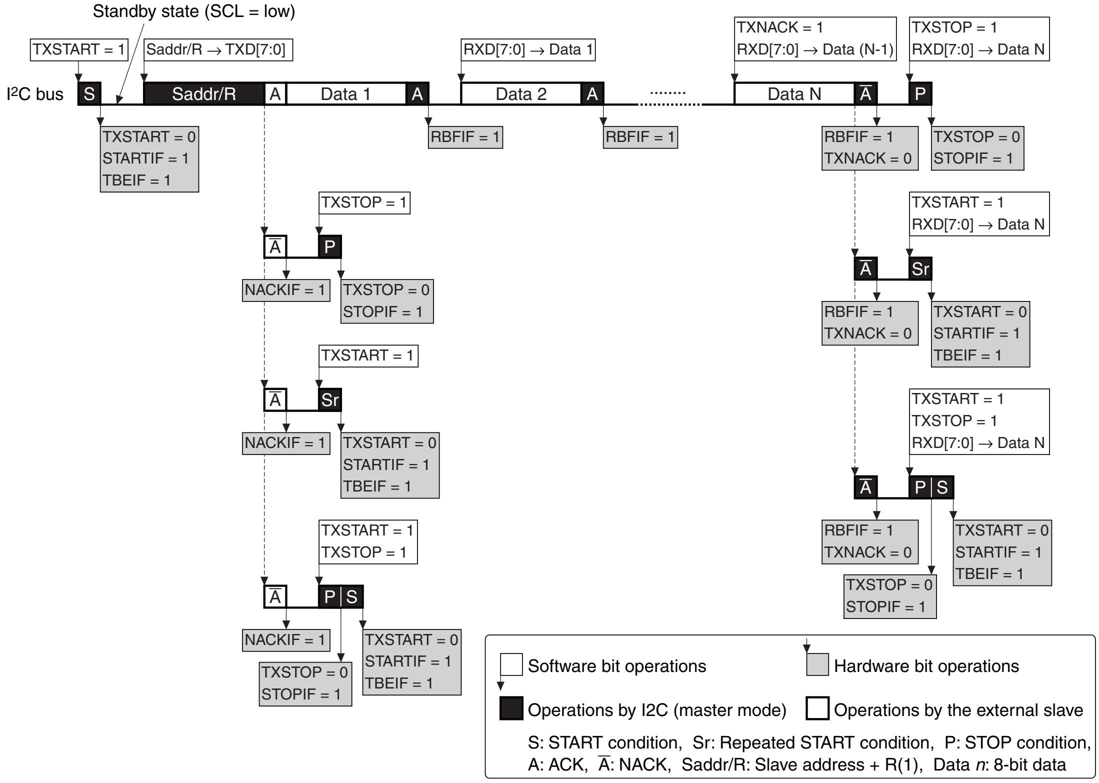


Figure 14.4.3.1 Example of Data Receiving Operations in Master Mode

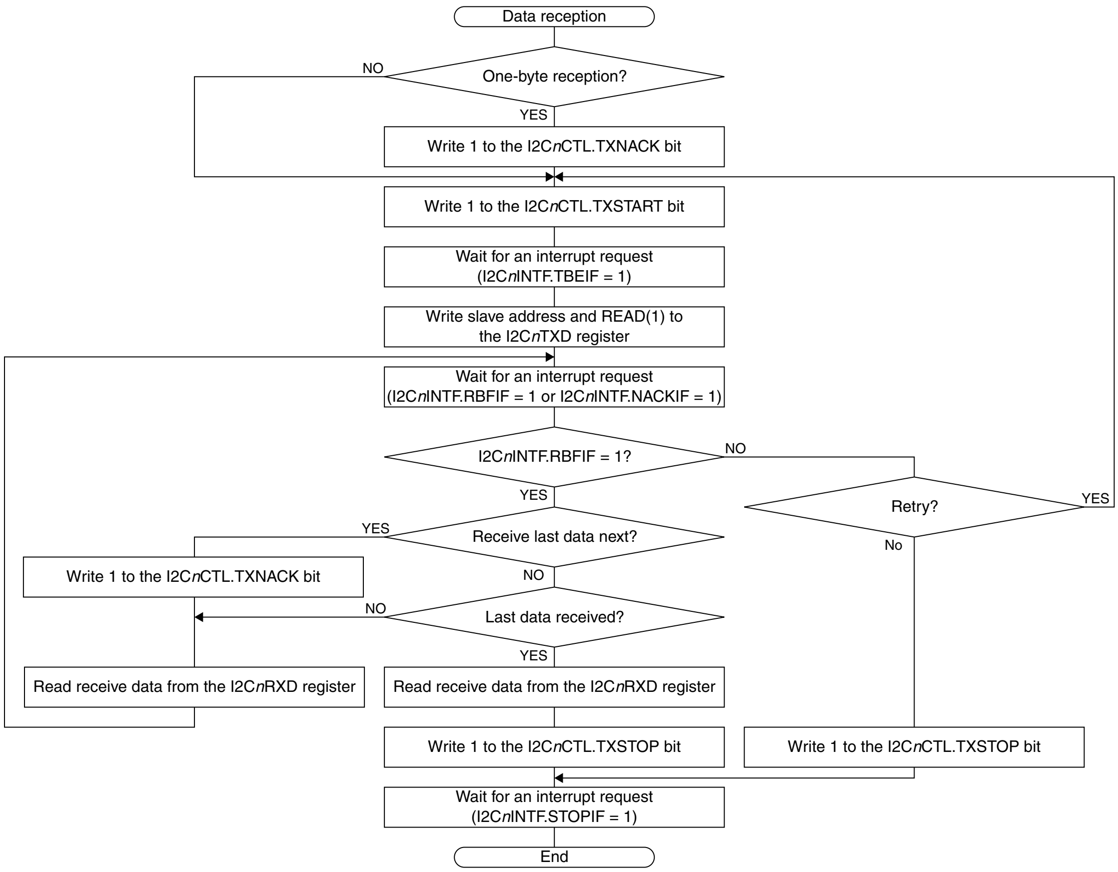


Figure 14.4.3.2 Master Mode Data Reception Flowchart

14.4.4 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.

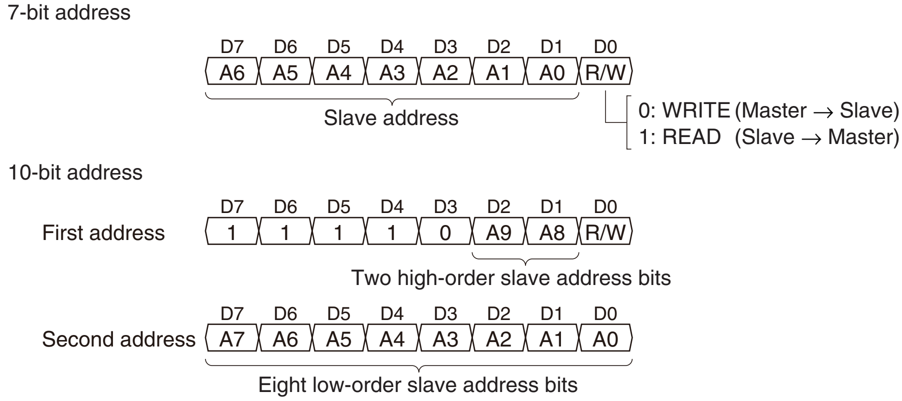


Figure 14.4.4.1 10-bit Address Configuration

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 14.4.4.2 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.

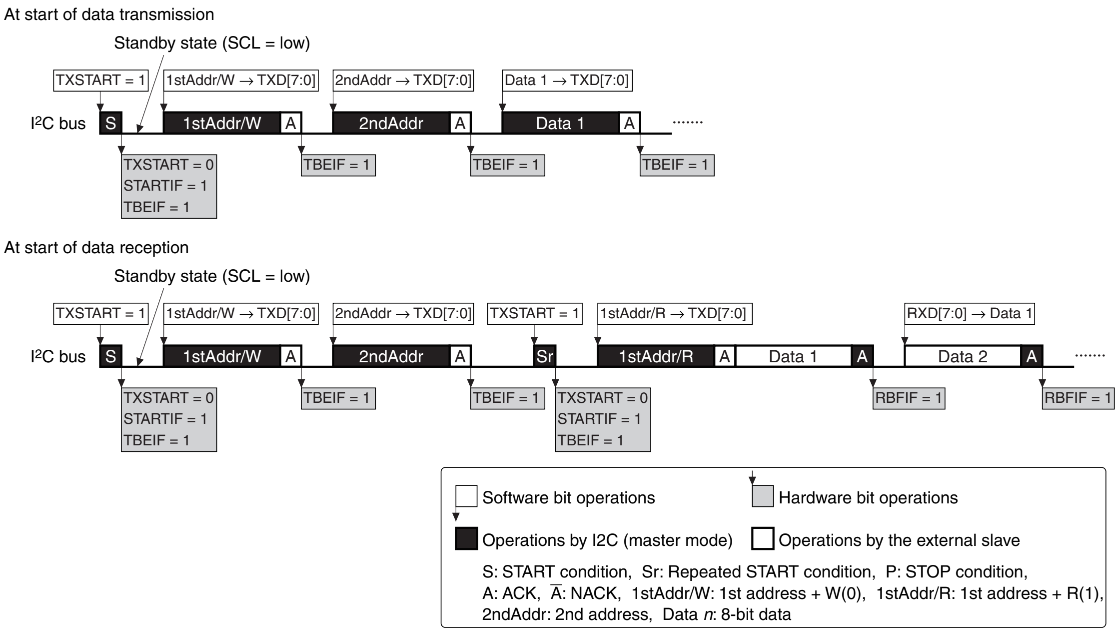


Figure 14.4.4.2 Example of Data Transfer Starting Operations in 10-bit Address Mode (Master Mode)

14.4.5 Data Transmission in Slave Mode

A data sending procedure in slave mode and the I2C Ch.*n* operations are shown below. Figures 14.4.5.1 and 14.4.5.2 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.

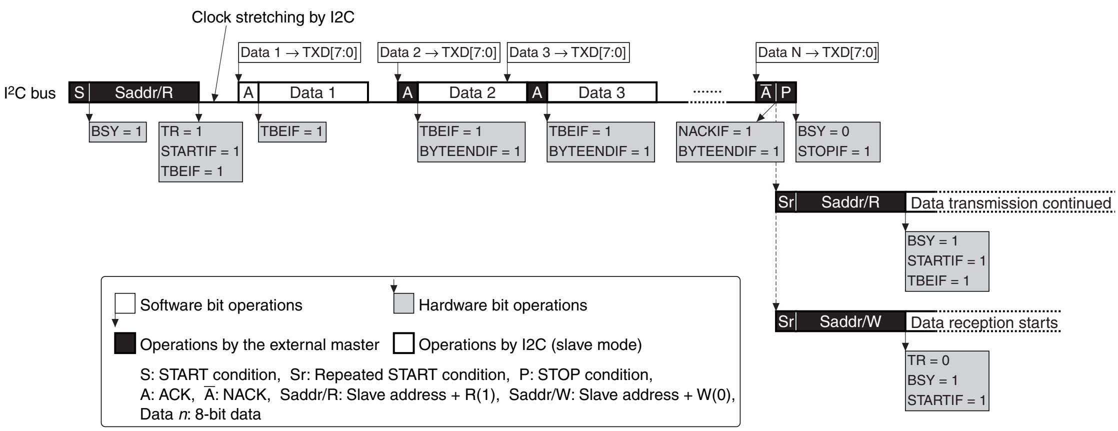


Figure 14.4.5.1 Example of Data Sending Operations in Slave Mode

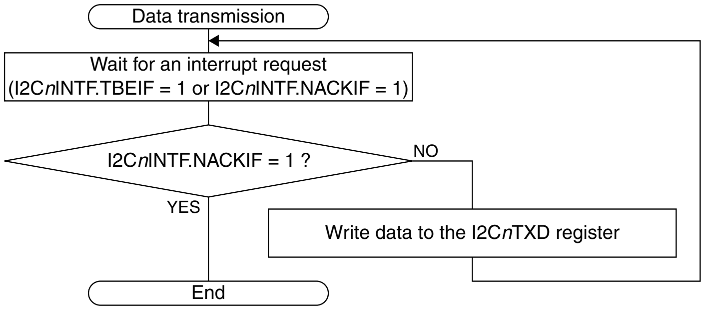


Figure 14.4.5.2 Slave Mode Data Transmission Flowchart

14.4.6 Data Reception in Slave Mode

A data receiving procedure in slave mode and the I2C Ch.*n* operations are shown below. Figures 14.4.6.1 and 14.4.6.2 show an operation example and a flowchart, respectively.

Data receiving procedure

1. When receiving one-byte data, write 1 to the I2C*n*CTL.TXNACK bit.
2. Wait for a START condition interrupt (I2C*n*INTF.STARTIF bit = 1).
3. 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.)
4. Clear the I2C*n*INTF.STARTIF bit by writing 1.
5. 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.
6. 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.
7. Read the received data from the I2C*n*RXD register.
8. Repeat Steps 5 to 7 until the end of data reception.
9. 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 10 when a STOP condition interrupt has occurred.
   ii. Go to Step 3 when a START condition interrupt has occurred.
10. 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.

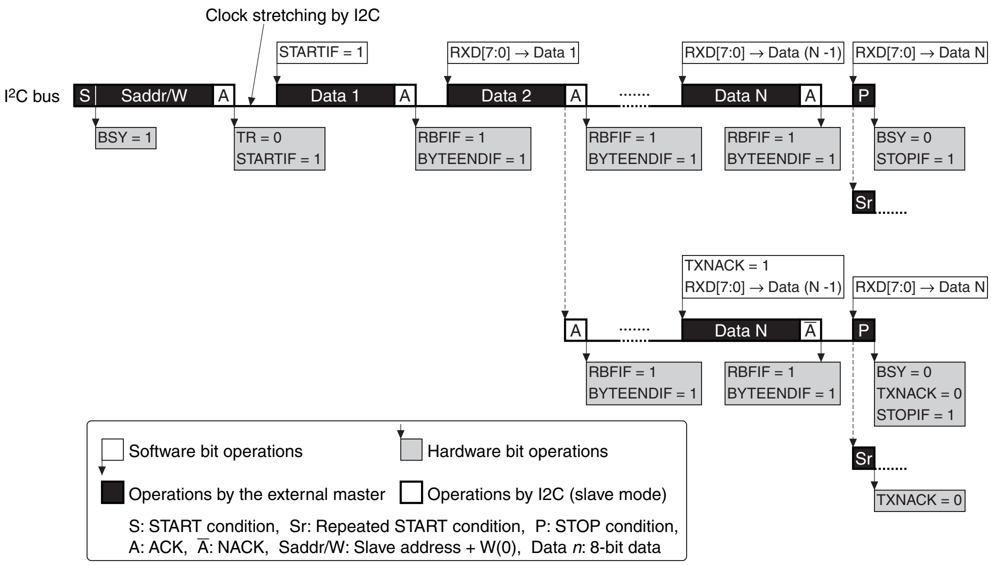


Figure 14.4.6.1 Example of Data Receiving Operations in Slave Mode

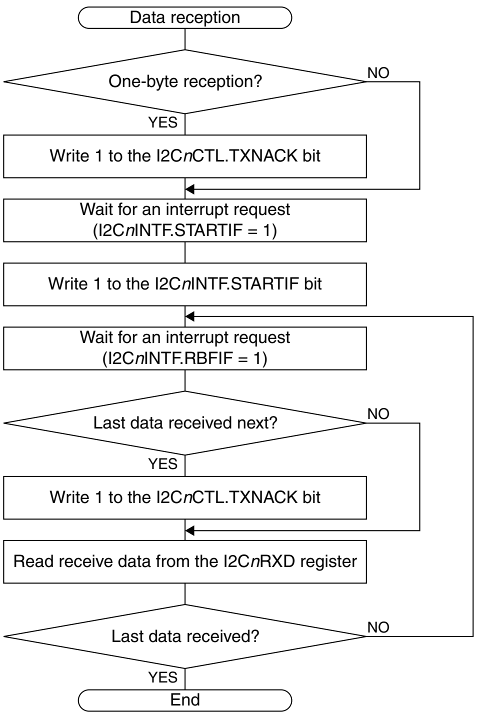


Figure 14.4.6.2 Slave Mode Data Reception Flowchart

14.4.7 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 14.4.7.1 shows an operation example. See Figure 14.4.4.1 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.

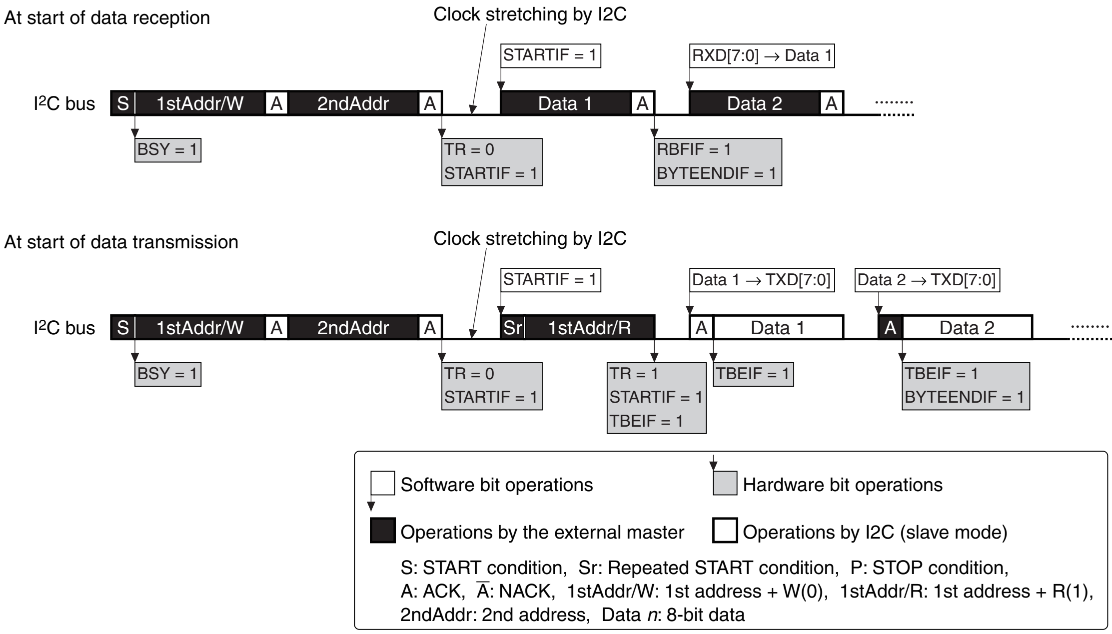


Figure 14.4.7.1 Example of Data Transfer Starting Operations in 10-bit Address Mode (Slave Mode)

14.4.8 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.

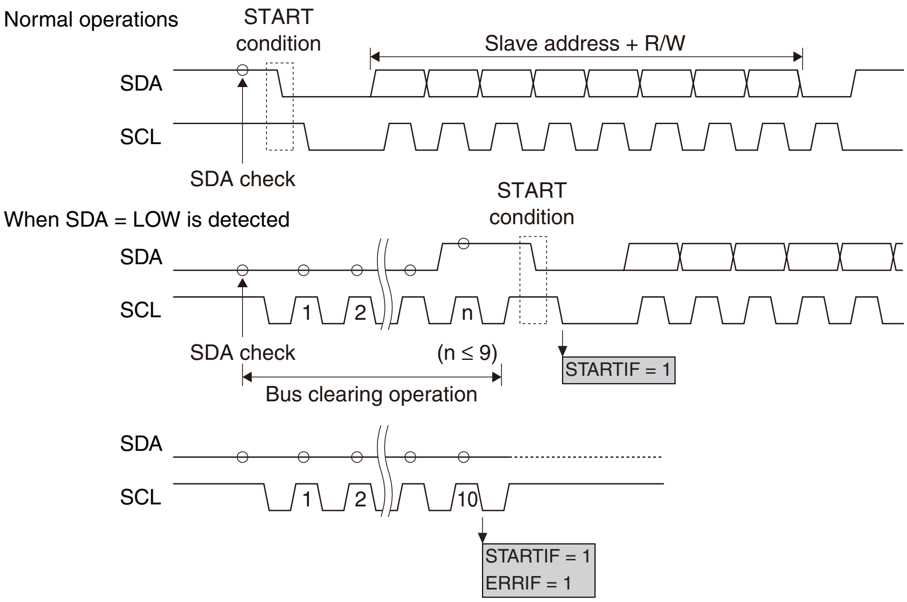


Figure 14.4.8.1 Automatic Bus Clearing Operation

14.4.9 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 14.4.9.1 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

14.5 Interrupts

The I2C has a function to generate the interrupts shown in Table 14.5.1.

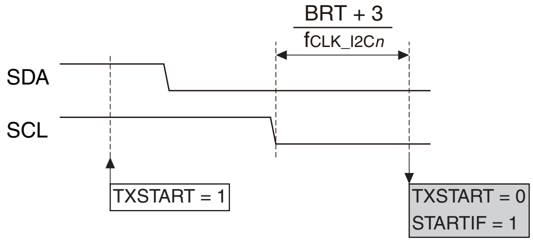
Table 14.5.1 I2C Interrupt Function

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

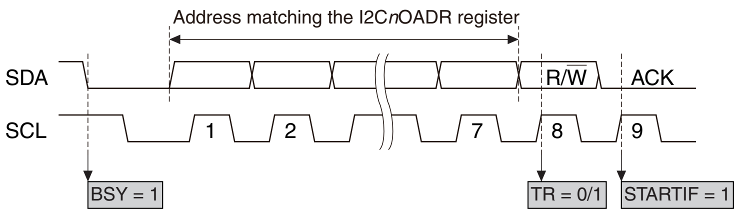
The I2C provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the interrupt controller only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the "Interrupt Controller" chapter.

(1) START condition interrupt

Master mode

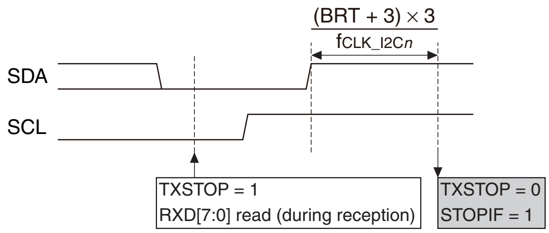


Slave mode

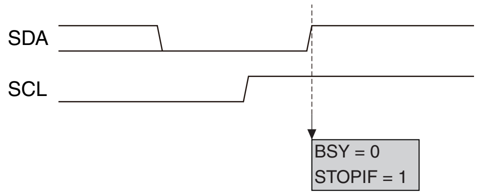


(2) STOP condition interrupt

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 14.5.1 START/STOP Condition Interrupt Timings

14.6 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	–	0	–	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 14.6.1 Clock Source and Division Ratio Settings

I2C*n*CLK.CLKDIV[1:0] bits	I2C*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	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 (t_{BUF} defined in the I^{2}C 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

15 16-bit PWM Timers (T16B)

15.1 Overview

T16B is a 16-bit PWM timer with comparator/capture functions. The features of T16B are listed below.

- Counter block
  - 16-bit up/down counter
  - A clock source and a clock division ratio for generating the count clock are selectable in each channel.
  - The count mode is configurable from combinations of up, down, or up/down count operations, and one-shot operations (counting for one cycle configured) or repeat operations (counting continuously until stopped via software).
  - Supports an event counter function using an external clock.
- Comparator/capture block
  - Supports up to six comparator/capture circuits to be included per one channel.
  - The comparator compares the counter value with the values specified via software to generate interrupt signals and a PWM waveform. (Can be used as an interval timer, PWM waveform generator, and external event counter.)
  - The capture circuit captures counter values using external/software trigger signals and generates interrupts. (Can be used to measure external event periods/cycles.)

Figure 15.1.1 shows the T16B configuration.

Table 15.1.1 T16B Channel Configuration of S1C17W15

Item	S1C17W15
Number of channels	2 channels (Ch.0 and Ch.1)
Event counter function	Ch.0: EXCL00 or EXCL01 pin input Ch.1: EXCL10 or EXCL11 pin input
Number of comparator/ capture circuits per channel	2 systems (0 and 1)
Timer generating signal output	Ch.0: TOUT00 and TOUT01 pin outputs (2 systems) Ch.1: TOUT10 and TOUT11 pin outputs (2 systems)
Capture signal input	Ch.0: CAP00 and CAP01 pin inputs (2 systems) Ch.1: CAP10 and CAP11 pin inputs (2 systems)

Note: In this chapter, '*n*' refers to a channel number, and '*m*' refers to an input/output pin number or a comparator/capture circuit number in a channel.

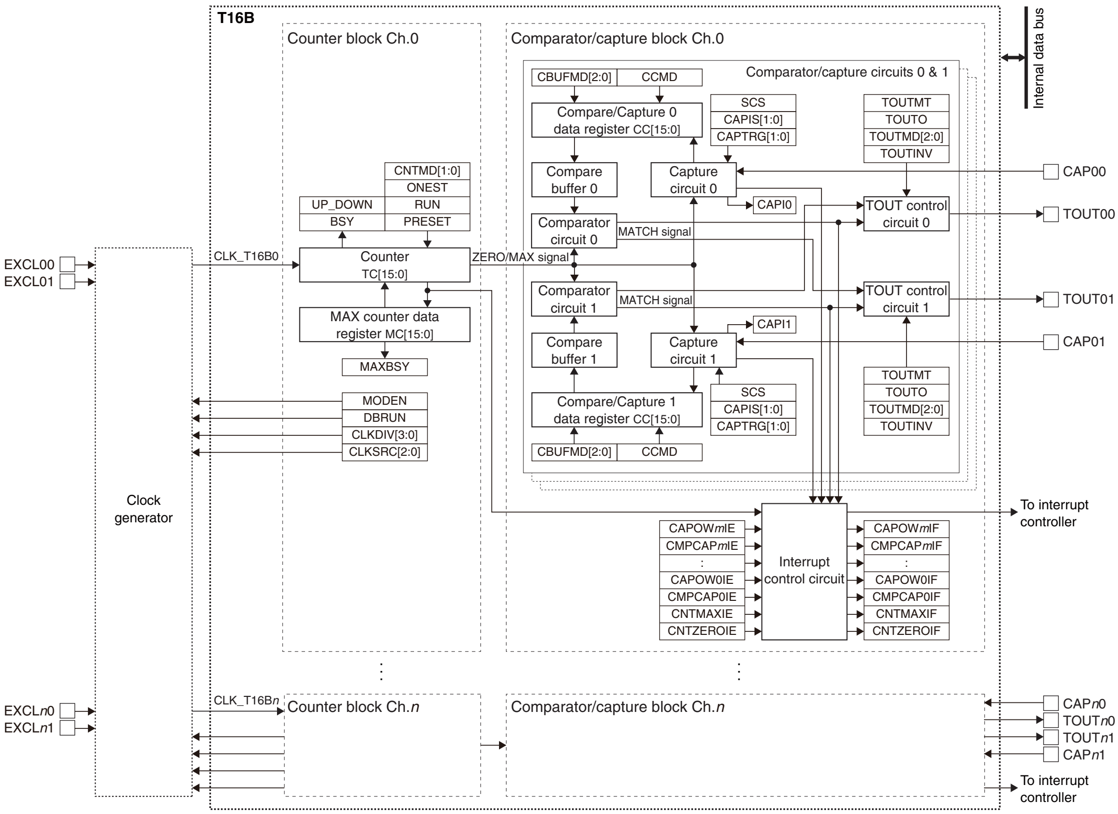


Figure 15.1.1 T16B Configuration

15.2 Input/Output Pins

Table 15.2.1 lists the T16B pins.

Table 15.2.1 List of T16B Pins

Pin name	I/O*	Initial status*	Function
EXCL*nm*	I	I (Hi-Z)	External clock input
TOUT*nm*/CAP*nm*	O or I	O (L)	TOUT signal output (in comparator mode) or capture trigger signal input (in capture mode)

* Indicates the status when the pin is configured for T16B.

If the port is shared with the T16B pin and other functions, the T16B input/output function must be assigned to the port before activating T16B. For more information, refer to the “I/O Ports” chapter.

15.3 Clock Settings

15.3.1 T16B Operating Clock

When using T16B Ch.*n*, the T16B Ch.*n* operating clock CLK_T16B*n* must be supplied to T16B Ch.*n* from the clock generator. The CLK_T16B*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*nm* pin function (refer to the “I/O Ports” chapter).

2. Set the following T16B*n*CLK register bits:
   - T16B*n*CLK.CLKSRC[2:0] bits (Clock source selection)
   - T16B*n*CLK.CLKDIV[3:0] bits (Clock division ratio selection = Clock frequency setting)

15.3.2 Clock Supply in SLEEP Mode

When using T16B during SLEEP mode, the T16B operating clock CLK_T16B*n* must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_T16B*n* clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_T16B*n* clock source is 1, the CLK_T16B*n* clock source is deactivated during SLEEP mode and T16B stops with the register settings and counter value maintained at those before entering SLEEP mode. After the CPU returns to normal mode, CLK_T16B*n* is supplied and the T16B operation resumes.

15.3.3 Clock Supply in DEBUG Mode

The CLK_T16B*n* supply during DEBUG mode should be controlled using the T16B*n*CLK.DBRUN bit.
The CLK_T16B*n* supply to T16B Ch.*n* is suspended when the CPU enters DEBUG mode if the T16B*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_T16B*n* supply resumes. Although T16B Ch.*n* stops operating when the CLK_T16B*n* supply is suspended, the counter and registers retain the status before DEBUG mode was entered. If the T16B*n*CLK.DBRUN bit = 1, the CLK_T16B*n* supply is not suspended and T16B Ch.*n* will keep operating in DEBUG mode.

15.3.4 Event Counter Clock

When EXCL*nm* is selected as the clock source using the T16B*n*CLK.CLKSRC[2:0] bits, the channel functions as a timer or event counter that counts the EXCL*nm* pin input clocks.
The counter counts rising edges of the input signal. This can be changed so that the counter will count falling edges of the original signal by selecting EXCL*nm* inverted input as the clock source.

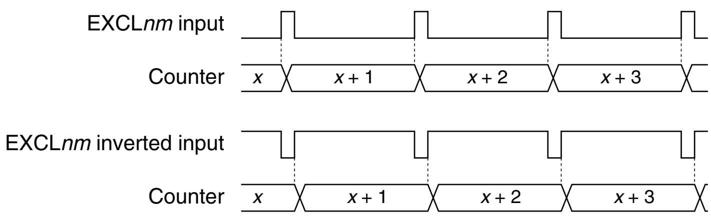


Figure 15.3.4.1 Count Timing (During Count Up Operation)

Note: When running the counter using the event counter clock, two dummy clocks must be input before the first counting up/down can be performed.

15.4 Operations

15.4.1 Initialization

T16B Ch.*n* should be initialized and started counting with the procedure shown below. Perform initial settings for comparator mode when using T16B as an interval timer, PWM waveform generator, or external event counter. Perform initial settings for capture mode when using T16B to measure external event periods/cycles.

Initial settings for comparator mode

1. Configure the T16B Ch.*n* operating clock.
2 Set the T16B*n*CTL.MODEN bit to 1. (Enable T16B operations)
3. Set the following T16B*n*CCCTL0 and T16B*n*CCCTL1 register bits:
   - Set the T16B*n*CCCTL*m*.CCMD bit to 0. * (Set comparator mode)
   - T16B*n*CCCTL*m*.CBUFMD[2:0] bits (Configure compare buffer)

   * Another circuit in the comparator/capture circuit pair (circuits 0 and 1, 2 and 3, 4 and 5) can be set to capture mode.

   Set the following bits when the TOUT*nm* output is used.
   - T16B*n*CCCTL*m*.TOUTMT bit (Select waveform generation signal)
   - T16B*n*CCCTL*m*.TOUTMD[2:0] bits (Select TOUT signal generation mode)
   - T16B*n*CCCTL*m*.TOUTINV bit (Select TOUT signal polarity)
4. Set the T16B*n*MC register. (Set MAX counter data)
5. Set the T16B*n*CCR0 and T16B*n*CCR1 registers. (Set the counter comparison value)
6. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the T16B*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the T16B*n*INTE register to 1. (Enable interrupts)
7. Set the following T16B*n*CTL register bits:
   - T16B*n*CTL.CNTMD[1:0] bits (Select count up/down operation)
   - T16B*n*CTL.ONEST bit (Select one-shot/repeat operation)
   - Set the T16B*n*CTL.PRESET bit to 1. (Reset counter)
   - Set the T16B*n*CTL.RUN bit to 1. (Start counting)

Initial settings for capture mode

1. Configure the T16B Ch.*n* operating clock.
2 Set the T16B*n*CTL.MODEN bit to 1. (Enable T16B operations)
3. Set the following T16B*n*CCCTL0 and T16B*n*CCCTL1 register bits:
   - Set the T16B*n*CCCTL*m*.CCMD bit to 1. * (Set capture mode)
   - T16B*n*CCCTL*m*.SCS bit (Set synchronous/asynchronous mode)
   - T16B*n*CCCTL*m*.CAPIS[1:0] bits (Set trigger signal)
   - T16B*n*CCCTL*m*.CAPTRG[1:0] bits (Select trigger edge)

   * Another circuit in the comparator/capture circuit pair (circuits 0 and 1, 2 and 3, 4 and 5) can be set to comparator mode.
4. Set the T16B*n*MC register. (Set MAX counter data)
5. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the T16B*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the T16B*n*INTE register to 1. (Enable interrupts)
6. Set the following T16B*n*CTL register bits:
   - T16B*n*CTL.CNTMD[1:0] bits (Select count up/down operation)
   - T16B*n*CTL.ONEST bit (Select one-shot/repeat operation)
   - Set the T16B*n*CTL.PRESET bit to 1. (Reset counter)
   - Set the T16B*n*CTL.RUN bit to 1. (Start counting)

15.4.2 Counter Block Operations

The counter in each counter block channel is a 16-bit up/down counter that counts the selected operating clock (count clock).

Count mode

The T16B*n*CTL.CNTMD[1:0] bits allow selection of up, down, and up/down mode. The T16B*n*CTL.ONEST bit allows selection of repeat and one-shot mode. The counter operates in six counter modes specified with a combination of these modes.

Repeat mode enables the counter to continue counting until stopped via software. Select this mode to generate periodic interrupts at desired intervals or to generate timer output waveforms.

One-shot mode enables the counter to stop automatically. Select this mode to stop the counter after an interrupt has occurred once, such as for measuring pulse width or external event intervals and checking a specific lapse of time.

Up, down, and up/down mode configures the counter as an up counter, down counter and up/down counter, respectively.

MAX counter data register

The MAX counter data register (T16B*n*MC.MC[15:0] bits) is used to set the maximum value of the counter (hereafter referred to as MAX value). This setting limits the count range to 0x0000–MAX value and determines the count and interrupt cycles. When the counter is set to repeat mode, the MAX value can be rewritten in the procedure shown below even if the counter is running.

1. Check to see if the T16B*n*CTL.MAXBSY bit is set to 0.
2. Write the MAX value to the T16B*n*MC.MC[15:0] bits.

Note: When rewriting the MAX value, the new MAX value should be written after the counter has been reset to the previously set MAX value.

Counter reset

Setting the T16B*n*CTL.PRESET bit to 1 resets the counter. This clears the counter to 0x0000 in up or up/down mode, or presets the MAX value to the counter in down mode.

The counter is also cleared to 0x0000 when the counter value exceeds the MAX value during count up operation.

Counting start

To start counting, set the T16B*n*CTL.RUN bit to 1. The counting stop control depends on the count mode set.

Counter value read

The counter value can be read out from the T16B*n*TC.TC[15:0] bits. However, since T16B operates on CLK_T16B*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.

Counter status check

The counter operating status can be checked using the T16B*n*CS.BSY bit. The T16B*n*CS.BSY bit is set to 1 while the counter is running or 0 while the counter is idle.

The current count direction can also be checked using the T16B*n*CS.UP_DOWN bit. The T16B*n*CS.UP_DOWN bit is set to 1 during count up operation or 0 during count down operation.

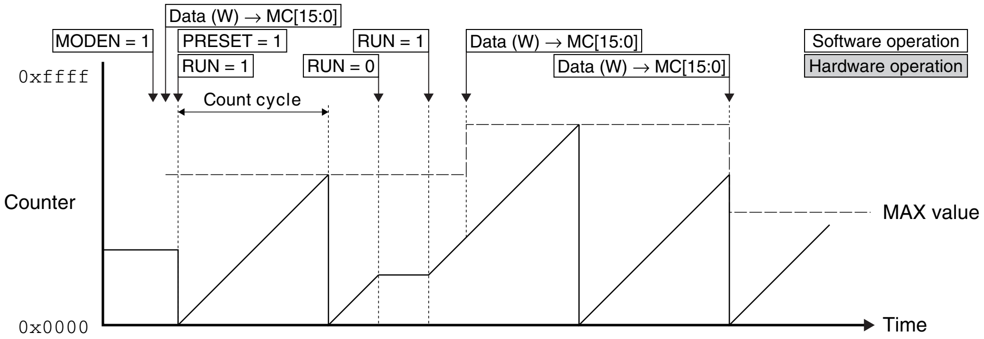
Operations in repeat up count and one-shot up count modes

In these modes, the counter operates as an up counter and counts from 0x0000 (or current value) to the MAX value.

In repeat up count mode, the counter returns to 0x0000 if it exceeds the MAX value and continues counting until the T16B*n*CTL.RUN bit is set to 0. If the MAX value is altered to a value larger than the current counter value during counting, the counter keeps counting up to the new MAX value. If the MAX value is altered to a value smaller than the current counter value, the counter is cleared to 0x0000 and continues counting up to the new MAX value.

In one-shot up count mode, the counter returns to 0x0000 if it exceeds the MAX value and stops automatically at that point.

(1) Repeat up count mode



(2) One-shot up count mode

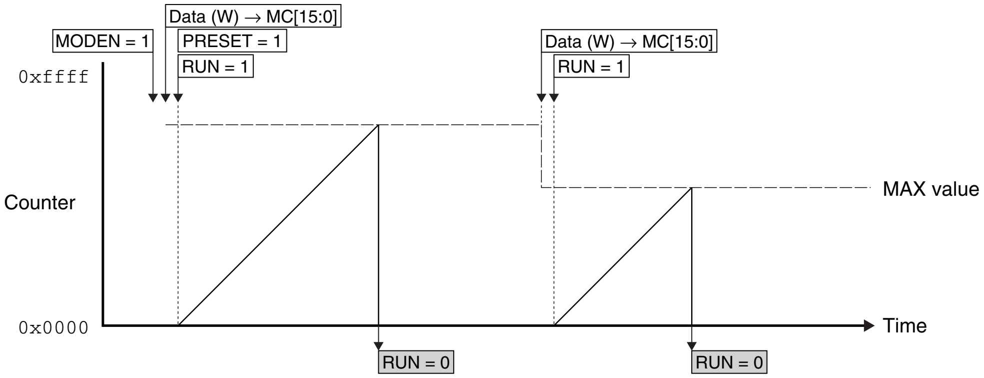


Figure 15.4.2.1 Operations in Repeat Up Count and One-shot Up Count Modes

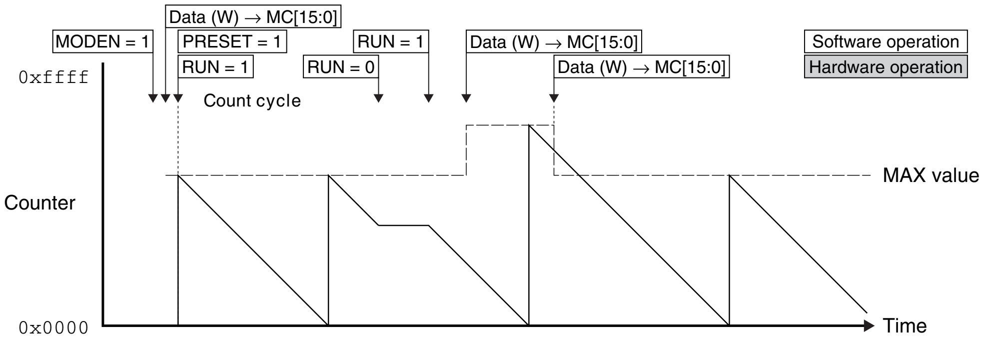
Operations in repeat down count and one-shot down count modes

In these modes, the counter operates as a down counter and counts from the MAX value (or current value) to 0x0000.

In repeat down count mode, the counter returns to the MAX value if a counter underflow occurs and continues counting until the T16B*n*CTL.RUN bit is set to 0. If the MAX value is altered during counting, the counter keeps counting down to 0x0000 and continues counting down from the new MAX value after a counter underflow occurs.

In one-shot down count mode, the counter returns to the MAX value if a counter underflow occurs and stops automatically at that point.

(1) Repeat down count mode



(2) One-shot down count mode

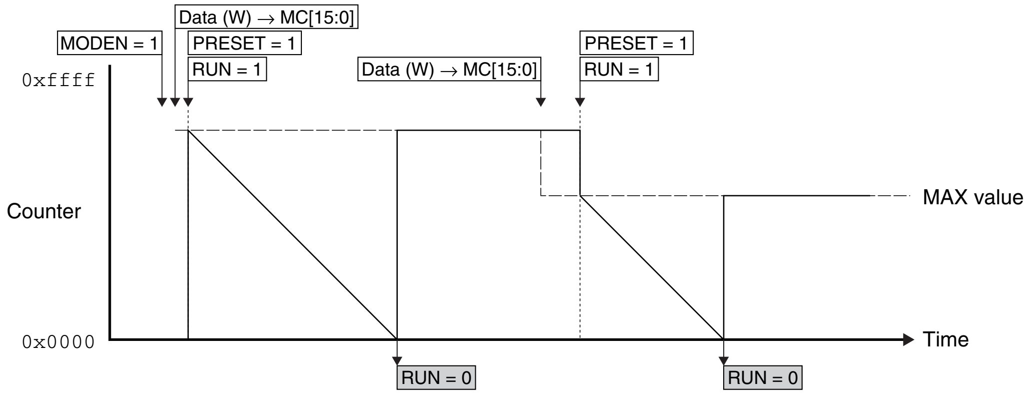


Figure 15.4.2.2 Operations in Repeat Down Count and One-shot Down Count Modes

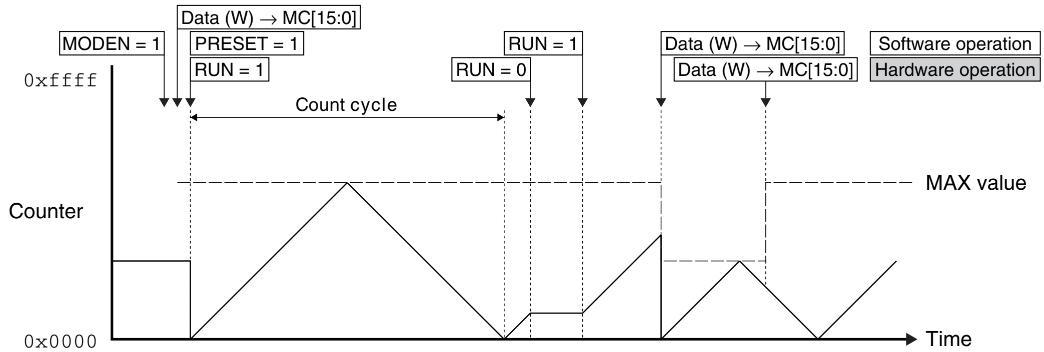
Operations in repeat up/down count and one-shot up/down count modes

In these modes, the counter operates as an up/down counter and counts as 0x0000 (or current value) → the MAX value → 0x0000.

In repeat up/down count mode, the counter repeats counting up from 0x0000 to the MAX value and counting down from the MAX value to 0x0000 until the T16B*n*CTL.RUN bit is set to 0. If the MAX value is altered to a value larger than the current counter value during count up operation, the counter keeps counting up to the new MAX value. If the MAX value is altered to a value smaller than the current counter value, the counter is cleared to 0x0000 and continues counting up to the new MAX value. If the MAX value is altered during count down operation, the counter keeps counting down to 0x0000 and then starts counting up to the new MAX value.

In one-shot up/down count mode, the counter stops automatically when it reaches 0x0000 during count down operation.

(1) Repeat up/down count mode



(2) One-shot up/down count mode

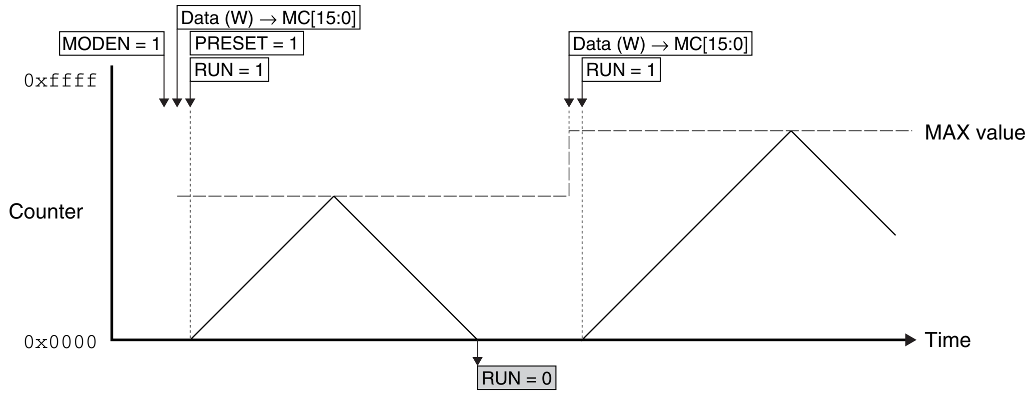


Figure 15.4.2.3 Operations in Repeat Up/Down Count and One-shot Up/Down Count Modes

15.4.3 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 circuit to capture counter values using the external/software trigger signals.

Comparator/capture block operating mode

The comparator/capture block includes two systems (four or six systems) of comparator/capture circuits and each system can be set to comparator mode or capture mode, individually.

Set the T16B*n*CCCTL*m*.CCMD bit to 0 to set the comparator/capture circuit *m* to comparator mode or 1 to set it to capture mode.

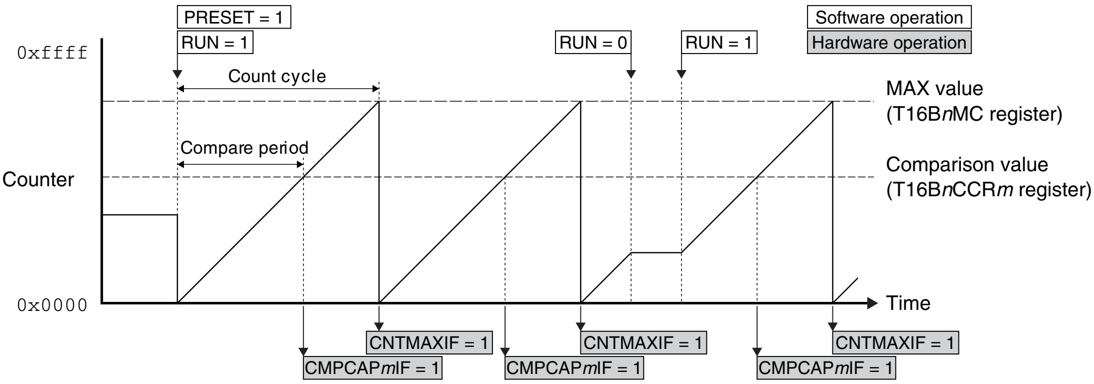
Operations in comparator mode

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 T16B*n*CCR*m* register functions as the compare data register used for setting a comparison value in this mode. The TOUT*nm*/CAP*nm* pin is configured to the TOUT*nm* pin.

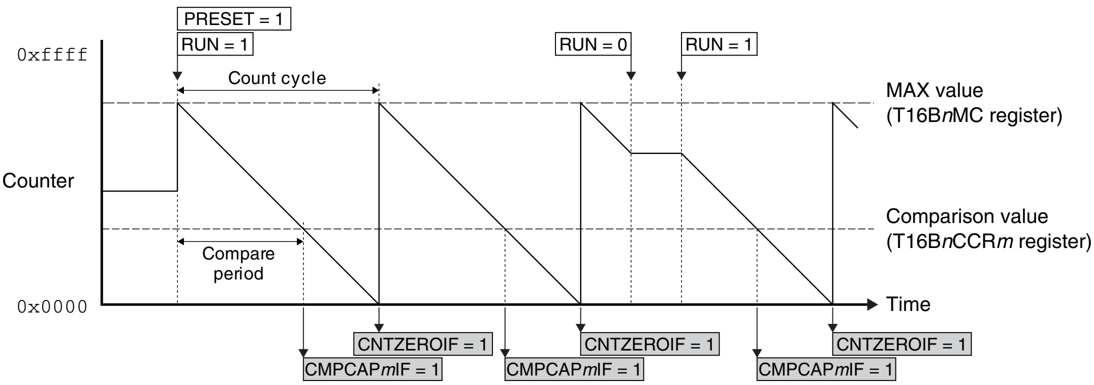
When the counter reaches the value set in the T16B*n*CCR*m* register during counting, the comparator asserts the MATCH signal and sets the T16B*n*INTF.COMPCAP*m*IF bit (compare interrupt flag) to 1.

When the counter reaches the MAX value in comparator mode, the T16B*n*INTF.CNTMAXIF bit (counter MAX interrupt flag) is set to 1. When the counter reaches 0x0000, the T16B*n*INTF.CNTZEROIF bit (counter zero interrupt flag) is set to 1.

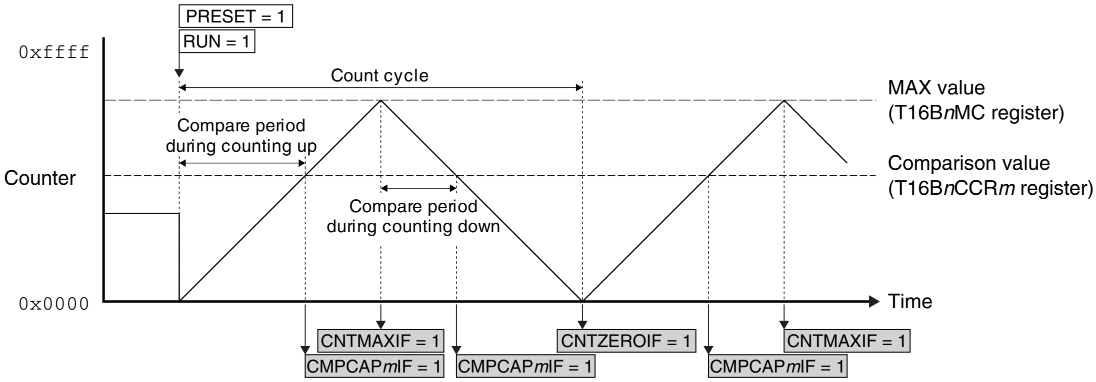
(1) Repeat up count mode



(2) Repeat down count mode



(3) Repeat up/down count mode



(Note that the T16B*n*INTF.CMPCAP*m*IF/CNTMAXIF/CNTZEROIF bit clearing operations via software are omitted from the figure.)

Figure 15.4.3.1 Operation Examples in Comparator Mode

The time from counter = 0x0000 or MAX value to occurrence of a compare interrupt (compare period) and the time to occurrence of a counter MAX or counter zero interrupt (count cycle) can be calculated as follows:

During counting up

$$\text{Compare period} = \frac{(CC + 1)}{f_{CLK\_T16B}} \text{ [s]} \qquad \text{Count cycle} = \frac{(MAX + 1)}{f_{CLK\_T16B}} \text{ [s]} \qquad \text{(Eq. 15.1)}$$

During counting down

$$\text{Compare period} = \frac{(MAX - CC + 1)}{f_{CLK\_T16B}} \text{ [s]} \qquad \text{Count cycle} = \frac{(MAX + 1)}{f_{CLK\_T16B}} \text{ [s]} \qquad \text{(Eq. 15.2)}$$

Where

CC: T16B*n*CCR*m* register setting value (0 to 65,535)

MAX: T16B*n*MC register setting value (0 to 65,535)

f_{CLK_T16B}: Count clock frequency [Hz]

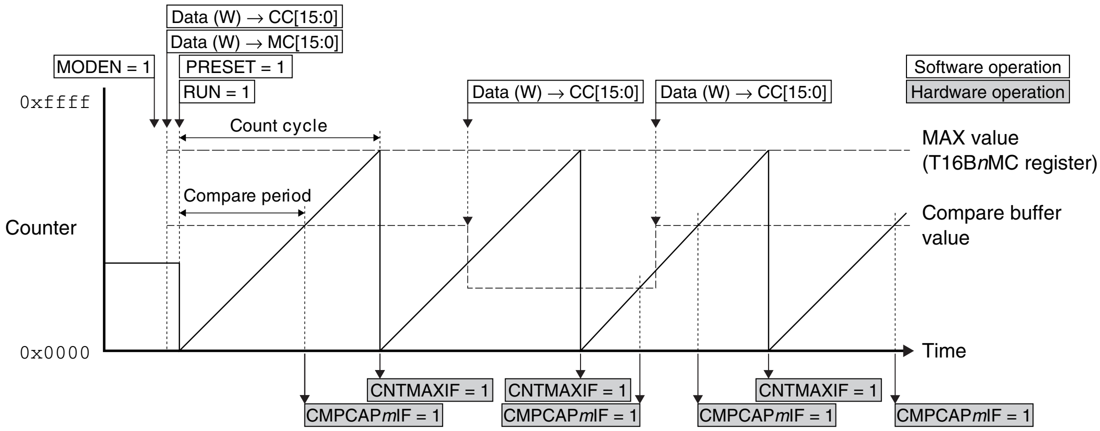
The comparator MATCH signal and counter MAX/ZERO signals are also used to generate a timer output waveform (TOUT). Refer to "TOUT Output Control" for more information.

Compare buffer

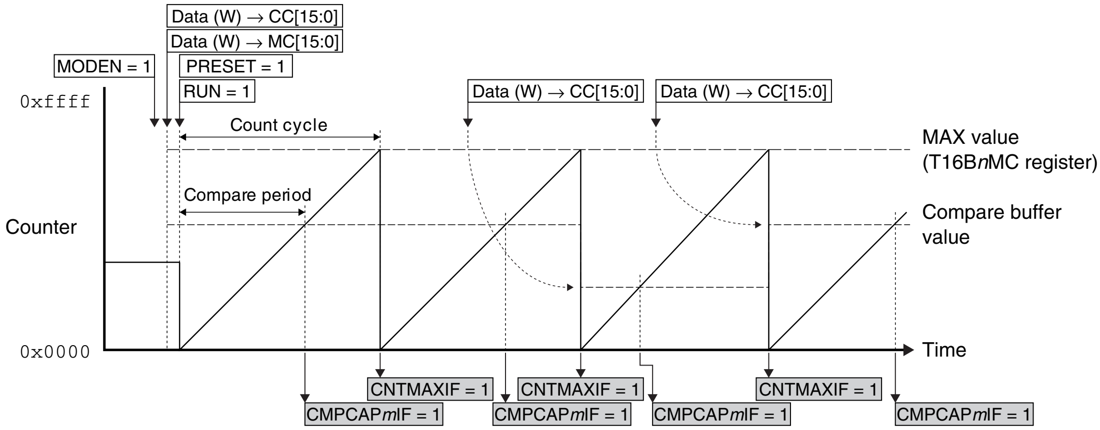
The comparator loads the comparison value, which has been written to the T16B*n*CCR*m* register, to the compare buffer before comparing it with the counter value. For example, when generating a PWM waveform, the waveform with the desired duty ratio may not be generated if the comparison value is altered asynchronous to the count operation. To avoid this problem, the timing to load the comparison value to the compare buffer can be configured using the T16B*n*CCCTL*m*.CBUFMD[2:0] bits for synchronization with the count operation.

(1) Repeat up count mode

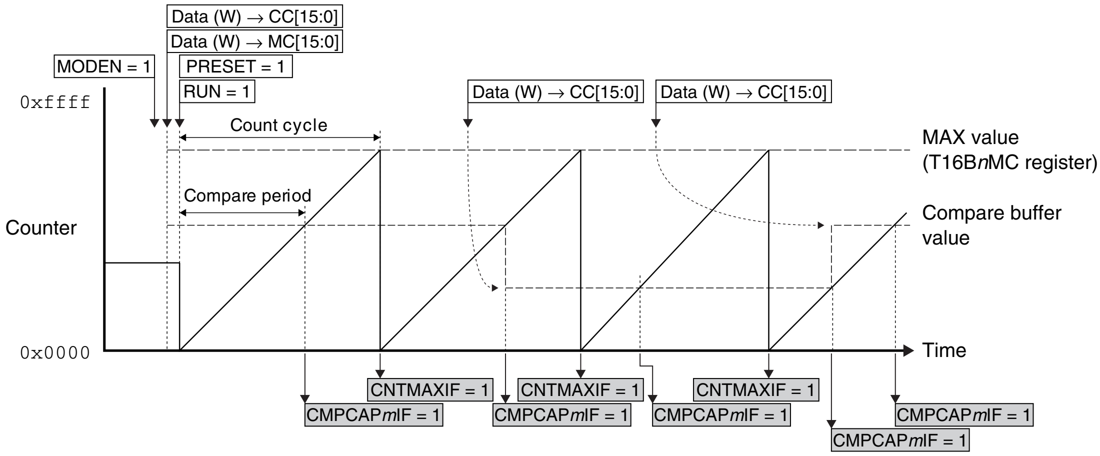
(1.1) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x0



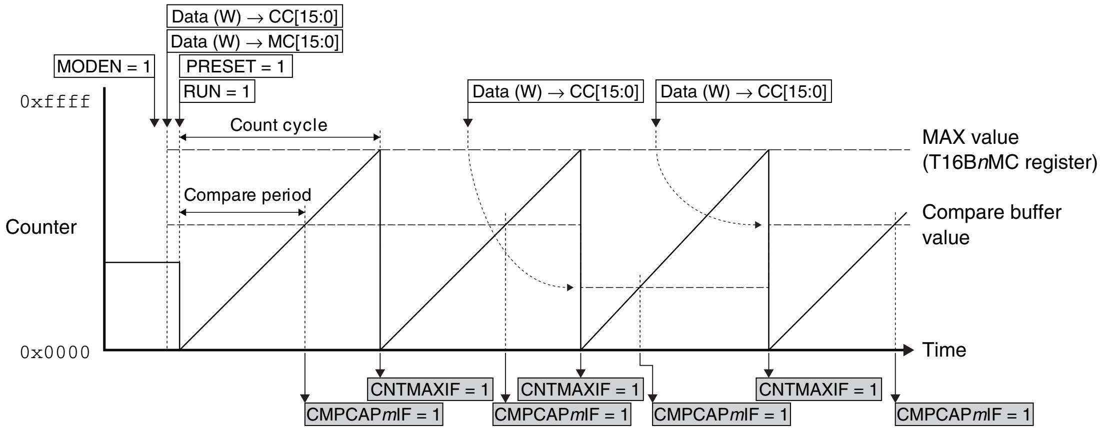
(1.2) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x1



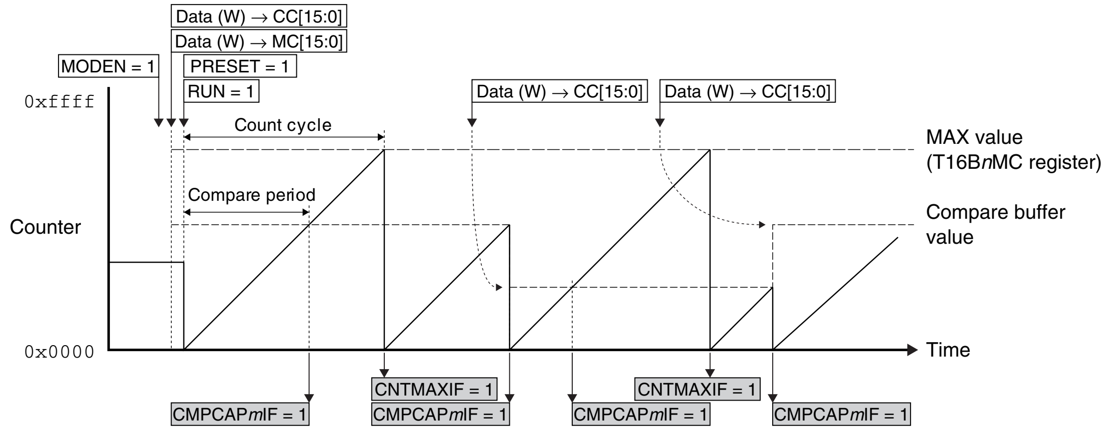
(1.3) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x2



(1.4) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x3

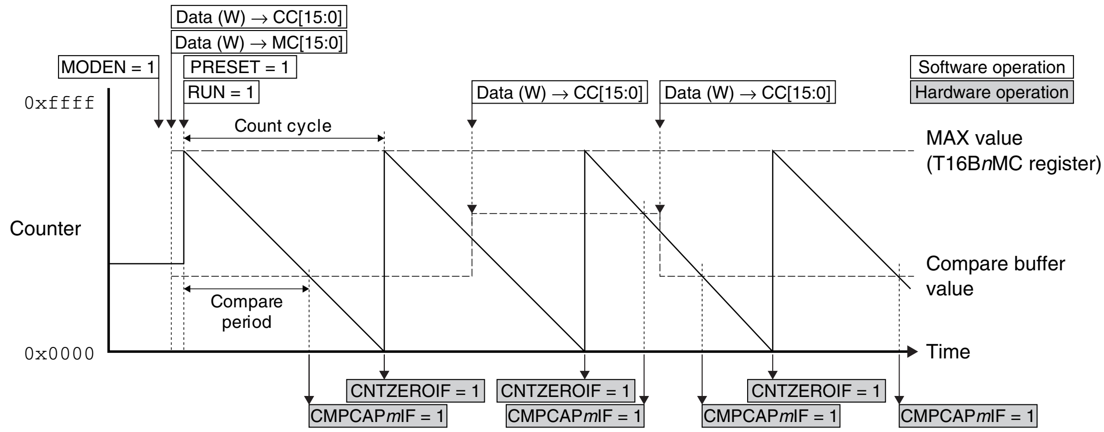


(1.5) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x4

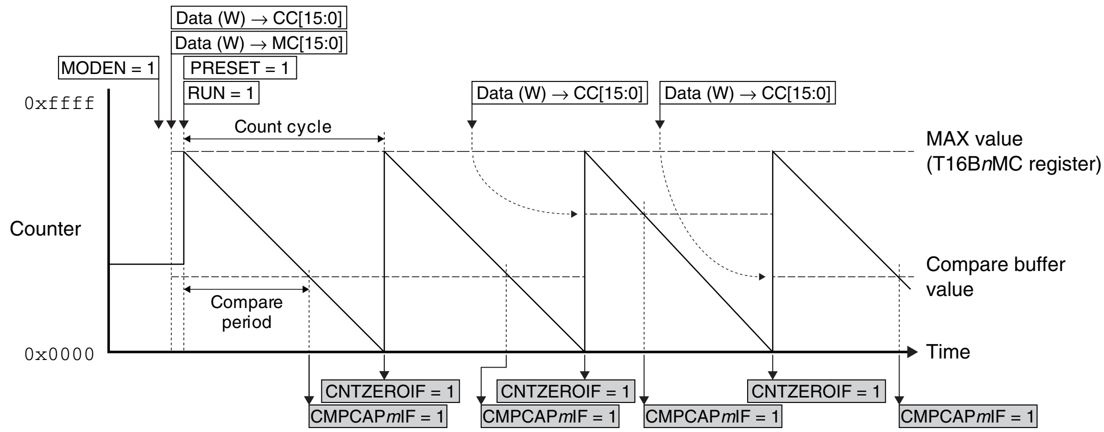


(2) Repeat down count mode

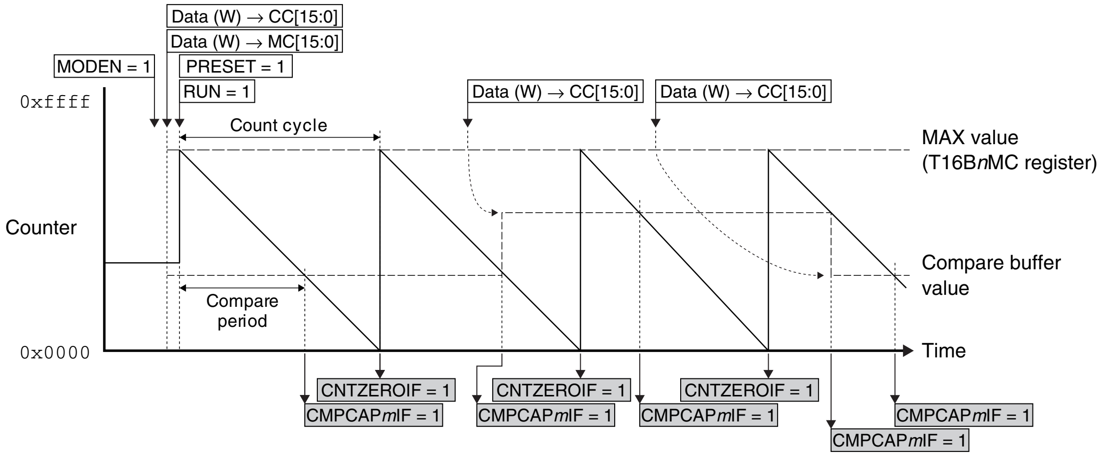
(2.1) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x0



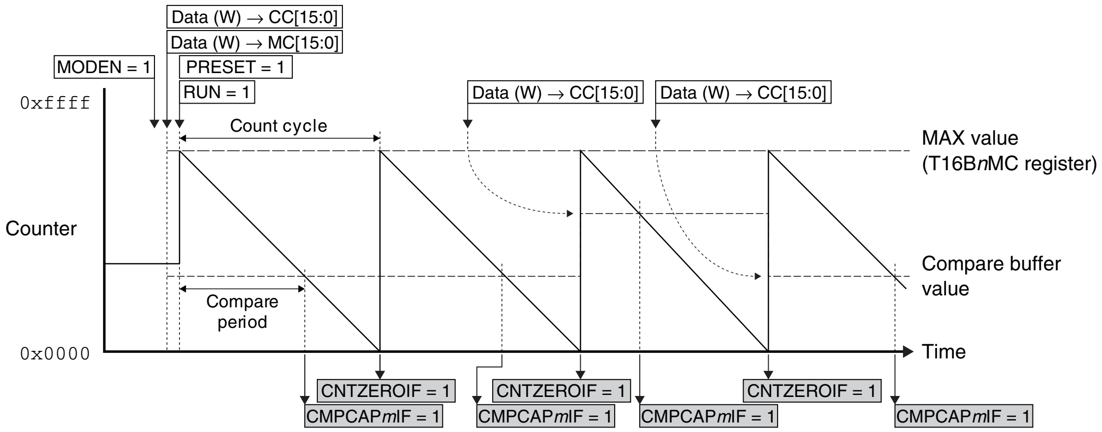
(2.2) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x1



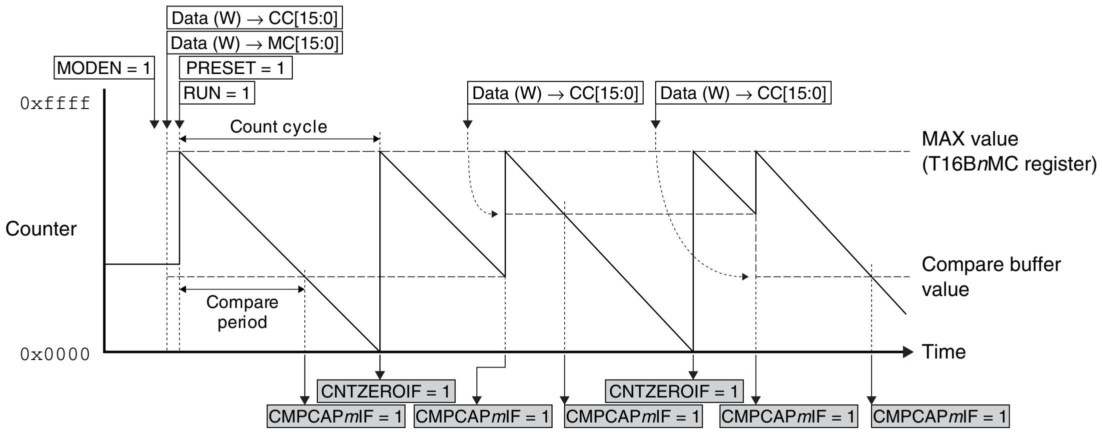
(2.3) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x2



(2.4) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x3

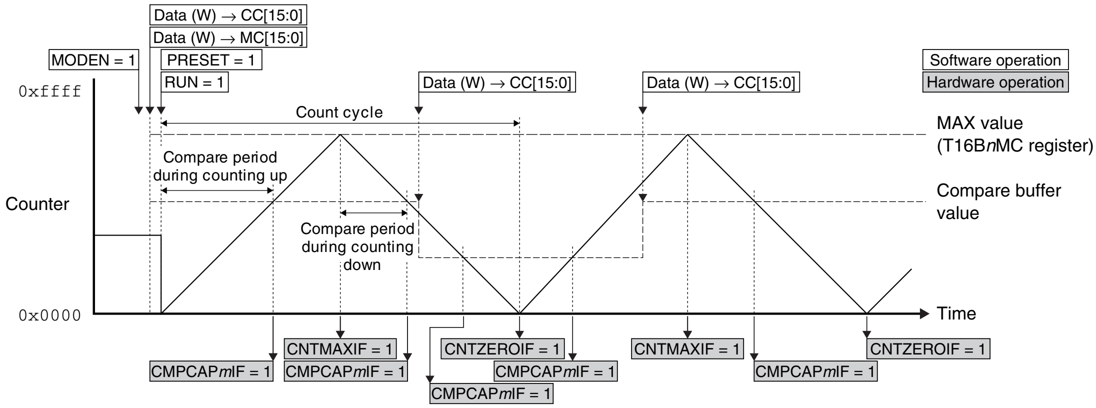


(2.5) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x4

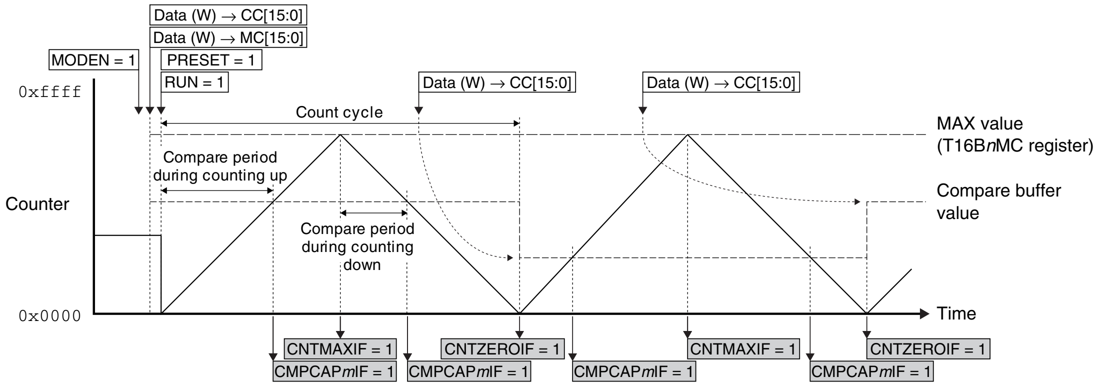


(3) Repeat up/down count mode

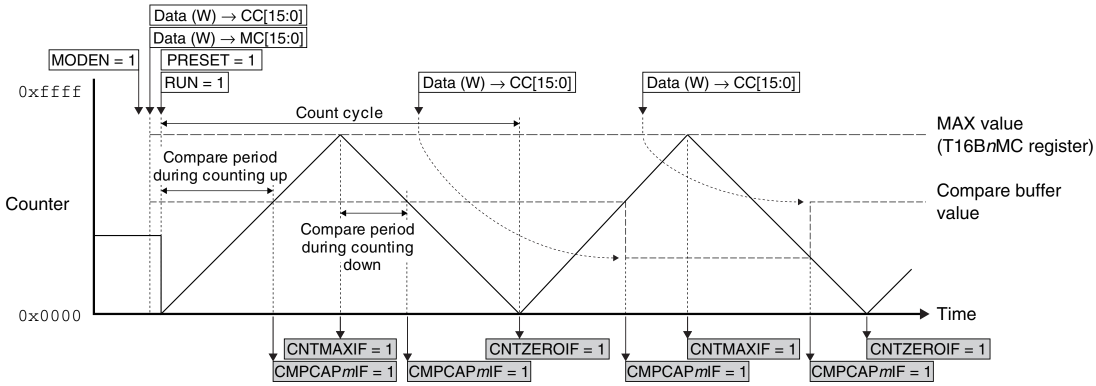
(3.1) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x0

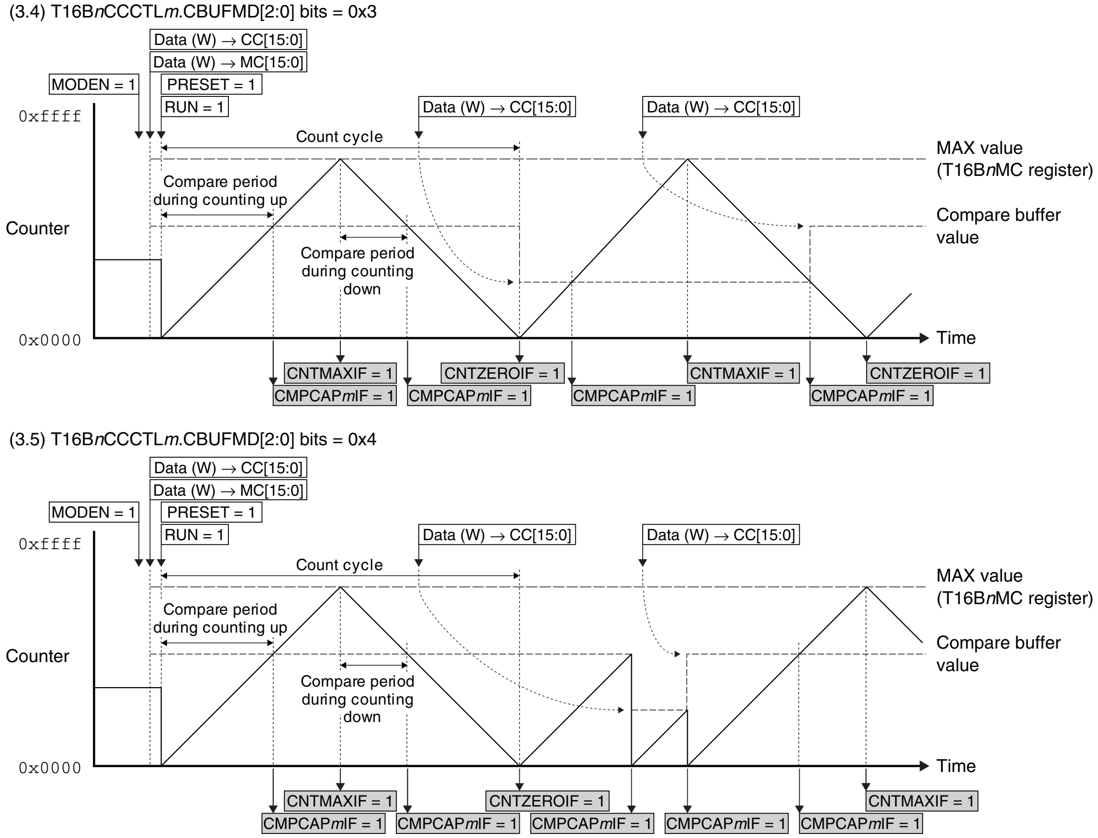


(3.2) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x1



(3.3) T16B*n*CCCTL*m*.CBUFMD[2:0] bits = 0x2





(Note that the T16B*n*INTF.CMPCAP*m*IF/CNTMAXIF/CNTZEROIF bit clearing operations via software are omitted from the figure.)

Figure 15.4.3.2 Compare Buffer Operations

Operations in capture mode

The capture mode captures the counter value when an external event, such as a key entry, occurs (at the specified edge of the external input/software trigger signal). In this mode, the T16B*n*CCR*m* register functions as the capture register from which the captured data is read. Furthermore, the TOUT*nm*/CAP*nm* pin is configured to the CAP*nm* pin.

The trigger signal and the trigger edge to capture the counter value are selected using the T16B*n*CCCTL*m*.CAPIS[1:0] bits and the T16B*n*CCCTL*m*.CAPTRG[1:0] bits, respectively.

When a specified trigger edge is input during counting, the current counter value is loaded to the T16B*n*CCR*m* register. At the same time the T16B*n*INTF.CMPCAP*m*IF bit is set. The interrupt occurred by this bit can be used to read the captured data from the T16B*n*CCR*m* 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 T16B*n*CCR*m* register is overwritten by the next trigger when the T16B*n*INTF.CMPCAP*m*IF bit is still set, an overwrite error occurs (the T16B*n*INTF.CAPOW*m*IF bit is set).

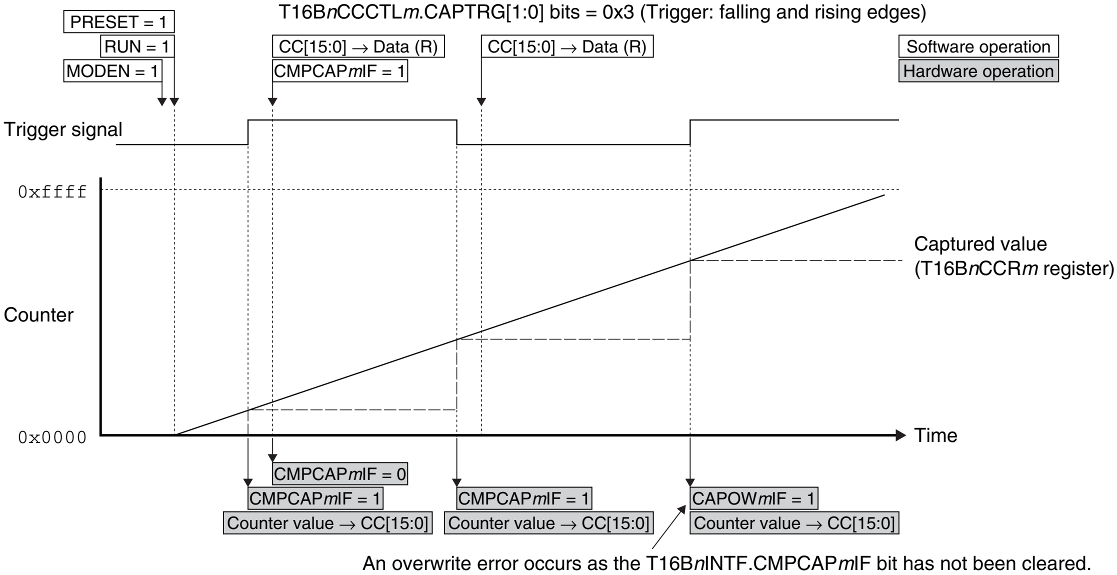


Figure 15.4.3.3 Operations in Capture Mode (Example in One-shot Up Count Mode)

Synchronous capture mode/asynchronous capture mode

The capture circuit can operate in two operating modes: synchronous capture mode and asynchronous capture mode.

Synchronous capture mode is provided to avoid the possibility of invalid data reading by capturing counter data simultaneously with the counter being counted up/down. Set the T16B*n*CCCTL*m*.SCS bit to 1 to set the capture circuit to synchronous capture mode. This mode captures counter data by synchronizing the capture signal with the counter clock.

On the other hand, asynchronous capture mode can capture counter data by detecting a trigger pulse even if the pulse is shorter than the counter clock cycle that becomes invalid in synchronous capture mode. Set the T16B*n*CCCTL*m*.SCS bit to 0 to set the capture circuit to asynchronous capture mode.

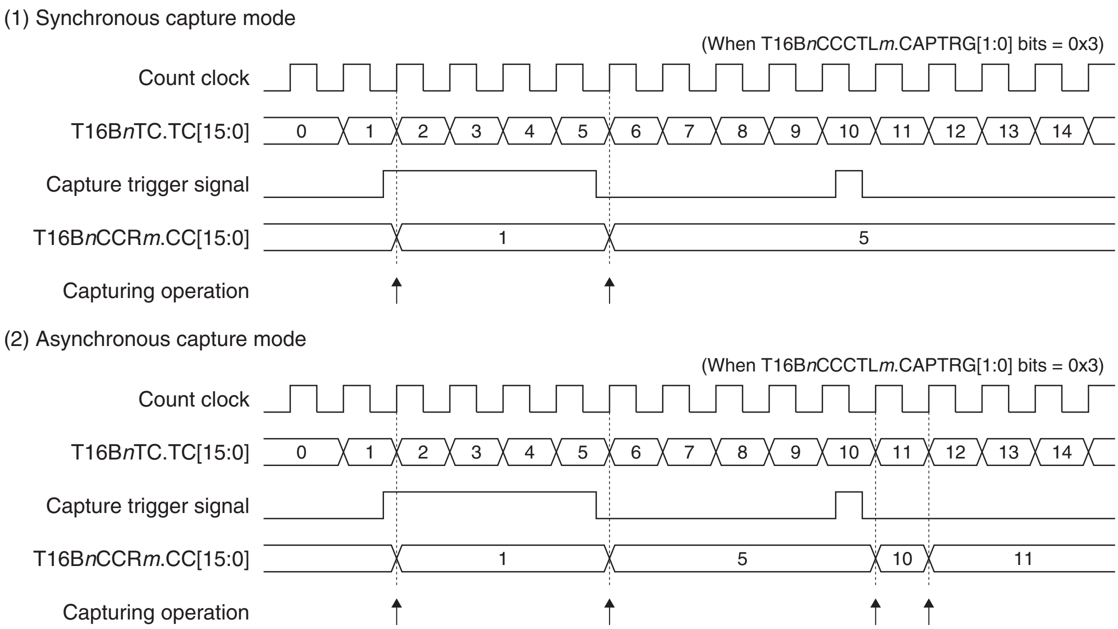


Figure 15.4.3.4 Synchronous Capture Mode/Asynchronous Capture Mode

15.4.4 TOUT Output Control

Comparator mode can generate TOUT signals using the comparator MATCH and counter MAX/ZERO signals. The generated signals can be output to outside the IC. Figure 15.4.4.1 shows the TOUT output circuits (circuits 0 and 1).

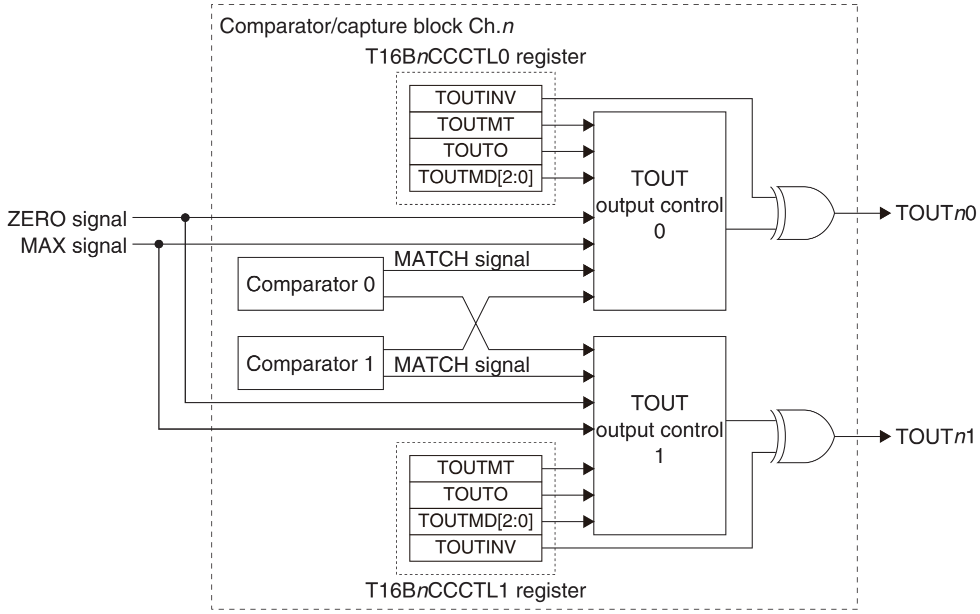


Figure 15.4.4.1 TOUT Output Circuits (Circuits 0 and 1)

Each timer channel includes two (four, or six) TOUT output circuits and their signal generation and output can be controlled individually.

TOUT generation mode

The T16B*n*CCCTL*m*.TOUTMD[2:0] bits are used to set how the TOUT signal waveform is changed by the MATCH and MAX/ZERO signals.

Furthermore, when the T16B*n*CCCTL*m*.TOUTMT bit is set to 1, the TOUT circuit uses the MATCH signal output from another system in the circuit pair (0 and 1, 2 and 3, 4 and 5). This makes it possible to change the signal twice within a counter cycle.

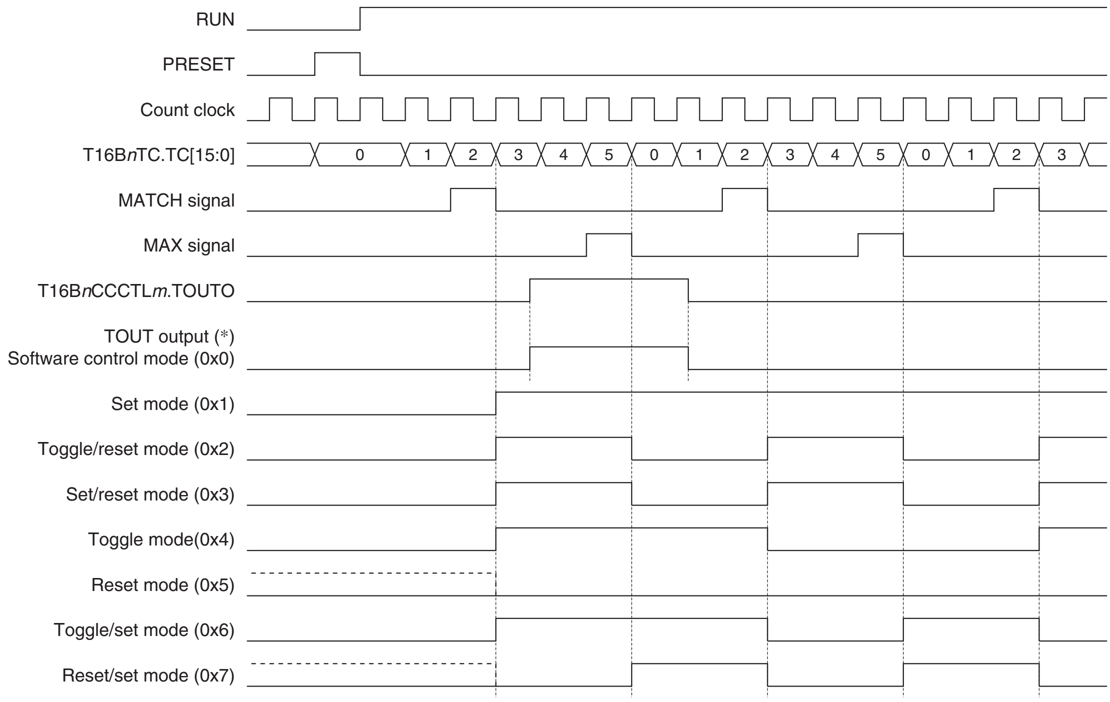
TOUT signal polarity

The TOUT signal polarity (active level) can be set using the T16B*n*CCCTL*m*.TOUTINV bit. It is set to active high by setting the T16B*n*CCCTL*m*.TOUTINV bit to 0 and active low by setting to 1.

Figures 15.4.4.2 and 15.4.4.3 show the TOUT output waveforms.

(1) Repeat up count mode

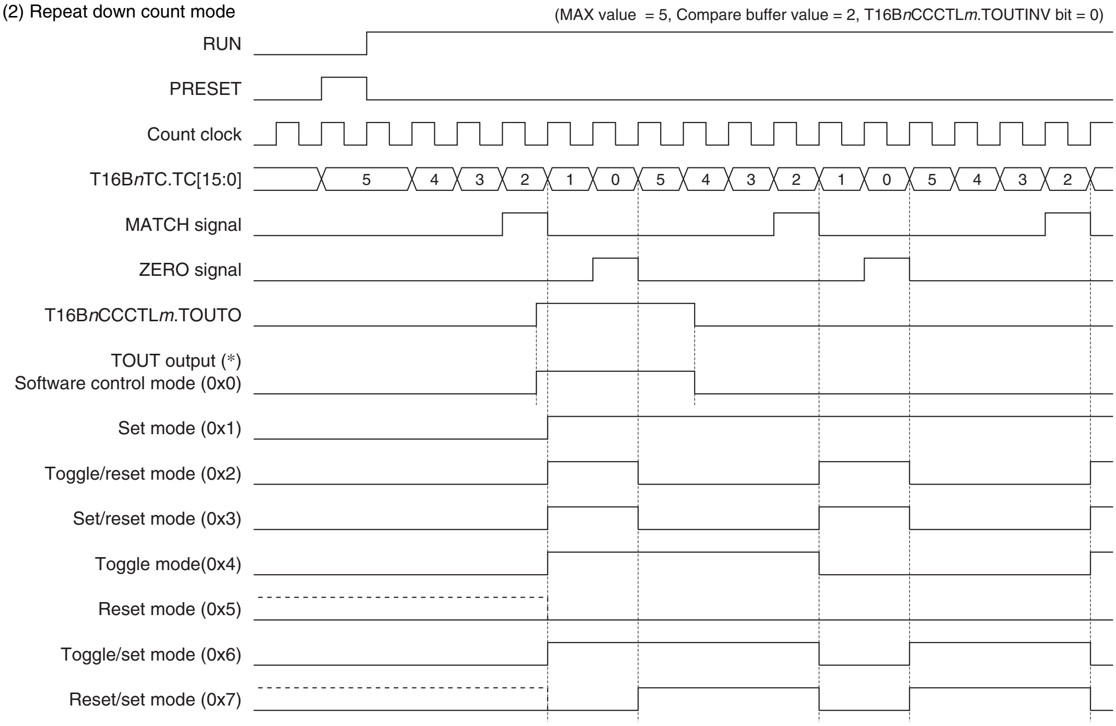
(MAX value = 5, Compare buffer value = 2, T16B*n*CCCTL*m*.TOUTINV bit = 0)



* ( ) indicates the T16B*n*CCCTL*m*.TOUTMD[2:0] bit-setting value.

(2) Repeat down count mode

(MAX value = 5, Compare buffer value = 2, T16B*n*CCCTL*m*.TOUTINV bit = 0)



* ( ) indicates the T16B*n*CCCTL*m*.TOUTMD[2:0] bit-setting value.

(3) Repeat up/down count mode

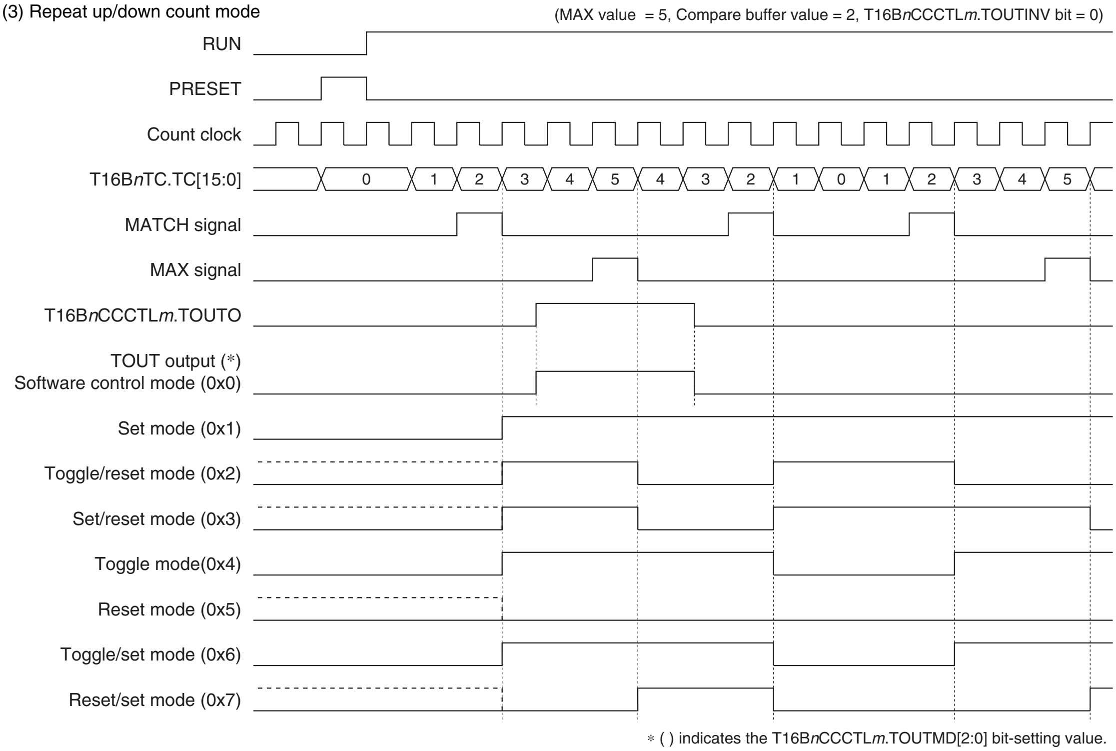
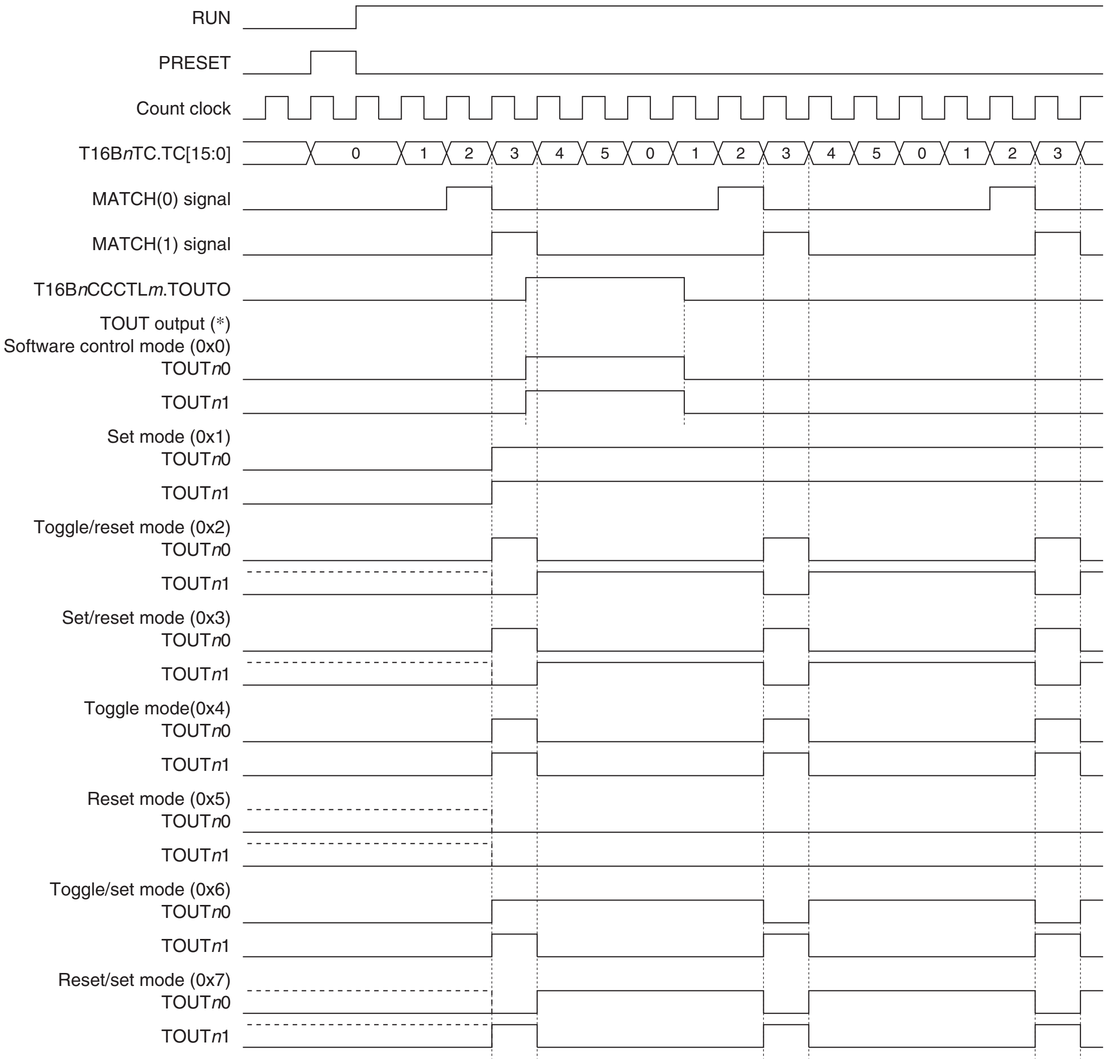


Figure 15.4.4.2 TOUT Output Waveform (T16B*n*CCCTL*m*.TOUTMT bit = 0)

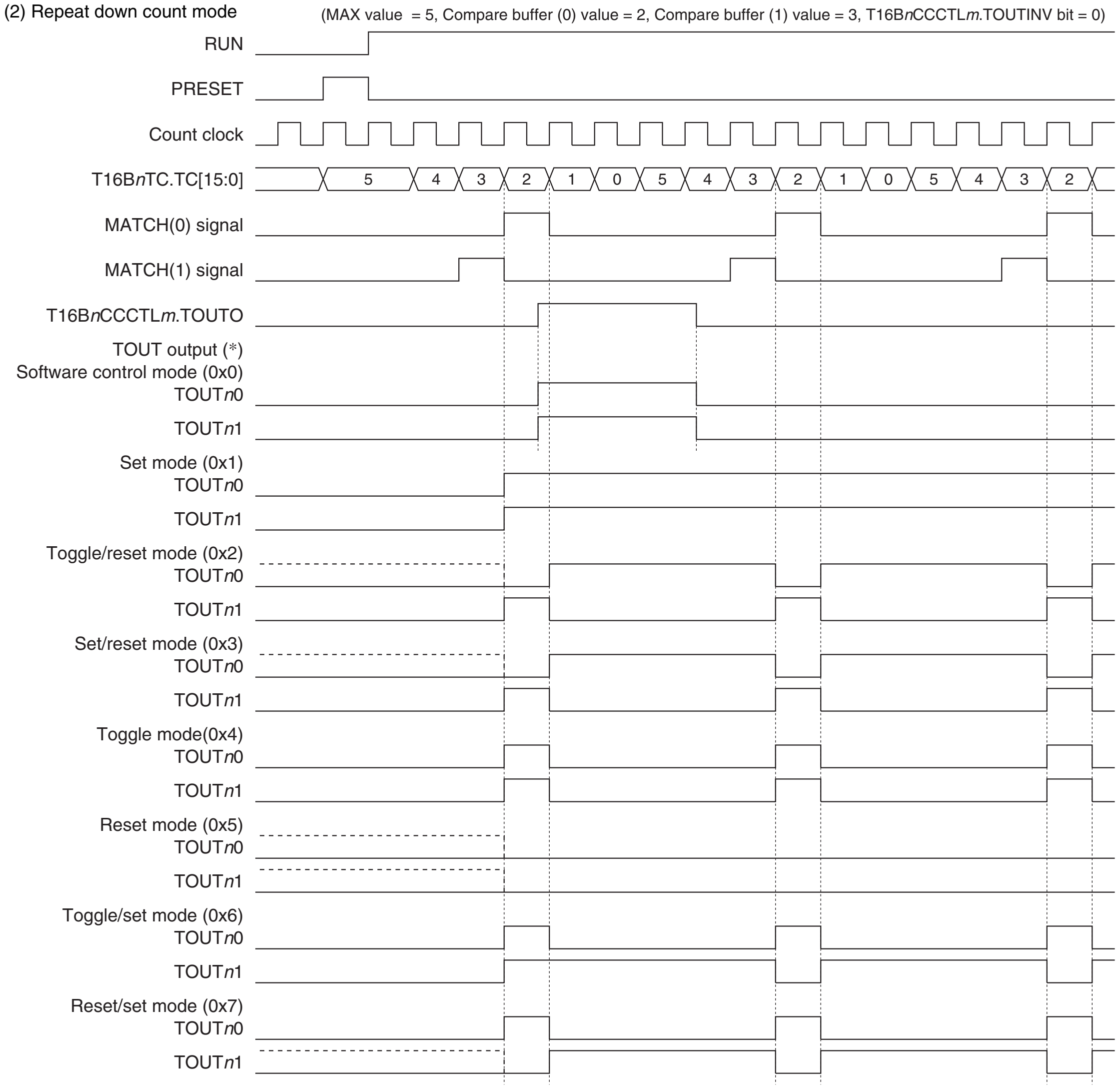
(1) Repeat up count mode

(MAX value = 5, Compare buffer (0) value = 2, Compare buffer (1) value = 3, T16B*n*CCCTL*m*.TOUTINV bit = 0)



* ( ) indicates the T16B*n*CCCTL*m*.TOUTMD[2:0] bit-setting value.

(2) Repeat down count mode



* ( ) indicates the T16BnCCCTLm.TOUTMD[2:0] bit-setting value.

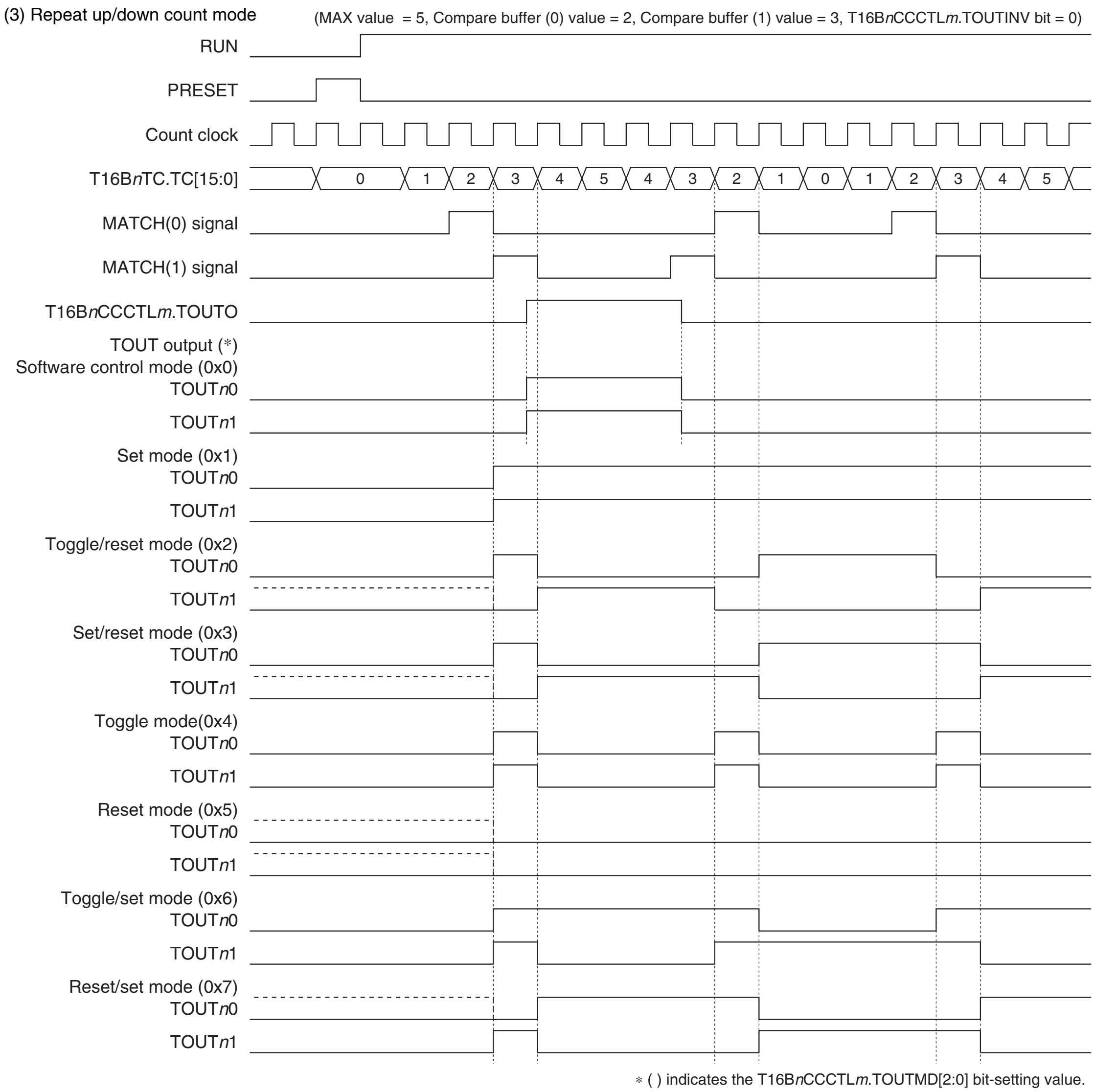


Figure 15.4.4.3 TOUT Output Waveform (T16B*n*CCCTL0.TOUTMT bit = 1, T16B*n*CCCTL1.TOUTMT bit = 0)

15.5 Interrupt

Each T16B channel has a function to generate the interrupt shown in Table 15.5.1.

Table 15.5.1 T16B Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Capture overwrite	T16B*n*INTF.CAPOW*m*IF	When the T16B*n*INTF.CMPCAP*m*IF bit =1 and the T16B*n*CCR*m* register is overwritten with new captured data in capture mode	Writing 1
Compare/ capture	T16B*n*INTF.CMPCAP*m*IF	When the counter value becomes equal to the compare buffer value in comparator mode When the counter value is loaded to the T16B*n*CCR*m* register by a capture trigger input in capture mode	Writing 1
Counter MAX	T16B*n*INTF.CNTMAXIF	When the counter reaches the MAX value	Writing 1
Counter zero	T16B*n*INTF.CNTZEROIF	When the counter reaches 0x0000	Writing 1

T16B 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.

15.6 Control Registers

T16B Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*CLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7–4	CLKDIV[3:0]	0x0	H0	R/W	
	3	–	0	–	R	
	2–0	CLKSRC[2:0]	0x0	H0	R/W	

Bits 15–9 Reserved

Bit 8 DBRUN

This bit sets whether the T16B 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 T16B Ch.*n* operating clock (counter clock).

Bit 3 Reserved

Bits 2–0 CLKSRC[2:0]

These bits select the clock source of T16B Ch.*n*.

Table 15.6.1 Clock Source and Division Ratio Settings

T16B*n*CLK.CLKDIV[3:0] bits	T16B*n*CLK.CLKSRC[2:0] bits							
	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
	IOSC	OSC1	OSC3	EXOSC	EXCL*n*0	EXCL*n*1	EXCL*n*0 inverted input	EXCL*n*1 inverted input
0xf	1/32,768	1/1	1/32,768	1/1	1/1	1/1	1/1	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) The oscillator circuits/external inputs that are not supported in this IC cannot be selected as the clock source.

T16B Ch.*n* Counter Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*CTL	15–9	–	0x00	–	R	–
	8	MAXBSY	0	H0	R	
	7–6	–	0x0	–	R	
	5–4	CNTMD[1:0]	0x0	H0	R/W	
	3	ONEST	0	H0	R/W	
	2	RUN	0	H0	R/W	
	1	PRESET	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 MAXBSY

This bit indicates whether data can be written to the T16B*n*MC register or not.

1 (R): Busy status (cannot be written)

0 (R): Idle (can be written)

While this bit is 1, the T16B*n*MC register is loading the MAX value. Data writing is prohibited during this period.

Bits 7–6 Reserved

Bits 5–4 CNTMD[1:0]

These bits select the counter up/down mode. The count mode is configured with this selection and the T16B*n*CTL.ONEST bit setting (see Table 15.6.2).

Bit 3 ONEST

This bit selects the counter repeat/one-shot mode. The count mode is configured with this selection and the T16B*n*CTL.CNTMD[1:0] bit settings (see Table 15.6.2).

Table 15.6.2 Count Mode

T16B*n*CTL.CNTMD[1:0] bits	Count mode	
	T16B*n*CTL.ONEST bit = 1	**T16B*n*CTL.ONEST bit = 0**
0x3	Reserved	
0x2	One-shot up/down count mode	Repeat up/down count mode
0x1	One-shot down count mode	Repeat down count mode
0x0	One-shot up count mode	Repeat up count mode

Bit 2 RUN

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 T16B*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 the T16B*n*CTL.RUN bit stops count operations. When the counter stops by the counter MAX/ZERO signal in one-shot mode, this bit is automatically cleared to 0.

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

In up mode or up/down mode, the counter is cleared to 0x0000 by writing 1 to this bit. In down mode, the MAX value, which has been set to the T16B*n*MC register, is preset to the counter. However, the T16B*n*CTL.MODEN bit must be set to 1 in conjunction with this bit or it must be set in advance.

Bit 0 MODEN

This bit enables the T16B Ch.*n* operations.
1 (R/W): Enable (Start supplying operating clock)
0 (R/W): Disable (Stop supplying operating clock)

Note: The counter reset operation using the T16B*n*CTL.PRESET bit and the counting start operation using the T16B*n*CTL.RUN bit take effect only when the T16B*n*CTL.MODEN bit = 1.

T16B Ch.*n* Max Counter Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*MC	15–0	MC[15:0]	0xffff	H0	R/W	–

Bits 15–0 MC[15:0]

These bits are used to set the MAX value to preset to the counter. For more information, refer to "Counter Block Operations - MAX counter data register."

Notes:
- When one-shot mode is selected, do not alter the T16B*n*MC.MC[15:0] bits (MAX value) during counting.
- Make sure the T16B*n*CTL.MODEN bit is set to 1 before writing data to the T16B*n*MC.MC[15:0] bits. If the T16B*n*CTL.MODEN bit = 0 when writing to the T16B*n*MC.MC[15:0] bits, set the T16B*n*CTL.MODEN bit to 1 until the T16B*n*CS.BSY bit is set to 0 from 1.
- Do not set the T16B*n*MC.MC[15:0] bits to 0x0000.

T16B Ch.*n* Timer Counter Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*TC	15–0	TC[15:0]	0x0000	H0	R	–

Bits 15–0 TC[15:0]

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

T16B Ch.*n* Counter Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*CS	15–8	–	0x00	–	R	–
	7	CAPI5	0	H0	R	
	6	CAPI4	0	H0	R	
	5	CAPI3	0	H0	R	
	4	CAPI2	0	H0	R	
	3	CAPI1	0	H0	R	
	2	CAPI0	0	H0	R	
	1	UP_DOWN	1	H0	R	
	0	BSY	0	H0	R	

Bits 15–8 Reserved

Bit 7 CAPI5
Bit 6 CAPI4
Bit 5 CAPI3
Bit 4 CAPI2
Bit 3 CAPI1
Bit 2 CAPI0

These bits indicate the signal level currently input to the CAP*nm* pin.

1 (R): Input signal = High level
0 (R): Input signal = Low level

The following shows the correspondence between the bit and the CAP*nm* pin:
T16B*n*CS.CAPI5 bit: CAP*n*5 pin
T16B*n*CS.CAPI4 bit: CAP*n*4 pin
T16B*n*CS.CAPI3 bit: CAP*n*3 pin
T16B*n*CS.CAPI2 bit: CAP*n*2 pin
T16B*n*CS.CAPI1 bit: CAP*n*1 pin
T16B*n*CS.CAPI0 bit: CAP*n*0 pin

Note: The configuration of the T16B*n*CS.CAPI*m* bits depends on the model. The bits corresponding to the CAP*nm* pins that do not exist are read-only bits and are always fixed at 0.

Bit 1 UP_DOWN

This bit indicates the currently set count direction.

1 (R): Count up
0 (R): Count down

Bit 0 BSY

This bit indicates the counter operating status.

1 (R): Running
0 (R): Idle

T16B Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*INTF	15–14	–	0x0	–	R	–
	13	CAPOW5IF	0	H0	R/W	Cleared by writing 1.
	12	CMPCAP5IF	0	H0	R/W	
	11	CAPOW4IF	0	H0	R/W	
	10	CMPCAP4IF	0	H0	R/W	
	9	CAPOW3IF	0	H0	R/W	
	8	CMPCAP3IF	0	H0	R/W	
	7	CAPOW2IF	0	H0	R/W	
	6	CMPCAP2IF	0	H0	R/W	
	5	CAPOW1IF	0	H0	R/W	
	4	CMPCAP1IF	0	H0	R/W	
	3	CAPOW0IF	0	H0	R/W	
	2	CMPCAP0IF	0	H0	R/W	
	1	CNTMAXIF	0	H0	R/W	
	0	CNTZEROIF	0	H0	R/W	

Bits 15–14 Reserved

Bit 13 CAPOW5IF
Bit 12 CMPCAP5IF
Bit 11 CAPOW4IF
Bit 10 CMPCAP4IF
Bit 9 CAPOW3IF
Bit 8 CMPCAP3IF
Bit 7 CAPOW2IF
Bit 6 CMPCAP2IF
Bit 5 CAPOW1IF
Bit 4 CMPCAP1IF
Bit 3 CAPOW0IF
Bit 2 CMPCAP0IF
Bit 1 CNTMAXIF
Bit 0 CNTZEROIF

These bits indicate the T16B 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:

T16B*n*INTF.CAPOW5IF bit: Capture 5 overwrite interrupt
T16B*n*INTF.CMPCAP5IF bit: Compare/capture 5 interrupt
T16B*n*INTF.CAPOW4IF bit: Capture 4 overwrite interrupt
T16B*n*INTF.CMPCAP4IF bit: Compare/capture 4 interrupt
T16B*n*INTF.CAPOW3IF bit: Capture 3 overwrite interrupt
T16B*n*INTF.CMPCAP3IF bit: Compare/capture 3 interrupt
T16B*n*INTF.CAPOW2IF bit: Capture 2 overwrite interrupt
T16B*n*INTF.CMPCAP2IF bit: Compare/capture 2 interrupt
T16B*n*INTF.CAPOW1IF bit: Capture 1 overwrite interrupt
T16B*n*INTF.CMPCAP1IF bit: Compare/capture 1 interrupt
T16B*n*INTF.CAPOW0IF bit: Capture 0 overwrite interrupt
T16B*n*INTF.CMPCAP0IF bit: Compare/capture 0 interrupt
T16B*n*INTF.CNTMAXIF bit: Counter MAX interrupt
T16B*n*INTF.CNTZEROIF bit: Counter zero interrupt

Note: The configuration of the T16B*n*INTF.CAPOW*m*IF and T16B*n*INTF.CMPCAP*m*IF bits depends on the model. The bits corresponding to the comparator/capture circuits that do not exist are read-only bits and are always fixed at 0.

T16B Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*INTE	15–14	–	0x0	–	R	–
	13	CAPOW5IE	0	H0	R/W	
	12	CMPCAP5IE	0	H0	R/W	
	11	CAPOW4IE	0	H0	R/W	
	10	CMPCAP4IE	0	H0	R/W	
	9	CAPOW3IE	0	H0	R/W	
	8	CMPCAP3IE	0	H0	R/W	
	7	CAPOW2IE	0	H0	R/W	
	6	CMPCAP2IE	0	H0	R/W	
	5	CAPOW1IE	0	H0	R/W	
	4	CMPCAP1IE	0	H0	R/W	
	3	CAPOW0IE	0	H0	R/W	
	2	CMPCAP0IE	0	H0	R/W	
	1	CNTMAXIE	0	H0	R/W	
	0	CNTZEROIE	0	H0	R/W	

Bits 15–14 Reserved

Bit 13 CAPOW5IE
Bit 12 CMPCAP5IE
Bit 11 CAPOW4IE
Bit 10 CMPCAP4IE
Bit 9 CAPOW3IE
Bit 8 CMPCAP3IE
Bit 7 CAPOW2IE
Bit 6 CMPCAP2IE
Bit 5 CAPOW1IE
Bit 4 CMPCAP1IE
Bit 3 CAPOW0IE
Bit 2 CMPCAP0IE
Bit 1 CNTMAXIE
Bit 0 CNTZEROIE

These bits enable T16B Ch.*n* interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
T16B*n*INTE.CAPOW5IE bit: Capture 5 overwrite interrupt
T16B*n*INTE.CMPCAP5IE bit: Compare/capture 5 interrupt
T16B*n*INTE.CAPOW4IE bit: Capture 4 overwrite interrupt
T16B*n*INTE.CMPCAP4IE bit: Compare/capture 4 interrupt
T16B*n*INTE.CAPOW3IE bit: Capture 3 overwrite interrupt
T16B*n*INTE.CMPCAP3IE bit: Compare/capture 3 interrupt
T16B*n*INTE.CAPOW2IE bit: Capture 2 overwrite interrupt
T16B*n*INTE.CMPCAP2IE bit: Compare/capture 2 interrupt
T16B*n*INTE.CAPOW1IE bit: Capture 1 overwrite interrupt
T16B*n*INTE.CMPCAP1IE bit: Compare/capture 1 interrupt
T16B*n*INTE.CAPOW0IE bit: Capture 0 overwrite interrupt
T16B*n*INTE.CMPCAP0IE bit: Compare/capture 0 interrupt
T16B*n*INTE.CNTMAXIE bit: Counter MAX interrupt
T16B*n*INTE.CNTZEROIE bit: Counter zero interrupt

Notes:
- The configuration of the T16B*n*INTE.CAPOW*m*IE and T16B*n*INTE.CMPCAP*m*IE bits depends on the model. The bits corresponding to the comparator/capture circuits that do not exist are read-only bits and are always fixed at 0.
- To prevent generating unnecessary interrupts, the corresponding interrupt flag should be cleared before enabling interrupts.

T16B Ch.*n* Comparator/Capture *m* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*CCCTL*m*	15	SCS	0	H0	R/W	–
	14–12	CBUFMD[2:0]	0x0	H0	R/W	
	11–10	CAPIS[1:0]	0x0	H0	R/W	
	9–8	CAPTRG[1:0]	0x0	H0	R/W	
	7	–	0	–	R	
	6	TOUTMT	0	H0	R/W	
	5	TOUTO	0	H0	R/W	
	4–2	TOUTMD[2:0]	0x0	H0	R/W	
	1	TOUTINV	0	H0	R/W	
	0	CCMD	0	H0	R/W	

Bit 15 SCS

This bit selects either synchronous capture mode or asynchronous capture mode.

1 (R/W): Synchronous capture mode

0 (R/W): Asynchronous capture mode

For more information, refer to “Comparator/Capture Block Operations - Synchronous capture mode/asynchronous capture mode.” The T16B*n*CCCTL*m*.SCS bit is control bit for capture mode and is ineffective in comparator mode.

Bits 14–12 CBUFMD[2:0]

These bits select the timing to load the comparison value written in the T16B*n*CCR*m* register to the compare buffer. The T16B*n*CCCTL*m*.CBUFMD[2:0] bits are control bits for comparator mode and are ineffective in capture mode.

Table 15.6.3 Timings to Load Comparison Value to Compare Buffer

T16B*n*CCCTL*m*.CBUFMD[2:0] bits	Count mode	Comparison Value load timing
0x7–0x5	Reserved	
0x4	Up mode	When the counter becomes equal to the comparison value set previously Also the counter is reset to 0x0000 simultaneously.
	Down mode	When the counter becomes equal to the comparison value set previously Also the counter is reset to the MAX value simultaneously.
	Up/down mode	When the counter becomes equal to the comparison value set previously Also the counter is reset to 0x0000 simultaneously.
0x3	Up mode	When the counter reverts to 0x0000
	Down mode	When the counter reverts to the MAX value
	Up/down mode	When the counter becomes equal to the comparison value set previously or when the counter reverts to 0x0000
0x2	Up mode	When the counter becomes equal to the comparison value set previously
	Down mode	
	Up/down mode	
0x1	Up mode	When the counter reaches the MAX value
	Down mode	When the counter reaches 0x0000
	Up/down mode	When the counter reaches 0x0000 or the MAX value
0x0	Up mode	At the CLK_16B*n* rising edge after writing to the T16B*n*CCR*m* register
	Down mode	
	Up/down mode	

Bits 11–10 CAPIS[1:0]

These bits select the trigger signal for capturing (see Table 15.6.4). The T16B*n*CCCTL*m*.CAPIS[1:0] bits are control bits for capture mode and are ineffective in comparator mode.

Bits 9–8 CAPTRG[1:0]

These bits select the trigger edge(s) of the trigger signal at which the counter value is captured in the T16B*n*CCR*m* register in capture mode (see Table 15.6.4). The T16B*n*CCCTL*m*.CAPTRG[1:0] bits are control bits for capture mode and are ineffective in comparator mode.

Table 15.6.4 Trigger Signal/Edge for Capturing Counter Value

T16B*n*CCCTL*m*. CAPTRG[1:0] bits (Trigger edge)	**Trigger condition**		
	T16B*n*CCCTL*m*.CAPIS[1:0] bits (Trigger signal)		
	0x0 (External trigger signal)	0x2 (Software trigger signal = L)	0x3 (Software trigger signal = H)
0x3 (↑ & ↓)	Rising or falling edge of the CAP*nm* pin input signal	Altering the T16B*n*CCCTL*m*.CAPIS[1:0] bits from 0x2 to 0x3, or from 0x3 to 0x2	
0x2 (↓)	Falling edge of the CAP*nm* pin input signal	Altering the T16B*n*CCCTL*m*.CAPIS[1:0] bits from 0x3 to 0x2	
0x1 (↑)	Rising edge of the CAP*nm* pin input signal	Altering the T16B*n*CCCTL*m*.CAPIS[1:0] bits from 0x2 to 0x3	
0x0	Not triggered (disable capture function)		

Bit 7 Reserved

Bit 6 TOUTMT

This bit selects whether the comparator MATCH signal of another system is used for generating the TOUT*nm* signal or not.

1 (R/W): Generate TOUT using two comparator MATCH signals of the comparator circuit pair (0 and 1, 2 and 3, 4 and 5)

0 (R/W): Generate TOUT using one comparator MATCH signal of comparator *m* and the counter MAX or ZERO signals

The T16B*n*CCCTL*m*.TOUTMT bit is control bit for comparator mode and is ineffective in capture mode.

Bit 5 TOUTO

This bit sets the TOUT*nm* signal output level when software control mode (T16B*n*CCCTL*m*.TOUTMD[2:0] = 0x0) is selected for the TOUT*nm* output.

1 (R/W): High level output

0 (R/W): Low level output

The T16B*n*CCCTL*m*.TOUTO bit is control bit for comparator mode and is ineffective in capture mode.

Bits 4–2 TOUTMD[2:0]

These bits configure how the TOUT*nm* signal waveform is changed by the comparator MATCH and counter MAX/ZERO signals.

The T16B*n*CCCTL*m*.TOUTMD[2:0] bits are control bits for comparator mode and are ineffective in capture mode.

Table 15.6.5 TOUT Generation Mode

T16B*n*CCCTL*m*. TOUTMD[2:0] bits	**TOUT generation mode and operations**			
	T16B*n*CCCTL*m*. TOUTMT bit	**Count mode**	**Output signal**	**Change in the signal**
0x7	**Reset/set mode**			
	0	Up count mode Up/down count mode	TOUT*nm*	The signal becomes inactive by the MATCH signal and it becomes active by the MAX signal.
		Down count mode	TOUT*nm*	The signal becomes inactive by the MATCH signal and it becomes active by the ZERO signal.
	1	All count modes	TOUT*nm*	The signal becomes inactive by the MATCH*m* signal and it becomes active by the MATCH*m*+1 signal.
			TOUT*nm*+1	The signal becomes inactive by the MATCH*m*+1 signal and it becomes active by the MATCH*m* signal.
0x6	**Toggle/set mode**			
	0	Up count mode Up/down count mode	TOUT*nm*	The signal is inverted by the MATCH signal and it becomes active by the MAX signal.
		Down count mode	TOUT*nm*	The signal is inverted by the MATCH signal and it becomes active by the ZERO signal.
	1	All count modes	TOUT*nm*	The signal is inverted by the MATCH*m* signal and it becomes active by the MATCH*m*+1 signal.
			TOUT*nm*+1	The signal is inverted by the MATCH*m*+1 signal and it becomes active by the MATCH*m* signal.
0x5	**Reset mode**			
	0	All count modes	TOUT*nm*	The signal becomes inactive by the MATCH signal.
	1	All count modes	TOUT*nm*	The signal becomes inactive by the MATCH*m* or MATCH*m*+1 signal.
			TOUT*nm*+1	The signal becomes inactive by the MATCH*m*+1 or MATCH*m* signal.

T16B*n*CCCTL*m*.TOUTMD[2:0] bits	TOUT generation mode and operations			
	T16B*n*CCCTL*m*.TOUTMT bit	Count mode	Output signal	Change in the signal
0x4	**Toggle mode**			
	0	All count modes	TOUT*nm*	The signal is inverted by the MATCH signal.
	1	All count modes	TOUT*nm*	The signal is inverted by the MATCH*m* or MATCH*m*+1 signal.
			TOUT*nm*+1	The signal is inverted by the MATCH*m*+1 or MATCH*m* signal.
0x3	**Set/reset mode**			
	0	Up count mode Up/down count mode	TOUT*nm*	The signal becomes active by the MATCH signal and it becomes inactive by the MAX signal.
		Down count mode	TOUT*nm*	The signal becomes active by the MATCH signal and it becomes inactive by the ZERO signal.
	1	All count modes	TOUT*nm*	The signal becomes active by the MATCH*m* signal and it becomes inactive by the MATCH*m*+1 signal.
			TOUT*nm*+1	The signal becomes active by the MATCH*m*+1 signal and it becomes inactive by the MATCH*m* signal.
0x2	**Toggle/reset mode**			
	0	Up count mode Up/down count mode	TOUT*nm*	The signal is inverted by the MATCH signal and it becomes inactive by the MAX signal.
		Down count mode	TOUT*nm*	The signal is inverted by the MATCH signal and it becomes inactive by the ZERO signal.
	1	All count modes	TOUT*nm*	The signal is inverted by the MATCH*m* signal and it becomes inactive by the MATCH*m*+1 signal.
			TOUT*nm*+1	The signal is inverted by the MATCH*m*+1 signal and it becomes inactive by the MATCH*m* signal.
0x1	**Set mode**			
	0	All count modes	TOUT*nm*	The signal becomes active by the MATCH signal.
	1	All count modes	TOUT*nm*	The signal becomes active by the MATCH*m* or MATCH*m*+1 signal.
			TOUT*nm*+1	The signal becomes active by the MATCH*m*+1 or MATCH*m* signal.
0x0	**Software control mode**			
	*	All count modes	TOUT*nm*	The signal becomes active by setting the T16B*n*CCCTL*m*.TOUTO bit to 1 and it becomes inactive by setting to 0.

Bit 1 TOUTINV

This bit selects the TOUT*nm* signal polarity.

1 (R/W): Inverted (active low)

0 (R/W): Normal (active high)

The T16B*n*CCCTL*m*.TOUTINV bit is control bit for comparator mode and is ineffective in capture mode.

Bit 0 CCMD

This bit selects the operating mode of the comparator/capture circuit *m*.

1 (R/W): Capture mode (T16*n*CCR*m* register = capture register)

0 (R/W): Comparator mode (T16*n*CCR*m* register = compare data register)

T16B Ch.*n* Compare/Capture *m* Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B*n*CCR*m*	15–0	CC[15:0]	0x0000	H0	R/W	–

Bits 15–0 CC[15:0]

In comparator mode, this register is configured as the compare data register and used to set the comparison value to be compared with the counter value.

In capture mode, this register is configured as the capture register and the counter value captured by the capture trigger signal is loaded.

16 Sound Generator (SNDA)

16.1 Overview

SNDA is a sound generator that generates melodies and buzzer signals. The features of the SNDA are listed below.

- Sound output mode is selectable from three types.
  1. Normal buzzer mode (for normal buzzer output of which the output duration is controlled via software)
     - Output frequency: Can be set within the range of 512 Hz to 16,384 Hz.
     - Duty ratio: Can be set within the range of 0 % to 100 %.
  2. One-shot buzzer mode (for short buzzer output such as a clicking sound)
     - Output frequency: Can be set within the range of 512 Hz to 16,384 Hz.
     - Duty ratio: Can be set within the range of 0 % to 100 %.
     - One-shot output duration: Can be set within the range of 15.6 ms to 250 ms. (16 types)
  3. Melody mode (for playing single note melody)
     - Pitch: Can be set within the range of 128 Hz to 16,384 Hz. (Scale: 3 octave from C3 to C6 with reference to A4 = 443 Hz)
     - Duration: Can be set within the range of half note/rest to thirty-second note/rest. (7 types)
     - Tempo: Can be set within the range of 30 to 480. (16 types)
     - Other: Tie can be specified (slur is not supported).
- A piezoelectric buzzer can be driven with the inverted and non-inverted output pins.
- Can control the non-inverted output pin status while sound stops.

Figure 16.1.1 shows the SNDA configuration.

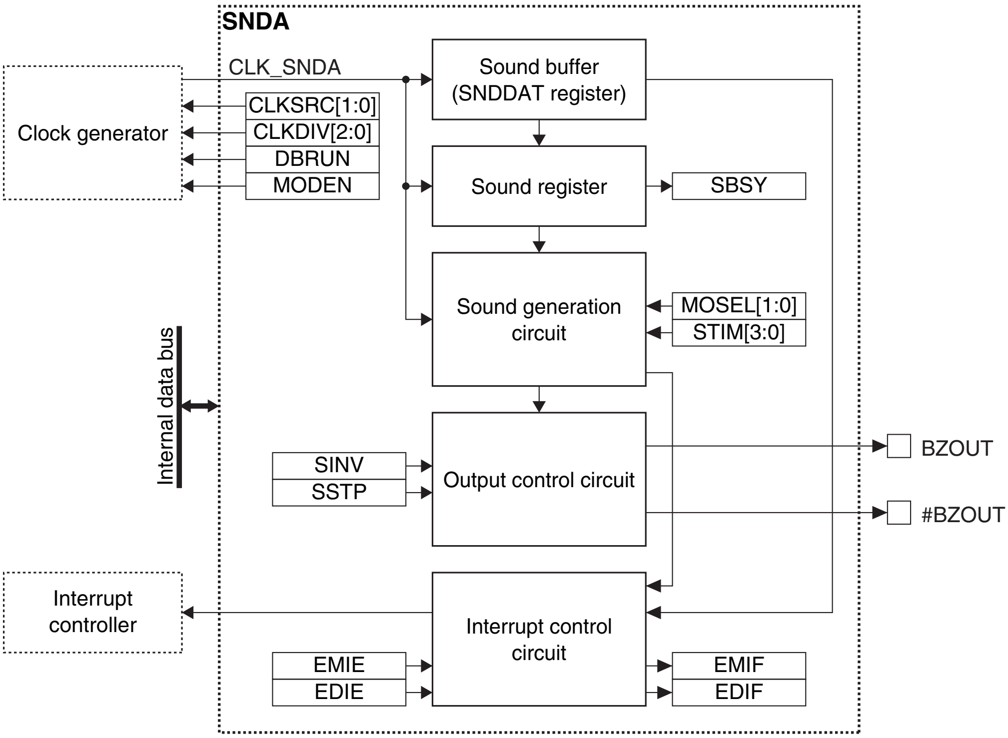


Figure 16.1.1 SNDA Configuration

16.2 Output Pins and External Connections

16.2.1 List of Output Pins

Table 16.2.1.1 lists the SNDA pins.

Table 16.2.1.1 List of SNDA Pins

Pin name	I/O*	Initial status*	Function
BZOUT	O	O (Low)	Non-inverted buzzer output pin
#BZOUT	O	O (Low)	Inverted buzzer output pin

* Indicates the status when the pin is configured for SNDA

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

16.2.2 Output Pin Drive Mode

The drive mode of the BZOUT and #BZOUT pins can be set to one of the two types shown below using the SNDSEL.SINV bit.

Direct drive mode (SNDSEL.SINV bit = 0)

This mode drives both the BZOUT and #BZOUT pins to low while the buzzer signal output is off to prevent the piezoelectric buzzer from applying unnecessary bias.

Normal drive mode (SNDSEL.SINV bit = 1)

In this mode, the #BZOUT pin always outputs the inverted signal of the BZOUT pin even when the buzzer output is off.

16.2.3 External Connections

Figures 16.2.2.1 and 16.2.2.2 show connection diagrams between SNDA and a piezoelectric buzzer.

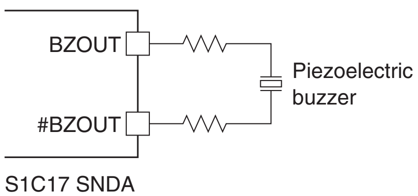


Figure 16.2.2.1 Connection between SNDA and Piezoelectric Buzzer (Direct Drive)

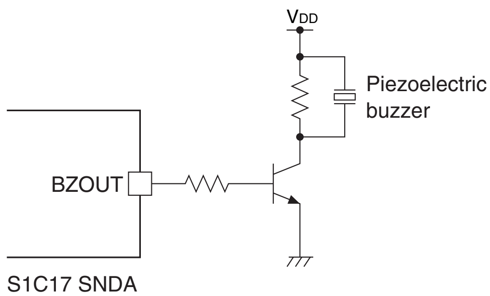


Figure 16.2.2.2 Connection between SNDA and Piezoelectric Buzzer (Single Pin Drive)

16.3 Clock Settings

16.3.1 SNDA Operating Clock

When using SNDA, the SNDA operating clock CLK_SNDA must be supplied to SNDA from the clock generator. The CLK_SNDA 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 SNDCLK register bits:
   - SNDCLK.CLKSRC[1:0] bits (Clock source selection)
   - SNDCLK.CLKDIV[2:0] bits (Clock division ratio selection = Clock frequency setting)

The CLK_SNDA frequency should be set to around 32,768 Hz.

16.3.2 Clock Supply in SLEEP Mode

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

16.3.3 Clock Supply in DEBUG Mode

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

16.4 Operations

16.4.1 Initialization

SNDA should be initialized with the procedure shown below.

1. Assign the SNDA output function to the ports. (Refer to the "I/O Ports" chapter.)
2. Configure the SNDA operating clock.
3. Set the SNDCTL.MODEN bit to 1. (Enable SNDA operations)
4. Set the SNDSEL.SINV bit. (Set output pin drive mode)
5. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the SNDINTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the SNDINTE register to 1. (Enable interrupts)

16.4.2 Buzzer Output in Normal Buzzer Mode

Normal buzzer mode generates a buzzer signal with the software specified frequency and duty ratio, and outputs the generated signal to outside the IC. The buzzer output duration can also be controlled via software.
An output start/stop procedure and the SNDA operations are shown below.

Normal buzzer output start/stop procedure

1. Set the SNDSEL.MOSEL[1:0] bits to 0x0. (Set normal buzzer mode)

2. Write data to the following sound buffer (SNDDAT register) bits. (Start buzzer output)
   - SNDDAT.SLEN[5:0] bits (Set buzzer output signal duty ratio)
   - SNDDAT.SFRQ[7:0] bits (Set buzzer output signal frequency)
3. Write 1 to the SNDCTL.SSTP bit after the output period has elapsed. (Stop buzzer output)

Normal buzzer output operations

When data is written to the sound buffer (SNDDAT register), SNDA clears the SNDINTF.EMIF bit (sound buffer empty interrupt flag) to 0 and starts buzzer output operations.

The data written to the sound buffer is loaded into the sound register in sync with the CLK_SNDA clock. At the same time, the SNDINTF.EMIF bit and SNDINTF.SBSY bit are both set to 1. The output pin outputs the buzzer signal with the frequency/duty ratio specified.

Writing 1 to the SNDCTL.SSTP bit stops buzzer output and sets the SNDINTF.EDIF bit (sound output completion interrupt flag) to 1. The SNDINTF.SBSY bit is cleared to 0.

Figure 16.4.2.1 shows a buzzer output timing chart in normal buzzer mode.

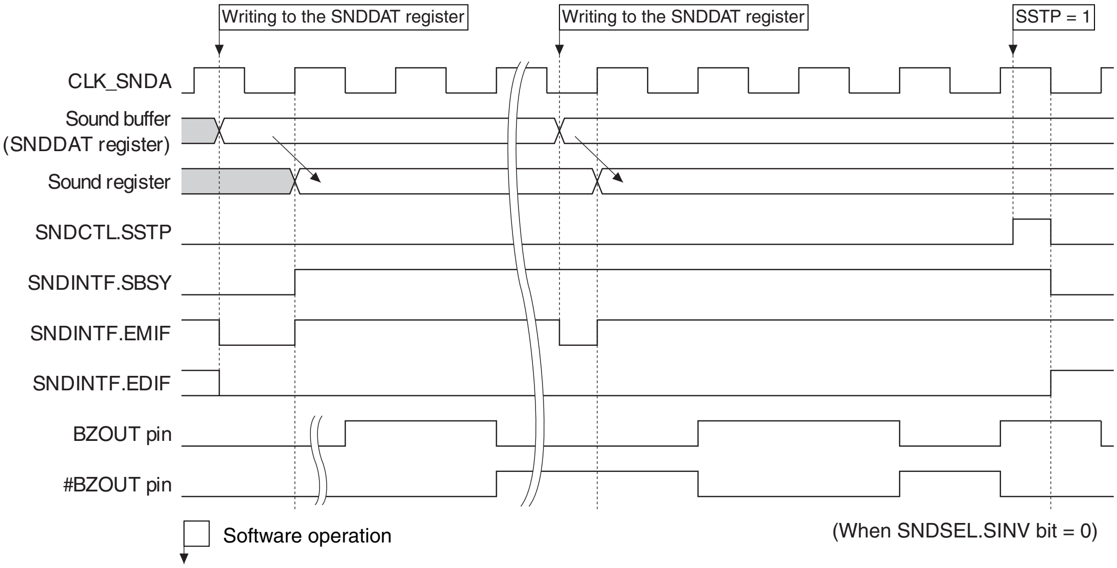


Figure 16.4.2.1 Buzzer Output Timing Chart in Normal Buzzer Mode

Buzzer output waveform configuration (normal buzzer mode/one-shot buzzer mode)

Set the buzzer signal frequency and duty ratio (high period/cycle) using the SNDDAT.SFRQ[7:0] and SNDDAT.SLEN[5:0] bits, respectively. Use the following equations to calculate these setting values.

$$\text{SNDDAT.SFRQ[7:0] bits} = \frac{f_{CLK\_SNDA}}{f_{BZOUT}} - 1 \qquad \text{(Eq. 16.1)}$$

$$\text{SNDDAT.SLEN[5:0] bits} = \left(\frac{f_{CLK\_SNDA}}{f_{BZOUT}} \times \frac{DUTY}{100}\right) - 1 \qquad \text{(Eq. 16.2)}$$

Where

f_{CLK_SNDA}: CLK_SNDA frequency [Hz]
f_{BZOUT}: Buzzer signal frequency [Hz]
DUTY: Buzzer signal duty ratio [%]

However, the following settings are prohibited:

- Settings as SNDDAT.SFRQ[7:0] bits ≤ SNDDAT.SLEN[5:0] bits
- Settings as SNDDAT.SFRQ[7:0] bits = 0x00

Table 16.4.2.1 Buzzer Frequency Settings (when f_{CLK_SNDA} = 32,768 Hz)

SNDDAT. SFRQ[7:0] bits	Frequency [Hz]	SNDDAT. SFRQ[7:0] bits	Frequency [Hz]	SNDDAT. SFRQ[7:0] bits	Frequency [Hz]	SNDDAT. SFRQ[7:0] bits	Frequency [Hz]
0x3f	512.0	0x2f	682.7	0x1f	1,024.0	0x0f	2,048.0
0x3e	520.1	0x2e	697.2	0x1e	1,057.0	0x0e	2,184.5
0x3d	528.5	0x2d	712.3	0x1d	1,092.3	0x0d	2,340.6
0x3c	537.2	0x2c	728.2	0x1c	1,129.9	0x0c	2,520.6
0x3b	546.1	0x2b	744.7	0x1b	1,170.3	0x0b	2,730.7
0x3a	555.4	0x2a	762.0	0x1a	1,213.6	0x0a	2,978.9
0x39	565.0	0x29	780.2	0x19	1,260.3	0x09	3,276.8
0x38	574.9	0x28	799.2	0x18	1,310.7	0x08	3,640.9
0x37	585.1	0x27	819.2	0x17	1,365.3	0x07	4,096.0
0x36	595.8	0x26	840.2	0x16	1,424.7	0x06	4,681.1
0x35	606.8	0x25	862.3	0x15	1,489.5	0x05	5,461.3
0x34	618.3	0x24	885.6	0x14	1,560.4	0x04	6,553.6
0x33	630.2	0x23	910.2	0x13	1,638.4	0x03	8,192.0
0x32	642.5	0x22	936.2	0x12	1,724.6	0x02	10,922.7
0x31	655.4	0x21	963.8	0x11	1,820.4	0x01	16,384.0
0x30	668.7	0x20	993.0	0x10	1,927.5	0x00	Cannot be set

Table 16.4.2.2 Buzzer Duty Ratio Setting Examples (when f_{CLK_SNDA} = 32,768 Hz)

SNDDAT. SLEN[5:0] bits	Duty ratio by buzzer frequency					
	16,384 Hz	8,192 Hz	4,096 Hz	2,048 Hz	1,024 Hz	512 Hz
0x3f	–	–	–	–	–	–
0x3e	–	–	–	–	–	98.4
0x3d	–	–	–	–	–	96.9
0x3c	–	–	–	–	–	95.3
0x3b	–	–	–	–	–	93.8
0x3a	–	–	–	–	–	92.2
0x39	–	–	–	–	–	90.6
0x38	–	–	–	–	–	89.1
0x37	–	–	–	–	–	87.5
0x36	–	–	–	–	–	85.9
0x35	–	–	–	–	–	84.4
0x34	–	–	–	–	–	82.8
0x33	–	–	–	–	–	81.3
0x32	–	–	–	–	–	79.7
0x31	–	–	–	–	–	78.1
0x30	–	–	–	–	–	76.6
0x2f	–	–	–	–	–	75.0
0x2e	–	–	–	–	–	73.4
0x2d	–	–	–	–	–	71.9
0x2c	–	–	–	–	–	70.3
0x2b	–	–	–	–	–	68.8
0x2a	–	–	–	–	–	67.2
0x29	–	–	–	–	–	65.6
0x28	–	–	–	–	–	64.1
0x27	–	–	–	–	–	62.5
0x26	–	–	–	–	–	60.9
0x25	–	–	–	–	–	59.4
0x24	–	–	–	–	–	57.8
0x23	–	–	–	–	–	56.3
0x22	–	–	–	–	–	54.7
0x21	–	–	–	–	–	53.1
0x20	–	–	–	–	–	51.6
0x1f	–	–	–	–	–	50.0
0x1e	–	–	–	–	96.9	48.4
0x1d	–	–	–	–	93.8	46.9
0x1c	–	–	–	–	90.6	45.3
0x1b	–	–	–	–	87.5	43.8
0x1a	–	–	–	–	84.4	42.2
0x19	–	–	–	–	81.3	40.6
0x18	–	–	–	–	78.1	39.1
0x17	–	–	–	–	75.0	37.5
0x16	–	–	–	–	71.9	35.9
0x15	–	–	–	–	68.8	34.4
0x14	–	–	–	–	65.6	32.8
0x13	–	–	–	–	62.5	31.3
0x12	–	–	–	–	59.4	29.7

SNDDAT. SLEN[5:0] bits	Duty ratio by buzzer frequency					
	16,384 Hz	8,192 Hz	4,096 Hz	2,048 Hz	1,024 Hz	512 Hz
0x11	–	–	–	–	56.3	28.1
0x10	–	–	–	–	53.1	26.6
0x0f	–	–	–	–	50.0	25.0
0x0e	–	–	–	93.8	46.9	23.4
0x0d	–	–	–	87.5	43.8	21.9
0x0c	–	–	–	81.3	40.6	20.3
0x0b	–	–	–	75.0	37.5	18.8
0x0a	–	–	–	68.8	34.4	17.2
0x09	–	–	–	62.5	31.3	15.6
0x08	–	–	–	56.3	28.1	14.1
0x07	–	–	–	50.0	25.0	12.5
0x06	–	–	87.5	43.8	21.9	10.9
0x05	–	–	75.0	37.5	18.8	9.4
0x04	–	–	62.5	31.3	15.6	7.8
0x03	–	–	50.0	25.0	12.5	6.3
0x02	–	75.0	37.5	18.8	9.4	4.7
0x01	–	50.0	25.0	12.5	6.3	3.1
0x00	50.0	25.0	12.5	6.3	3.1	1.6

16.4.3 Buzzer Output in One-shot Buzzer Mode

One-shot buzzer mode is provided for clicking sound and short-duration buzzer output. This mode generates a buzzer signal with the software specified frequency and duty ratio, and outputs the generated signal for the short duration specified.

An output start procedure and the SNDA operations are shown below. For the buzzer output waveform, refer to “Buzzer Output in Normal Buzzer Mode.”

One-shot buzzer output start procedure

1. Set the following SNDSEL register bits:
   - Set the SNDSEL.MOSEL[1:0] bits to 0x1. (Set one-shot buzzer mode)
   - SNDSEL.STIM[3:0] bits (Set output duration)
2. Write data to the following sound buffer (SNDDAT register) bits. (Start buzzer output)
   - SNDDAT.SLEN[5:0] bits (Set buzzer output signal duty ratio)
   - SNDDAT.SFRQ[7:0] bits (Set buzzer output signal frequency)

One-shot buzzer output operations

When data is written to the sound buffer (SNDDAT register), SNDA clears the SNDINTF.EMIF bit (sound buffer empty interrupt flag) to 0 and starts buzzer output operations.

The data written to the sound buffer is loaded into the sound register in sync with the CLK_SNDA clock. At the same time, the SNDINTF.EMIF bit and SNDINTF.SBSY bit are both set to 1. The output pin outputs the buzzer signal with the frequency/duty ratio specified.

The buzzer output automatically stops when the duration specified by the SNDSEL.STIM[3:0] bits has elapsed. At the same time, the SNDINTF.EDIF bit (sound output completion interrupt flag) is set to 1 and the SNDINTF.SBSY bit is cleared to 0.

Figure 16.4.3.1 shows a buzzer output timing chart in one-shot buzzer mode.

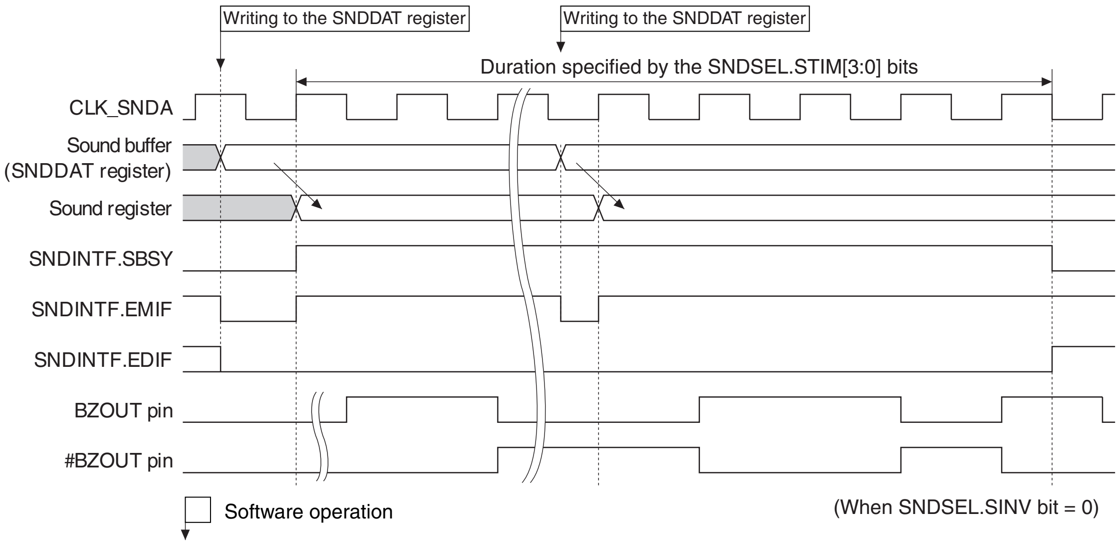


Figure 16.4.3.1 Buzzer Output Timing Chart in One-shot Buzzer Mode

16.4.4 Output in Melody Mode

Melody mode generates the buzzer signal with a melody according to the data written to the sound buffer (SNDDAT register) successively, and outputs the generated signal to outside the IC. An output start procedure and the SNDA operations are shown below.

Melody output start procedure

1. Set the following SNDSEL register bits:
   - Set the SNDSEL.MOSEL[1:0] bits to 0x2. (Set melody mode)
   - SNDSEL.STIM[3:0] bits (Set tempo)
2. Write data to the following sound buffer (SNDDAT register) bits. (Start sound output)
   - SNDDAT.MDTI bit (Set tie)
   - SNDDAT.MDRS bit (Set note/rest)
   - SNDDAT.SLEN[5:0] bits (Set duration)
   - SNDDAT.SFRQ[7:0] bits (Set scale)
3. Check to see if the SNDINTF.EMIF bit is set to 1 (an interrupt can be used).
4. Repeat Steps 2 and 3 until the end of the melody.

Melody output operations

When data is written to the sound buffer (SNDDAT register), SNDA clears the SNDINTF.EMIF bit (sound buffer empty interrupt flag) to 0 and starts sound output operations.

The data written to the sound buffer is loaded into the sound register by the internal trigger signal. At the same time, the SNDINTF.EMIF bit and SNDINTF.SBSY bit are both set to 1. The output pin outputs the sound specified.

The sound output stops if data is not written to the sound buffer (SNDDAT register) until the next trigger is issued. At the same time, the SNDINTF.EDIF bit (sound output completion interrupt flag) is set to 1 and the SNDINTF.SBSY bit is cleared to 0.

Figure 16.4.4.1 shows a melody mode operation timing chart.

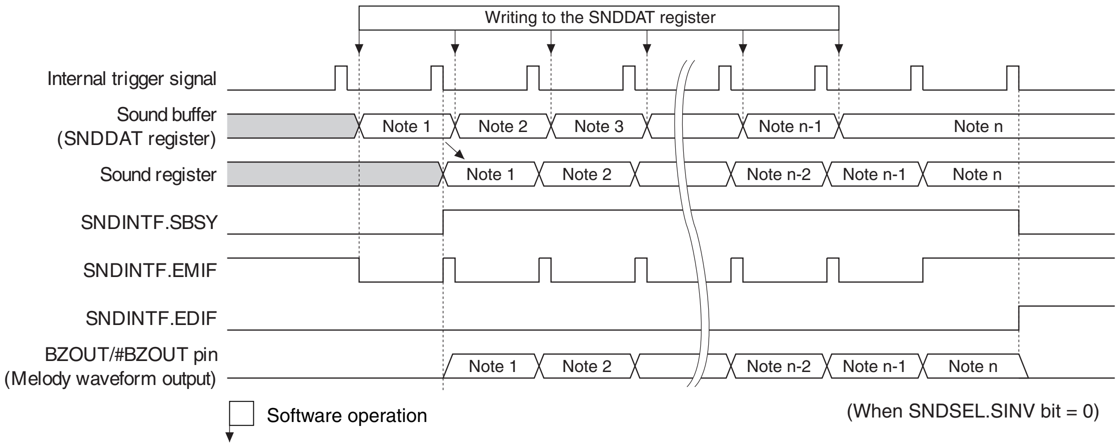


Figure 16.4.4.1 Melody Mode Operation Timing Chart

Melody output waveform configuration

Note/rest (duration) specification

Notes and rests can be specified using the SNDDAT.MDRS and SNDDAT.SLEN[5:0] bits.

Table 16.4.4.1 Note/Rest Specification (when f_{CLK_SNDA} = 32,768 Hz)

SNDDAT.SLEN[5:0] bits	SNDDAT.MDRS bit	
	0: Note	**1: Rest**
0x0f	Half note	Half rest
0x0b	Dotted quarter note	Dotted quarter rest
0x07	Quarter note	Quarter rest
0x05	Dotted eighth note	Dotted eighth rest
0x03	Eighth note	Eighth rest
0x01	Sixteenth note	Sixteenth rest
0x00	Thirty-second note	Thirty-second rest
Other	Setting prohibited	

Tie specification

A tie takes effect by setting the SNDDAT.MDTI bit to 1 and the previous note and the current note are played continuously as a single note. The notes to be tied must be the same pitch.

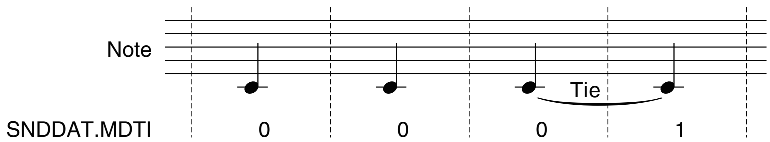


Figure 16.4.4.2 Tie

Note: The SNDDAT.MDTI bit cannot be set to 1 at the first note that begins playing.

Scale specification

Scales can be specified using the SNDDAT.SFRQ[7:0] bits.

Table 16.4.4.2 Scale Specification (when f_{CLK_SNDA} = 32,768 Hz)

SNDDAT.SFRQ[7:0] bits	Scale	Frequency [Hz]
0xf8	C3	131.60
0xea	C#3	139.44
0xdd	D3	147.60
0xd1	D#3	156.04
0xc5	E3	165.49
0xba	F3	175.23
0xaf	F#3	186.18
0xa5	G3	197.40
0x9c	G#3	208.71
0x93	A3	221.41
0x8b	A#3	234.06

SNDDAT.SFRQ[7:0] bits	Scale	Frequency [Hz]
0x83	B3	248.24
0x7c	C4	262.14
0x75	C#4	277.69
0x6e	D4	295.21
0x68	D#4	312.08
0x62	E4	330.99
0x5c	F4	352.34
0x57	F#4	372.36
0x52	G4	394.80
0x4e	G#4	414.78
0x49	A4	442.81
0x45	A#4	468.11
0x41	B4	496.48
0x3d	C5	528.52
0x3a	C#5	555.39
0x37	D5	585.14
0x33	D#5	630.15
0x30	E5	668.73
0x2e	F5	697.19
0x2b	F#5	744.73
0x29	G5	780.19
0x26	G#5	840.21
0x24	A5	885.62
0x22	A#5	936.23
0x20	B5	992.97
0x1e	C6	1057.03

16.5 Interrupts

SNDA has a function to generate the interrupts shown in Table 16.5.1.

Table 16.5.1 SNDA Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Sound buffer empty	SNDINTF.EMIF	When data in the sound buffer (SNDDAT register) is transferred to the sound register or 1 is written to the SNDCTL.SSTP bit	Writing to the SNDDAT register
Sound output completion	SNDINTF.EDIF	When a sound output has completed	Writing 1 or writing to the SNDDAT register

SNDA 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

SNDA Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SNDCLK	15–9	–	0x00	–	R	–
	8	DBRUN	0	H0	R/W	
	7	–	0	–	R	
	6–4	CLKDIV[2:0]	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 SNDA 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

Bit 7 Reserved

Bits 6–4 CLKDIV[2:0]

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

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of SNDA.

Table 16.6.1 Clock Source and Division Ratio Settings

SNDCLK. CLKDIV[2:0] bits	SNDCLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x7	Reserved	1/1	Reserved	1/1
0x6				
0x5	1/128		1/128	
0x4	1/64		1/64	
0x3	1/32		1/32	
0x2	1/16		1/16	
0x1	1/8		1/8	
0x0	1/4		1/4	

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

Note: The SNDCLK register settings can be altered only when the SNDCTL.MODEN bit = 0.

SNDA Select Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SNDSEL	15–12	–	0x0	–	R	–
	11–8	STIM[3:0]	0x0	H0	R/W	
	7–3	–	0x00	–	R	
	2	SINV	0	H0	R/W	
	1–0	MOSEL[1:0]	0x0	H0	R/W	

Bits 15–12 Reserved

Bits 11–8 STIM[3:0]

These bits select a tempo (when melody mode is selected) or a one-shot buzzer output duration (when one-shot buzzer mode is selected).

Table 16.6.2 Tempo/One-shot Buzzer Output Duration Selections (when f_{CLK_SNDA} = 32,768 Hz)

SNDSEL. STIM[3:0] bits	Tempo (= Quarter note/minute)	One-shot buzzer output duration [ms]
0xf	30	240.2
0xe	32	224.6
0xd	34.3	208.8
0xc	36.9	193.4
0xb	40	177.7
0xa	43.6	162.1
0x9	48	146.5
0x8	53.3	129.0
0x7	60	115.2
0x6	68.6	99.6
0x5	80	84.0
0x4	96	68.4
0x3	120	52.7
0x2	160	37.1
0x1	240	21.5
0x0	480	5.9

Note: Be sure to avoid altering these bits when SNDINTF.SBSY bit = 1.

Bits 7–3 Reserved

Bit 2 **SINV**

This bit selects an output pin drive mode.

1 (R/W): Normal drive mode

0 (R/W): Direct drive mode

For more information, refer to "Output Pin Drive Mode."

Bits 1–0 **MOSEL[1:0]**

These bits select a sound output mode.

Table 16.6.3 Sound Output Mode Selection

SNDSEL.MOSEL[1:0] bits	Sound output mode
0x3	Reserved
0x2	Melody mode
0x1	One-shot buzzer mode
0x0	Normal buzzer mode

SNDA Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SNDCTL	15–9	–	0x00	–	R	–
	8	SSTP	0	H0	R/W	
	7–1	–	0x00	–	R	
	0	MODEN	0	H0	R/W	

Bits 15–9 **Reserved**

Bit 8 **SSTP**

This bit stops sound output.

1 (W): Stop sound output

0 (W): Ineffective

1 (R): In stop process

0 (R): Stop process completed/Idle

The SNDCTL.SSTP bit is used to stop buzzer output in normal buzzer mode. After 1 is written, this bit is cleared to 0 when the sound output has completed. Also in one-shot buzzer mode/melody mode, writing 1 to this bit can forcibly terminate the sound output.

Note: If the SNDCTL.SSTP bit is set to 1 while the sound buffer contains written data (SNDINTF.EMIF bit = 0), the sound buffer data will be loaded to the sound register immediately after the currently output waveform ends and the subsequent sound output begins.

Bits 7–1 **Reserved**

Bit 0 **MODEN**

This bit enables the SNDA operations.

1 (R/W): Enable SNDA operations (The operating clock is supplied.)

0 (R/W): Disable SNDA operations (The operating clock is stopped.)

SNDA Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SNDDAT	15	MDTI	0	H0	R/W	–
	14	MDRS	0	H0	R/W	
	13–8	SLEN[5:0]	0x00	H0	R/W	
	7–0	SFRQ[7:0]	0xff	H0	R/W	

This register functions as a sound buffer. Writing data to this register starts sound output. For detailed information on the setting data, refer to "Buzzer output waveform configuration (normal buzzer mode/one-shot buzzer mode)" and "Melody output waveform configuration."

Bit 15 MDTI

This bit specifies a tie (continuous play with the previous note) in melody mode.
1 (R/W): Enable tie
0 (R/W): Disable tie

This bit is ignored in normal buzzer mode/one-shot buzzer mode. A tie at the first note data of the music is ineffective.

Bit 14 MDRS

This bit selects the output type in melody mode from a note or a rest .
1 (R/W): Rest
0 (R/W): Note

When a rest is selected, the BZOUT pin goes low and the #BZOUT pin goes high during the output duration. This bit is ignored in normal buzzer mode/one-shot buzzer mode.

Bits 13–8 SLEN[5:0]

These bits select a duration (when melody mode is selected) or a buzzer signal duty ratio (when normal buzzer mode/one-shot buzzer mode is selected).

Bits 7–0 SFRQ[7:0]

These bits select a scale (when melody mode is selected) or a buzzer signal frequency (when normal buzzer mode/one-shot buzzer mode is selected).

Notes:
- In normal buzzer mode/one-shot buzzer mode, only the low-order 6 bits (SNDDAT.SFRQ[5:0] bits) are effective within the SNDDAT.SFRQ[7:0] bits. Always set the SNDDAT.SFRQ[7:6] bits to 0x0.
- The SNDDAT register allows 16-bit data writing only. Data writings in 8-bit size will be ignored.

SNDA Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SNDINTF	15–9	–	0x00	–	R	–
	8	SBSY	0	H0	R	
	7–2	–	0x00	–	R	
	1	EMIF	1	H0	R	Cleared by writing to the SNDDAT register.
	0	EDIF	0	H0	R/W	Cleared by writing 1 or writing to the SNDDAT register.

Bits 15–9 Reserved

Bit 8 SBSY

This bit indicates the sound output status. (See Figures 16.4.2.1, 16.4.3.1, and 16.4.4.1.)
1 (R): Outputting
0 (R): Idle

Bits 7–2 Reserved

Bit 1 EMIF
Bit 0 EDIF

These bits indicate the SNDA 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:
SNDINTF.EMIF bit: Sound buffer empty interrupt
SNDINTF.EDIF bit: Sound output completion interrupt

SNDA Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SNDINTE	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	EMIE	0	H0	R/W	
	0	EDIE	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 EMIE

Bit 0 EDIE

These bits enable SNDA interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:

SNDINTE.EMIE bit: Sound buffer empty interrupt

SNDINTE.EDIE bit: Sound output completion interrupt

17 LCD Driver (LCD8B)

17.1 Overview

LCD8B is an LCD driver to drive an LCD panel. The features of the LCD8B are listed below.

- The frame frequency is configurable into 16 steps.
- Provides all on, all off, and inverse display functions as well as normal display.
- The segment and common pin assignments can be inverted.
- Provides a partial common output drive function.
- Provides an n-segment-line inverse AC drive function.
- The LCD contrast is adjustable into 32 steps.
- Includes a power supply for 1/3 bias and 1/4 bias driving (allows external voltages to be applied).
- Can generate interrupts every frame.

Figure 17.1.1 shows the LCD8B configuration.

Table 17.1.1 LCD8B Configuration of S1C17W15

Item	S1C17W15 100-pin package/chip	S1C17W15 80-pin package	S1C17W15 64-pin package
Number of segments supported	Max. 136 segments (34 segments × 4 commons)	Max. 128 segments (32 segments × 4 commons)	Max. 96 segments (24 segments × 4 commons)
	Max. 240 segments (30 segments × 8 commons)	Max. 224 segments (28 segments × 8 commons)	Max. 160 segments (20 segments × 8 commons)
SEG/COM outputs	34SEG × 1–4COM 30SEG × 5–8COM	32SEG × 1–4COM 28SEG × 5–8COM	24SEG × 1–4COM 20SEG × 5–8COM
Drive bias	1/3 bias, 1/4 bias		
Embedded display data RAM	68 bytes		

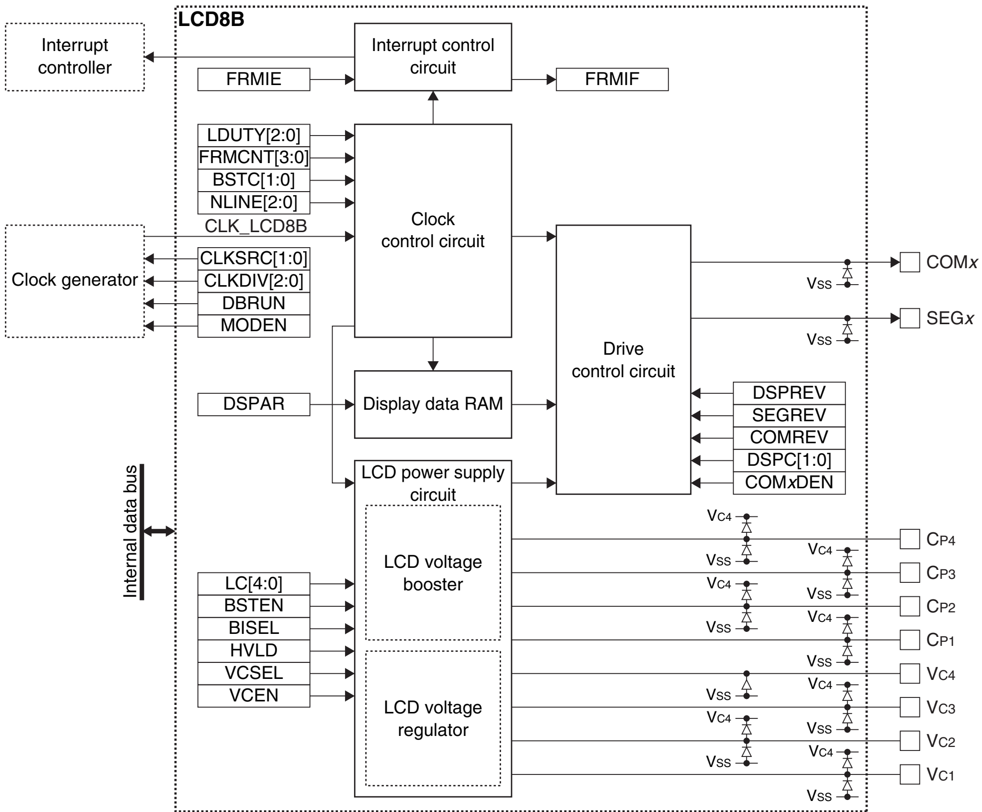


Figure 17.1.1 LCD8B Configuration

17.2 Output Pins and External Connections

17.2.1 List of Output Pins

Table 17.2.1.1 lists the LCD8B pins.

Table 17.2.1.1 List of LCD8B Pins

Pin name	I/O*1	Initial status*1	Function
COM0–3	A	Hi-Z / O (L)*2	Common data output-only pin
COM4–7/SEG0–3	A	Hi-Z / O (L)*2	Common data output/segment data output pin
SEG4–15	A	Hi-Z / O (L)*2	Segment data output-only pin
SEG16–23	A	O (L)	General-purpose IO/segment data output pin
SEG24–27	A	Hi-Z / O (L)*2	Segment data output-only pin (Not available in the 64-pin package)
SEG28–29	A	Hi-Z / O (L)*2	Segment data output-only pin (Not available in the 64-pin/80-pin package)
SEG30–33	A	Hi-Z / O (L)*2	Segment data output-only pin (Not available in the 64-pin package)
V_{C1}	P	–	LCD panel drive power supply pin
V_{C2}	P	–	LCD panel drive power supply pin
V_{C3}	P	–	LCD panel drive power supply pin
V_{C4}	P	–	LCD panel drive power supply pin
C_{P1}	A	–	LCD voltage booster capacitor connecting pin
C_{P2}	A	–	LCD voltage booster capacitor connecting pin
C_{P3}	A	–	LCD voltage booster capacitor connecting pin
C_{P4}	A	–	LCD voltage booster capacitor connecting pin

*1: Indicates the status when the pin is configured for LCD8B. *2: When LCD8CTL.MODEN bit = 1

If the port is shared with the LCD8B pin and other functions, the LCD8B output function must be assigned to the port before activating the LCD8B. For more information, refer to the "I/O Ports" chapter.
The COM4–7 outputs and SEG0–4 outputs share the pins and selecting a drive duty switches the pins to COM pins or SEG pins. For the pin configuration, refer to "Drive Duty Switching."

Notes: • Be sure to avoid using the V_{C1} to V_{C4} pin outputs for driving external circuits.
• When an LCD panel is connected, set the LCD8CTL.MODEN bit to 1, as activating the LCD panel when it is set to 0 may cause the LCD panel characteristics to fluctuate.

17.2.2 External Connections

Figure 17.2.2.1 shows a connection diagram between LCD8B and an LCD panel.

Note: When the panel is connected, the LCD8CTL.MODEN bit must be set to 1 to bias the panel even if display is turned off.

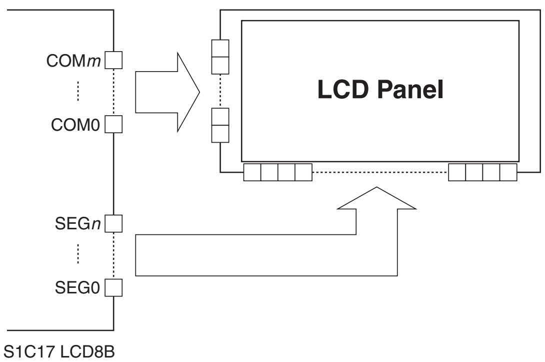


Figure 17.2.2.1 Connections between LCD8B and an LCD Panel

17.3 Clock Settings

17.3.1 LCD8B Operating Clock

When using LCD8B, the LCD8B operating clock CLK_LCD8B must be supplied to LCD8B from the clock generator. The CLK_LCD8B 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 LCD8CLK register bits:
   - LCD8CLK.CLKSRC[1:0] bits (Clock source selection)
   - LCD8CLK.CLKDIV[2:0] bits (Clock division ratio selection = Clock frequency setting)

The CLK_LCD8B frequency should be set to around 32 kHz.

17.3.2 Clock Supply in SLEEP Mode

When using LCD8B during SLEEP mode, the LCD8B operating clock CLK_LCD8B must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_LCD8B clock source.

17.3.3 Clock Supply in DEBUG Mode

The CLK_LCD8B supply during DEBUG mode should be controlled using the LCD8CLK.DBRUN bit.
The CLK_LCD8B supply to LCD8B is suspended when the CPU enters DEBUG mode if the LCD8CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_LCD8B supply resumes. Although LCD8B stops operating and the display is turned off when the CLK_LCD8B supply is suspended, the registers retain the status before DEBUG mode was entered. If the LCD8CLK.DBRUN bit = 1, the CLK_LCD8B supply is not suspended and LCD8B will keep operating in DEBUG mode.

17.3.4 Frame Frequency

The LCD8B frame signal is generated by dividing CLK_LCD8B. The frame frequency is determined by selecting a division ratio from 16 variations depending on the drive duty using the LCD8TIM1.FRMCNT[3:0] bits. Use the following equation to calculate the frame frequency.

$$f_{FR} = \frac{f_{CLK\_LCD8B}}{16 \times (FRMCNT + 1) \times (LDUTY + 1)} \qquad \text{(Eq. 17.1)}$$

Where
- f_{FR}: Frame frequency [Hz]
- f_{CLK_LCD8B}: LCD8B operating clock frequency [Hz]
- FRMCNT: LCD8TIM1.FRMCNT[3:0] setting value (0 to 15)
- LDUTY: LCD8TIM1.LDUTY[2:0] setting value (0 to 7)

Table 17.3.4.1 lists frame frequency settings when f_{CLK_LCD8B} = 32,768 Hz as an example.

Table 17.3.4.1 Frame Frequency Settings (when f_{CLK_LCD8B} = 32,768 Hz)

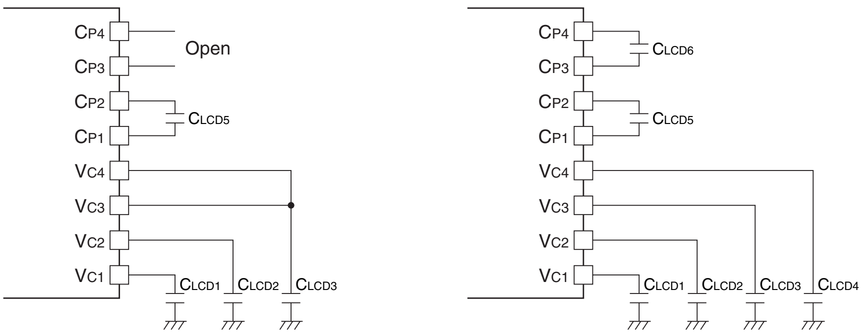
LCD8TIM1.FRMCNT[3:0] bits	Frame frequency [Hz]							
	1/8 duty	1/7 duty	1/6 duty	1/5 duty	1/4 duty	1/3 duty	1/2 duty	Static
0xf	16.0	18.3	21.3	25.6	32.0	42.7	64.0	128.0
0xe	17.1	19.5	22.8	27.3	34.1	45.5	68.3	136.5
0xd	18.3	20.9	24.4	29.3	36.6	48.8	73.1	146.3
0xc	19.7	22.5	26.3	31.5	39.4	52.5	78.8	157.5
0xb	21.3	24.4	28.4	34.1	42.7	56.9	85.3	170.7
0xa	23.3	26.6	31.0	37.2	46.5	62.1	93.1	186.2
0x9	25.6	29.3	34.1	41.0	51.2	68.3	102.4	204.8
0x8	28.4	32.5	37.9	45.5	56.9	75.9	113.8	227.6
0x7	32.0	36.6	42.7	51.2	64.0	85.3	128.0	256.0
0x6	36.6	41.8	48.8	58.5	73.1	97.5	146.3	292.6
0x5	42.7	48.8	56.9	68.3	85.3	113.8	170.7	341.3
0x4	51.2	58.5	68.3	81.9	102.4	136.5	204.8	409.6
0x3	64.0	73.1	85.3	102.4	128.0	170.7	256.0	512.0
0x2	85.3	97.5	113.8	136.5	170.7	227.6	341.3	682.7
0x1	128.0	146.3	170.7	204.8	256.0	341.3	512.0	1,024.0
0x0	256.0	292.6	341.3	409.6	512.0	682.7	1,024.0	2,048.0

17.4 LCD Power Supply

The LCD drive voltages V_{C1} to V_{C4} can be generated by the internal LCD power supply circuit (LCD voltage regulator and LCD voltage booster). One or all voltages can also be applied from outside the IC.

17.4.1 Internal Generation Mode

This mode generates all the LCD drive voltages V_{C1} to V_{C4} on the chip. To put LCD8B into internal generation mode, set both the LCD8PWR.VCEN and LCD8PWR.BSTEN bits to 1 to turn both the LCD voltage regulator and LCD voltage booster on. In addition to this, select either 1/3 bias or 1/4 bias using the LCD8PWR.BISEL bit. Figure 17.4.1.1 shows an external connection example for internal generation mode.

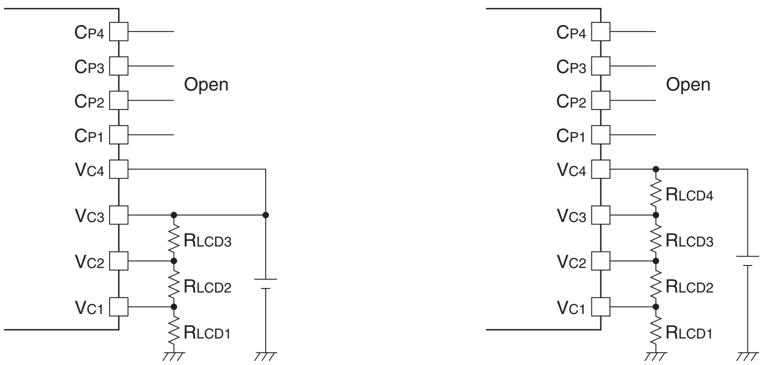


When 1/3 bias is selected (LCD8PWR.BISEL bit = 0) When 1/4 bias is selected (LCD8PWR.BISEL bit = 1)

Figure 17.4.1.1 External Connection Example for Internal Generation Mode

17.4.2 External Voltage Application Mode 1

In this mode, all the LCD drive voltages V_{C1} to V_{C4} are applied from outside the IC. To put LCD8B into external voltage application mode 1, set both the LCD8PWR.VCEN and LCD8PWR.BSTEN bits to 0 to turn both the LCD voltage regulator and LCD voltage booster off. Figure 17.4.2.1 shows an external connection example for external voltage application mode 1.



When 1/3 bias is selected (LCD8PWR.BISEL bit = 0) When 1/4 bias is selected (LCD8PWR.BISEL bit = 1)

Figure 17.4.2.1 External Connection Example for External Voltage Application Mode 1 (resistor divider)

17.4.3 External Voltage Application Mode 2

In this mode, the LCD drive voltage V_{C1} or V_{C2} is applied from outside the IC and other voltages are internally generated. To put LCD8B into external voltage application mode 2, set the LCD8PWR.VCEN bit to 0 to turn the LCD voltage regulator off and the LCD8PWR.BSTEN bit to 1 to turn the LCD voltage booster on. Figure 17.4.3.1 shows an external connection example for external voltage application mode 2.

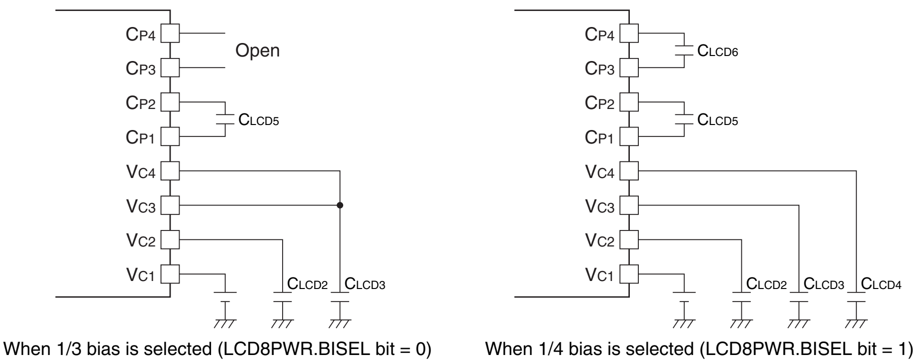


Figure 17.4.3.1 External Connection Example for External Voltage Application Mode 2 (when V_{C1} is applied)

17.4.4 LCD Voltage Regulator Settings

When using internal generation mode, select the reference voltage for boosting voltage generated by the LCD voltage regulator according to the power supply voltage V_{DD}. Refer to "LCD Driver (LCD8B) Characteristics" in the "Electrical Characteristics" chapter and set the LCD8PWR.VCSEL bit. Current consumption can be reduced by selecting reference voltage V_{C2} as compared with reference voltage V_{C1}. By setting the LCD8PWR.HVLD bit to 1, the LCD voltage regulator enters heavy load protection mode and ensures stable V_{C1} to V_{C4} outputs. Heavy load protection mode should be set when the display has inconsistencies in density. Current consumption increases in heavy load protection mode, therefore do not set heavy load protection mode if unnecessary.

17.4.5 LCD Voltage Booster Setting

Set the booster clock frequency used in the LCD voltage booster using the LCD8TIM2.BSTC[1:0] bits. Set it to the frequency that provides the best V_{C1}–V_{C4} output stability after being evaluated using the actual circuit board.

17.4.6 LCD Contrast Adjustment

The LCD panel contrast can be adjusted within 32 levels using the LCD8PWR.LC[4:0] bits. This function is realized by controlling the voltage output from the LCD voltage regulator. Therefore, the LCD8PWR.LC[4:0] bits cannot be used for contrast adjustment in external voltage application modes 1 and 2.

17.5 Operations

17.5.1 Initialization

The LCD8B should be initialized with the procedure shown below.

1. Assign the LCD8B output function to the ports. (Refer to the "I/O Ports" chapter.)
2. Configure the LCD8CLK.CLKSRC[1:0] and LCD8CLK.CLKDIV[2:0] bits. (Configure operating clock)
3. Write 1 to the LCD8CTL.MODEN bit. (Enable LCD8B operating clock)
4. Configure the following LCD8TIM1 register bits:
   - LCD8TIM1.LDUTY[2:0] bits (Set drive duty)
   - LCD8TIM1.FRMCNT[3:0] bits (Set frame frequency)
5. Configure the following LCD8TIM2 register bits:
   - LCD8TIM2.NLINE[2:0] bits (Set n-line inverse AC drive)
   - LCD8TIM2.BSTC[1:0] bits (Set booster clock frequency)
6. Configure the following LCD8PWR register bits:
   - LCD8PWR.VCEN bit (Enable LCD voltage regulator)
   - LCD8PWR.VCSEL bit (Set reference voltage for boosting)
   - LCD8PWR.BISEL bit (Set bias)
   - LCD8PWR.BSTEN bit (Enable LCD voltage booster)
   - LCD8PWR.LC[4:0] bits (Set LCD contrast initial value)

7. Configure the following LCD8DSP register bits:
   - LCD8DSP.DSPAR bit (Select display area)
   - LCD8DSP.COMREV bit (Select COM pin assignment direction)
   - LCD8DSP.SEGREV bit (Select SEG pin assignment direction)
8. Write display data to the display data RAM.
9. Set the following bits when using the interrupt:
   - Write 1 to the LCD8INTF.FRMIF bit. (Clear interrupt flag)
   - Set the LCD8INTE.FRMIE bit to 1. (Enable LCD8B interrupt)

17.5.2 Display On/Off

The LCD display state is controlled using the LCD8DSP.DSPC[1:0] bits.

Table 17.5.2.1 LCD Display Control

LCD8DSP.DSPC[1:0] bits	LCD display
0x3	All off (static drive)
0x2	All on
0x1	Normal display
0x0	Display off

When "Display off" is selected, the drive voltage supply stops and the LCD driver pin outputs are all set to V_{SS} level.
Since "All on" and "All off" directly control the driving waveform output by the LCD driver, data in the display data RAM is not altered. The common pins are set to dynamic drive for "All on" and to static drive for "All off." This function can be used to make the display flash on and off without altering the display memory.

Note: When "Display off" is selected, the electric charges of V_{C4} (or V_{C3}) must be discharged in the following procedure.

<When the LCD drive voltages are generated in internal generation mode>
1. Set the LCD8DSP.DSPC[1:0] bit to 0x0. (Display off)
2. Wait until the electric charge of the LCD panel is discharged.
3. Set the LCD8PWR.VCEN bit to 0. (Disable LCD voltage regulator)
4. Set the LCD8PWR.HVLD bit to 1. (Enable heavy load protection mode)
5. Before stopping CLK_LCD8B, make certain that the V_{C4} (or V_{C3}) voltage has dropped to V_{DD} - 1 V or less.

<When the LCD drive voltages are generated in an external generation mode>
1. Set the LCD8DSP.DSPC[1:0] bit to 0x0. (Display off)
2. Wait until the electric charge of the LCD panel is discharged.
3. Turn the external power supply off.
4. Set the LCD8PWR.HVLD bit to 1. (Enable heavy load protection mode)
5. Before stopping CLK_LCD8B, make certain that the V_{C4} (or V_{C3}) voltage has dropped to V_{DD} - 1 V or less.

To turn the display on again, perform the above procedure in the reverse order.

17.5.3 Inverted Display

The LCD panel display can be inverted (black/white inversion) using merely control bit manipulation, without rewriting the display data RAM. Setting the LCD8DSP.DSPREV bit to 0 inverts the display; setting it to 1 returns the display to normal status. Note that the display will not be inverted when the LCD8DSP.DSPC[1:0] bits = 0x3 (All off).

17.5.4 Drive Duty Switching

Drive duty can be set to 1/8 to 1/2 or static drive using the LCD8TIM1.LDUTY[2:0] bits. Table 17.5.4.1 shows the correspondence between the LCD8TIM1.LDUTY[2:0] bit settings, drive duty, and maximum number of display segments.

Table 17.5.4.1 Drive Duty Settings

LCD8TIM1. LDUTY[2:0] bits	Duty	Valid COM pins	Valid SEG pins			Max. number of display dots/segments		
			64pin	80pin	100pin/Chip	64pin	80pin	100pin/Chip
0x7	1/8	COM0–7	SEG4–23	SEG4–27, SEG30–33	SEG4–33	160	224	240
0x6	1/7	COM0–6				140	196	210
0x5	1/6	COM0–5				120	168	180
0x4	1/5	COM0–4				100	140	150
0x3	1/4	COM0–3	SEG0–23	SEG0–27, SEG30–33	SEG0–33	96	128	136
0x2	1/3	COM0–2				72	96	102
0x1	1/2	COM0–1				48	64	68
0x0	Static	COM0				24	32	34

Unused common pins output an OFF waveform that turns the segments off.

The some pins are shared with a SEG output and a COM output, and they are configured to the SEG or COM pin according to the drive duty selected.

Table 17.5.4.2 SEG/COM Pin Configuration

Pin	Duty							
	1/8	1/7	1/6	1/5	1/4	1/3	1/2	Static
COM0	COM0	COM0	COM0	COM0	COM0	COM0	COM0	COM0
COM1	COM1	COM1	COM1	COM1	COM1	COM1	COM1	Unused
COM2	COM2	COM2	COM2	COM2	COM2	COM2	Unused	Unused
COM3	COM3	COM3	COM3	COM3	COM3	Unused	Unused	Unused
COM4/SEG0	COM4	COM4	COM4	COM4	SEG0	SEG0	SEG0	SEG0
COM5/SEG1	COM5	COM5	COM5	Unused	SEG1	SEG1	SEG1	SEG1
COM6/SEG2	COM6	COM6	Unused	Unused	SEG2	SEG2	SEG2	SEG2
COM7/SEG3	COM7	Unused	Unused	Unused	SEG3	SEG3	SEG3	SEG3
SEG4–33	SEG4–33	SEG4–33	SEG4–33	SEG4–33	SEG4–33	SEG4–33	SEG4–33	SEG4–33

17.5.5 Drive Waveforms

Figures 17.5.5.1 to 17.5.5.4 show some drive waveform examples.

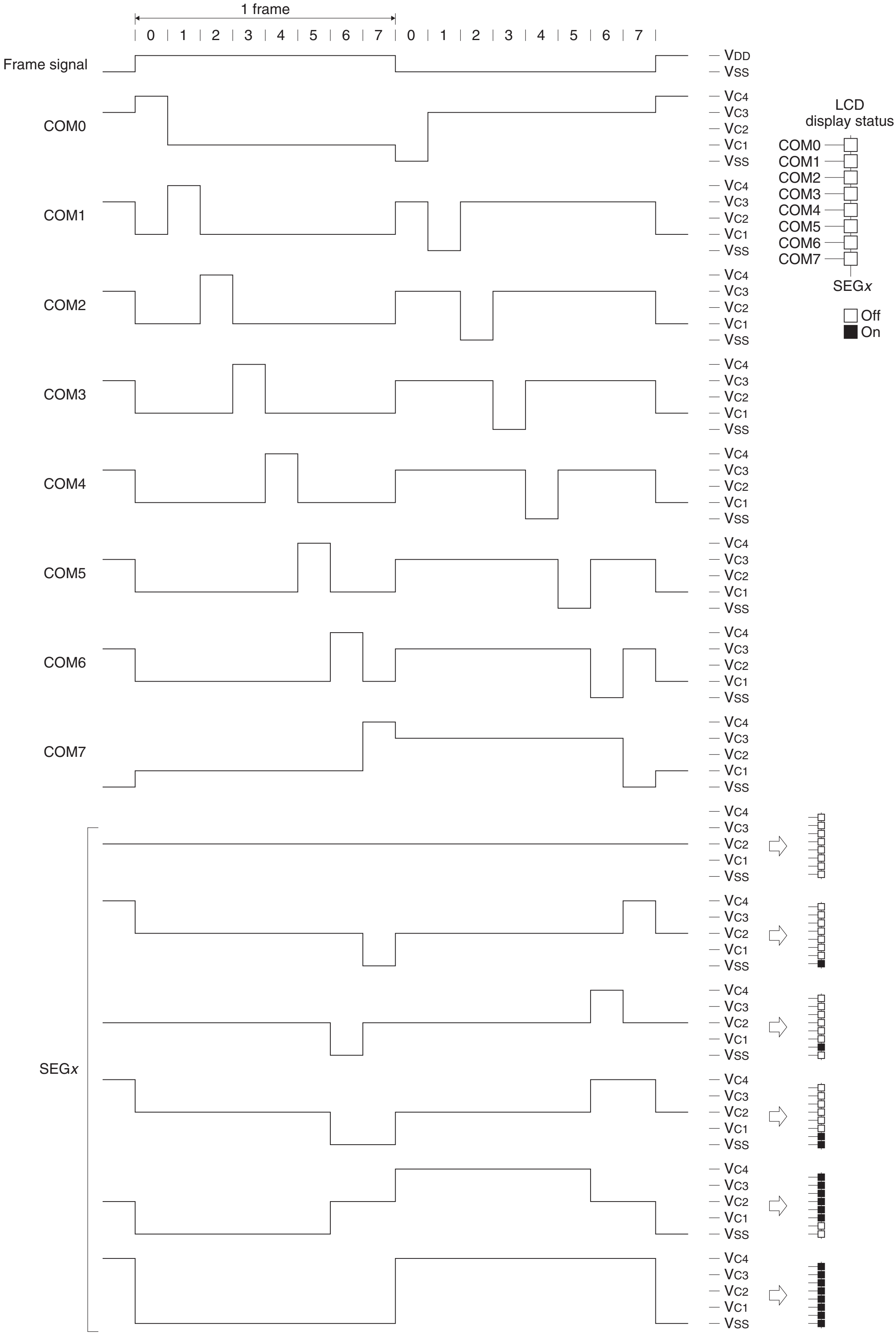


Figure 17.5.5.1 1/8 Duty Drive Waveform (1/4 bias)

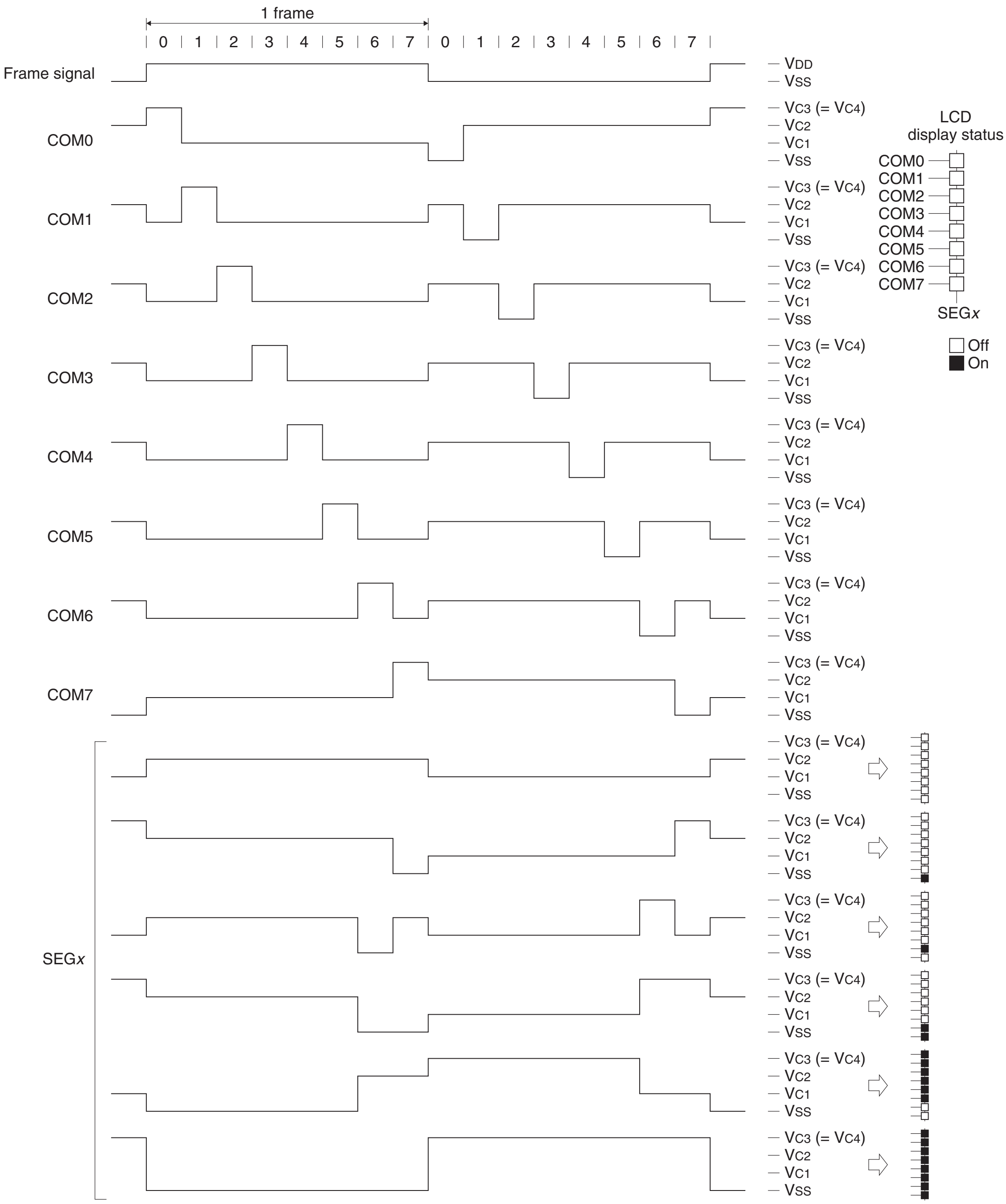


Figure 17.5.5.2 1/8 Duty Drive Waveform (1/3 bias)

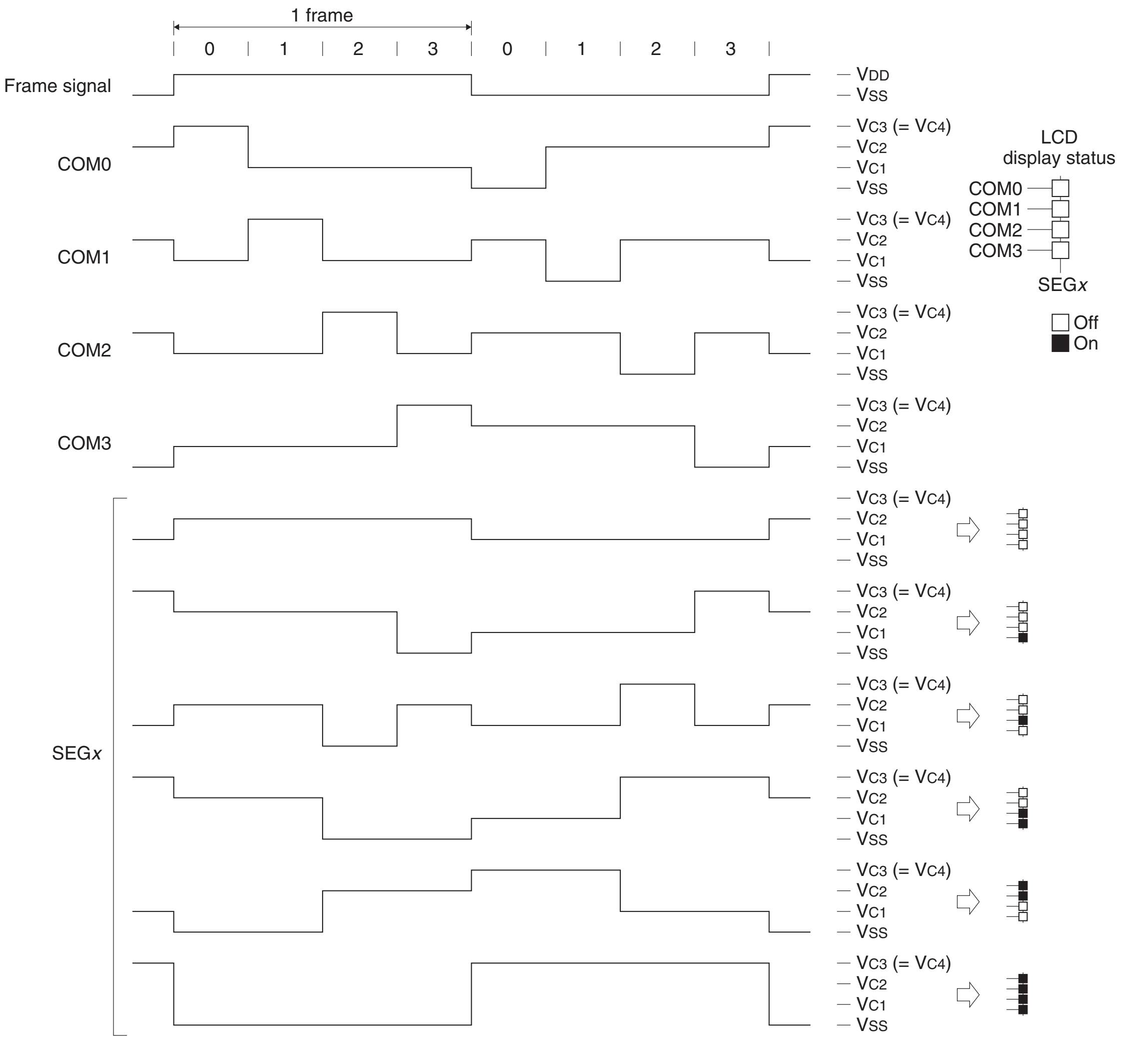


Figure 17.5.5.3 1/4 Duty Drive Waveform (1/3 bias)

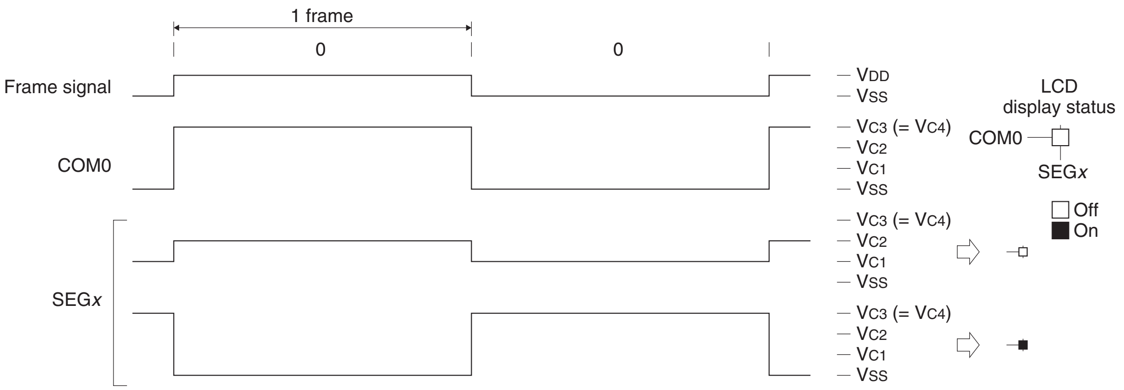


Figure 17.5.5.4 Static Drive Waveform (1/3 bias)

17.5.6 Partial Common Output Drive

By setting the LCD8COMC∗.COM*x*DEN bit (*x* = COM No.) to 0, any common outputs can be set to off waveform regardless of the display data RAM contents. The partial common output drive function limits the display to the required area only to reduce power consumption.

17.5.7 n-Segment-Line Inverse AC Drive

The n-line inverse AC drive function may improve the display quality when being reduced such as when cross-talk occurs. To activate the n-line inverse AC drive function, select the number of lines to be inverted using the LCD8TIM2.NLINE[2:0] bits. The setting value should be determined after being evaluated using the actual circuit board. Note that using the n-line inverse AC drive function increases current consumption.

Table 17.5.7.1 Selecting Number of Inverse Lines

LCD8TIM2.NLINE[2:0] bits	Number of inverse lines
0x7	7 lines
:	:
0x1	1 line
0x0	Normal drive

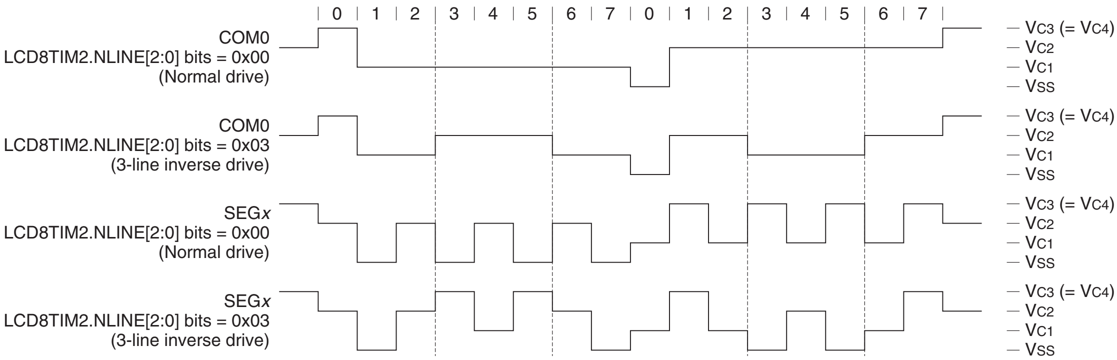


Figure 17.5.7.1 1/8 Duty (1/3 bias) Normal Drive Waveform and 3-line Inverse Drive Waveform

17.6 Display Data RAM

The display data RAM is located beginning with address 0x7000.

The correspondence between the memory bits of the display data RAM and the common/segment pins varies depending on the selected conditions below.

- Drive duty (1/8 to 1/2 or static drive)
- Segment pin assignment (normal or inverse)
- Common pin assignment (normal or inverse)

Figures 17.6.3.1 to 17.6.3.4 show the correspondence between display data RAM and the common/segment pins in some drive duties.

Writing 1 to the display data RAM bit corresponding to a segment on the LCD panel turns the segment on, while writing 0 turns the segment off. Since the display memory is a RAM allowing reading and writing, bits can be controlled individually using logic operation instructions (read-modify-write instructions).

The area unused for display can be used as general-purpose RAM.

17.6.1 Display Area Selection

In the display data RAM, two screen areas can be allocated and the LCD8DSP.DSPAR bit can be used to switch between the screens. Setting the LCD8DSP.DSPAR bit to 0 selects display area 0; setting to 1 selects display area 1.

17.6.2 Segment Pin Assignment

The display data RAM address assignment for the segment pins can be inverted using the LCD8DSP.SEGREV bit. When the LCD8DSP.SEGREV bit is set to 1, memory addresses are assigned to segment pins in ascending order. When the LCD8DSP.SEGREV bit is set to 0, memory addresses are assigned to segment pins in descending order.

17.6.3 Common Pin Assignment

The display data RAM bit assignment for the common pins can be inverted using the LCD8DSP.COMREV bit. When the LCD8DSP.COMREV bit is set to 1, memory bits are assigned to common pins in ascending order. When the LCD8DSP.COMREV bit is set to 0, memory bits are assigned to common pins in descending order.

Bit	Address		LCD8DSP. COMREV bit = 1	LCD8DSP. COMREV bit = 0
D0	0x7000 … 0x7003 Unused area (general-purpose RAM)	0x7004, 0x7005, … 0x7021 Display area 0	COM0	COM7
D1			COM1	COM6
D2			COM2	COM5
D3			COM3	COM4
D4			COM4	COM3
D5			COM5	COM2
D6			COM6	COM1
D7			COM7	COM0
D0	0x7040 … 0x7043 Unused area (general-purpose RAM)	0x7044, 0x7045, … 0x7061 Display area 1	COM0	COM7
D1			COM1	COM6
D2			COM2	COM5
D3			COM3	COM4
D4			COM4	COM3
D5			COM5	COM2
D6			COM6	COM1
D7			COM7	COM0
LCD8DSP.SEGREV bit = 1		SEG4, SEG5, . . . SEG33		
LCD8DSP.SEGREV bit = 0		SEG33, SEG32, . . . SEG4		

* In the 80-pin package, addresses 0x701c to 0x701d and 0x705c to 0x705d (when the LCD8DSP.SEGREV bit = 1) or addresses 0x7008 to 0x7009 and 0x7048 to 0x7049 (when the LCD8DSP.SEGREV bit = 0) that are allocated to SEG28 to SEG29 are unused areas (general-purpose RAM).

(1) 100-pin package/80-pin package/chip

Bit	Address			LCD8DSP. COMREV bit = 1	LCD8DSP. COMREV bit = 0
D0	0x7000 … 0x7003 Unused area (general-purpose RAM)	0x7004, 0x7005, … 0x7017 Display area 0	0x7018 … 0x7021 Unused area (general-purpose RAM)	COM0	COM7
D1				COM1	COM6
D2				COM2	COM5
D3				COM3	COM4
D4				COM4	COM3
D5				COM5	COM2
D6				COM6	COM1
D7				COM7	COM0
D0	0x7040 … 0x7043 Unused area (general-purpose RAM)	0x7044, 0x7045, … 0x7057 Display area 1	0x7058 … 0x7061 Unused area (general-purpose RAM)	COM0	COM7
D1				COM1	COM6
D2				COM2	COM5
D3				COM3	COM4
D4				COM4	COM3
D5				COM5	COM2
D6				COM6	COM1
D7				COM7	COM0
LCD8DSP.SEGREV bit = 1		SEG4, SEG5, . . . SEG23			

(2) 64-pin package (LCD8DSP.SEGREV bit = 1)

Bit	Address							LCD8DSP. COMREV bit = 1	LCD8DSP. COMREV bit = 0
D0	0x7000	Unused area (general-purpose RAM)	0x700d	0x700e	0x700f	Display area 0	0x7021	COM0	COM7
D1								COM1	COM6
D2								COM2	COM5
D3								COM3	COM4
D4								COM4	COM3
D5								COM5	COM2
D6								COM6	COM1
D7								COM7	COM0
D0	0x7040	Unused area (general-purpose RAM)	0x704d	0x704e	0x704f	Display area 1	0x7061	COM0	COM7
D1								COM1	COM6
D2								COM2	COM5
D3								COM3	COM4
D4								COM4	COM3
D5								COM5	COM2
D6								COM6	COM1
D7								COM7	COM0
LCD8DSP.SEGREV bit = 0				SEG23	SEG22	. . .	SEG4		

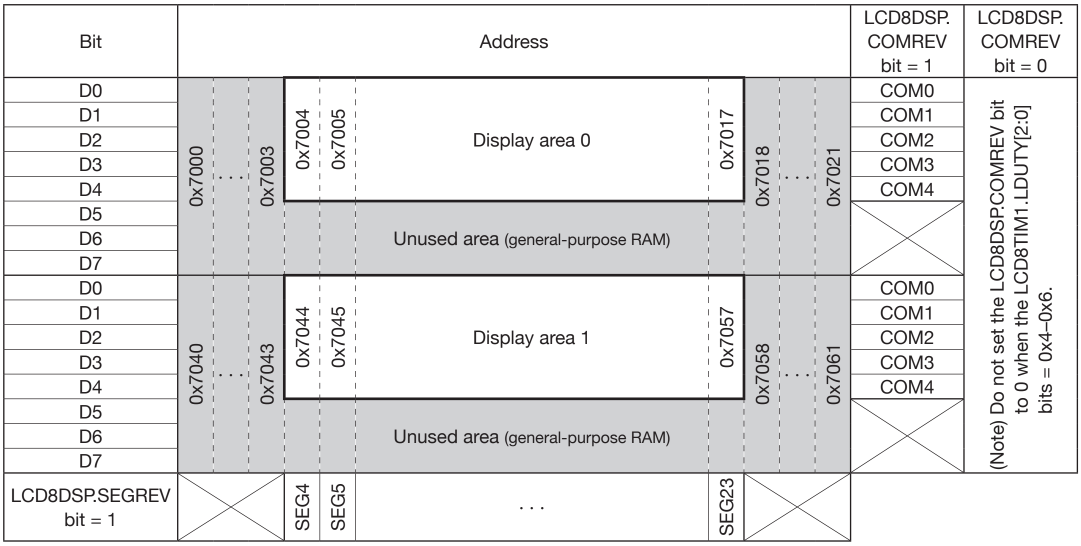
(3) 64-pin package (LCD8DSP.SEGREV bit = 0)

Figure 17.6.3.1 Display Data RAM Map (1/8 duty)

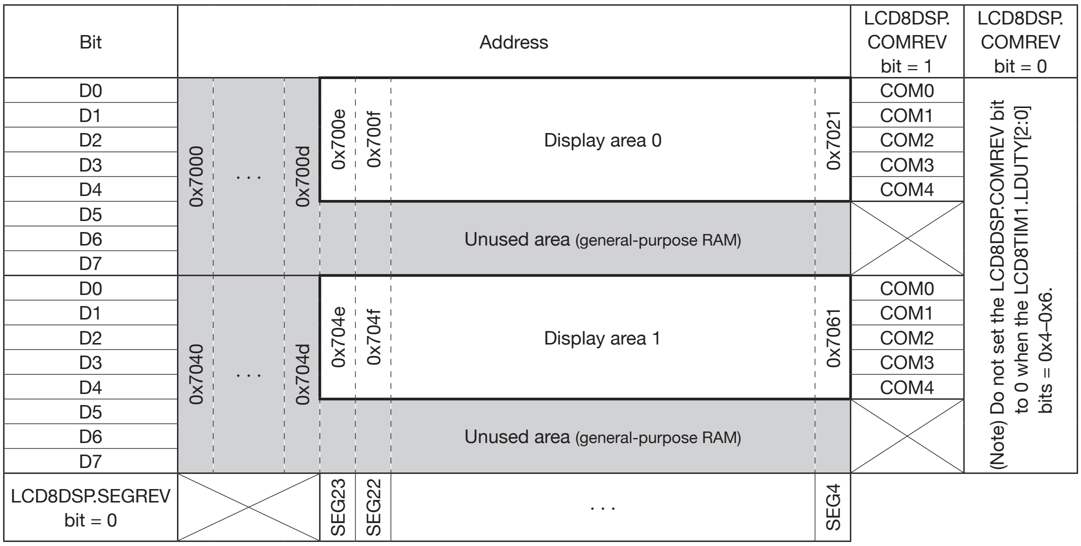
Bit	Address							LCD8DSP. COMREV bit = 1	LCD8DSP. COMREV bit = 0
D0	0x7000	. . .	0x7003	0x7004	0x7005	Display area 0	0x7021	COM0	(Note) Do not set the LCD8DSP.COMREV bit to 0 when the LCD8TIM1.LDUTY[2:0] bits = 0x4–0x6.
D1								COM1	
D2								COM2	
D3								COM3	
D4								COM4	
D5				Unused area (general-purpose RAM)					
D6									
D7									
D0	0x7040	. . .	0x7043	0x7044	0x7045	Display area 1	0x7061	COM0	
D1								COM1	
D2								COM2	
D3								COM3	
D4								COM4	
D5				Unused area (general-purpose RAM)					
D6									
D7									
LCD8DSP.SEGREV bit = 1				SEG4	SEG5	. . .	SEG33		
LCD8DSP.SEGREV bit = 0				SEG33	SEG32	. . .	SEG4		

* In the 80-pin package, addresses 0x701c to 0x701d and 0x705c to 0x705d (when the LCD8DSP.SEGREV bit = 1) or addresses 0x7008 to 0x7009 and 0x7048 to 0x7049 (when the LCD8DSP.SEGREV bit = 0) that are allocated to SEG28 to SEG29 are unused areas (general-purpose RAM).

(1) 100-pin package/80-pin package/chip



(2) 64-pin package (LCD8DSP.SEGREV bit = 1)



(3) 64-pin package (LCD8DSP.SEGREV bit = 0)

Figure 17.6.3.2 Display Data RAM Map (1/5 duty)

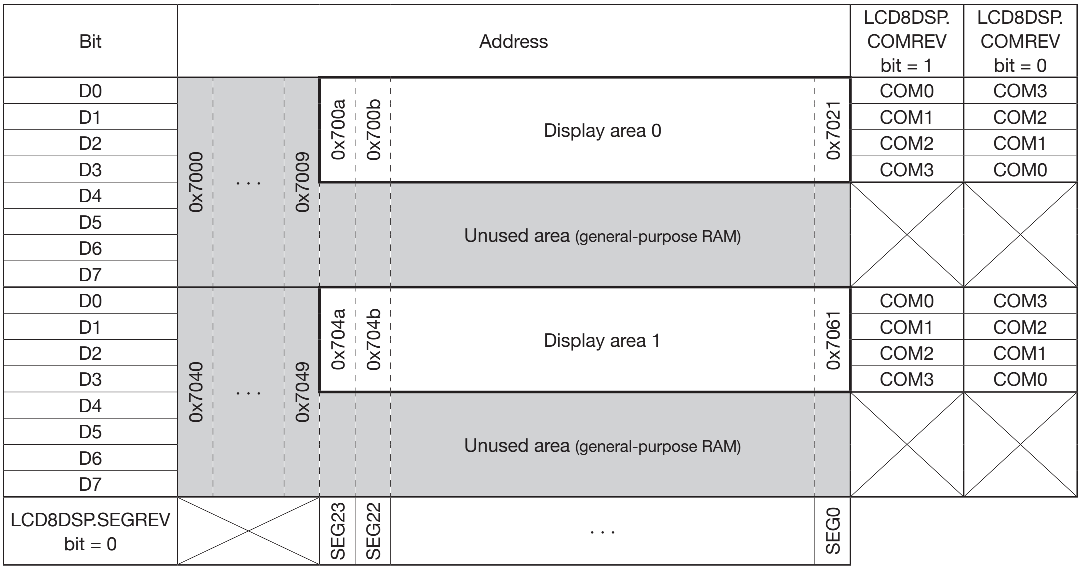
Bit	Address				LCD8DSP. COMREV bit = 1	LCD8DSP. COMREV bit = 0
D0	0x7000	0x7001	Display area 0	0x7021	COM0	COM3
D1					COM1	COM2
D2					COM2	COM1
D3					COM3	COM0
D4			Unused area (general-purpose RAM)			
D5						
D6						
D7						
D0	0x7040	0x7041	Display area 1	0x7061	COM0	COM3
D1					COM1	COM2
D2					COM2	COM1
D3					COM3	COM0
D4			Unused area (general-purpose RAM)			
D5						
D6						
D7						
LCD8DSP.SEGREV bit = 1	SEG0	SEG1	. . .	SEG33		
LCD8DSP.SEGREV bit = 0	SEG33	SEG32	. . .	SEG0		

* In the 80-pin package, addresses 0x701c to 0x701d and 0x705c to 0x705d (when the LCD8DSP.SEGREV bit = 1) or addresses 0x7004 to 0x7005 and 0x7044 to 0x7045 (when the LCD8DSP.SEGREV bit = 0) that are allocated to SEG28 to SEG29 are unused areas (general-purpose RAM).

(1) 100-pin package/80-pin package/chip

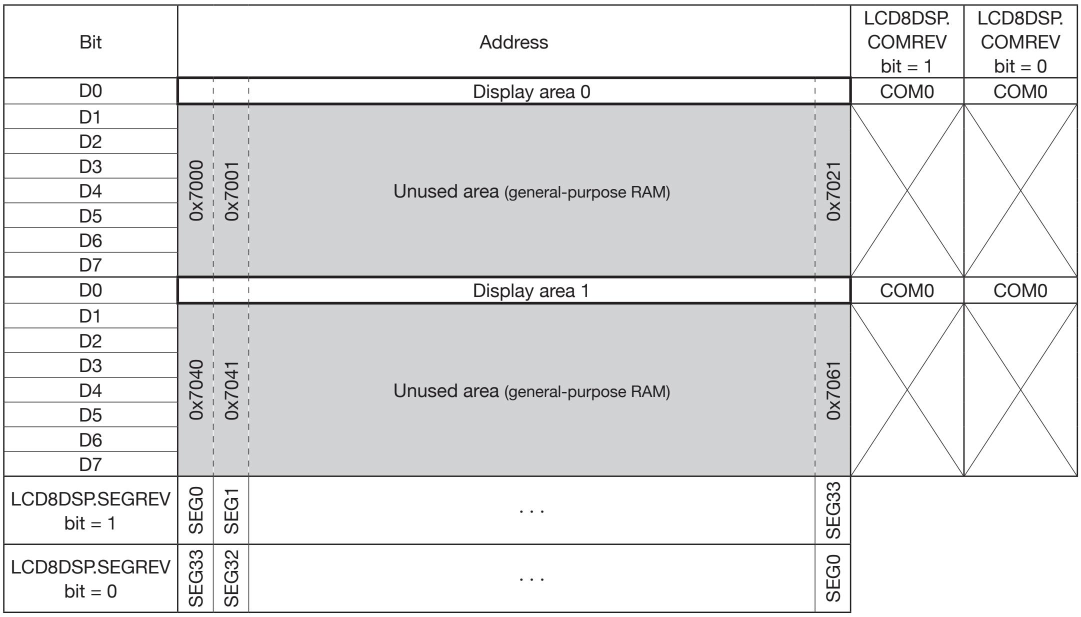
Bit	Address							LCD8DSP. COMREV bit = 1	LCD8DSP. COMREV bit = 0
D0	0x7000	0x7001	Display area 0	0x7017	0x7018	. . .	0x7021	COM0	COM3
D1								COM1	COM2
D2								COM2	COM1
D3								COM3	COM0
D4			Unused area (general-purpose RAM)						
D5									
D6									
D7									
D0	0x7040	0x7041	Display area 1	0x7057	0x7058	. . .	0x7061	COM0	COM3
D1								COM1	COM2
D2								COM2	COM1
D3								COM3	COM0
D4			Unused area (general-purpose RAM)						
D5									
D6									
D7									
LCD8DSP.SEGREV bit = 1	SEG0	SEG1	. . .	SEG23					

(2) 64-pin package (LCD8DSP.SEGREV bit = 1)



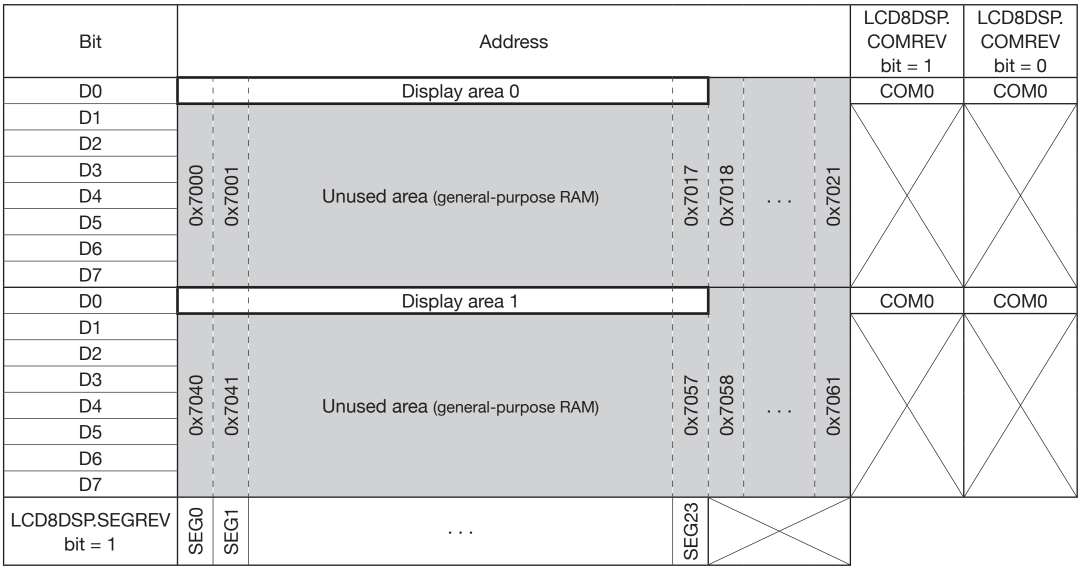
(3) 64-pin package (LCD8DSP.SEGREV bit = 0)

Figure 17.6.3.3 Display Data RAM Map (1/4 duty)

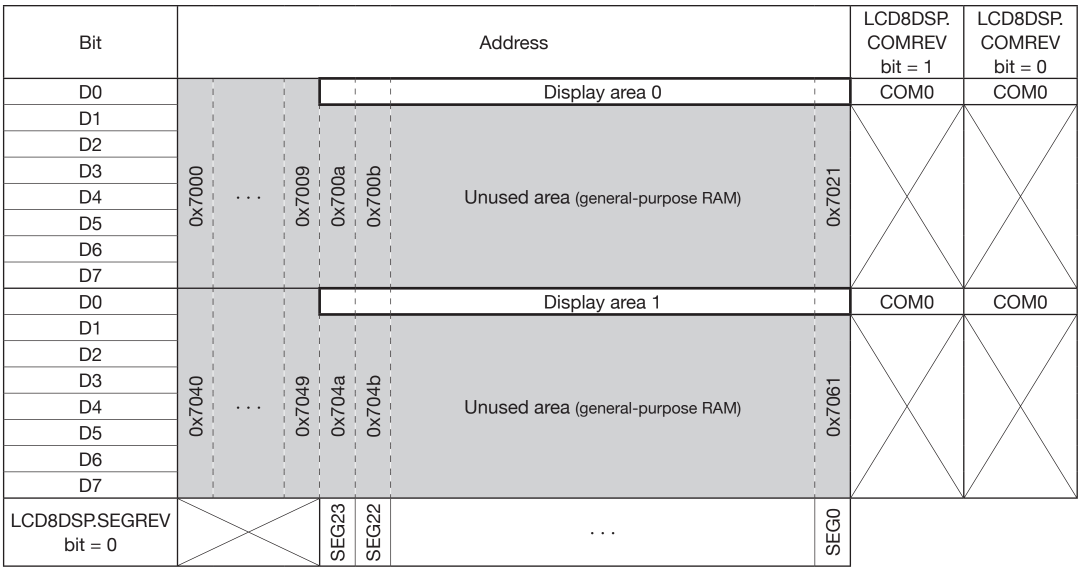


* In the 80-pin package, addresses 0x701c to 0x701d and 0x705c to 0x705d (when the LCD8DSP.SEGREV bit = 1) or addresses 0x7004 to 0x7005 and 0x7044 to 0x7045 (when the LCD8DSP.SEGREV bit = 0) that are allocated to SEG28 to SEG29 are unused areas (general-purpose RAM).

(1) 100-pin package/80-pin package/chip



(2) 64-pin package (LCD8DSP.SEGREV bit = 1)



(3) 64-pin package (LCD8DSP.SEGREV bit = 0)

Figure 17.6.3.4 Display Data RAM Map (static drive)

17.7 Interrupt

The LCD8B has a function to generate the interrupt shown in Table 17.7.1.

Table 17.7.1 LCD8B Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Frame	LCD8INTF.FRMIF	Frame switching	Writing 1

The LCD8B 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.

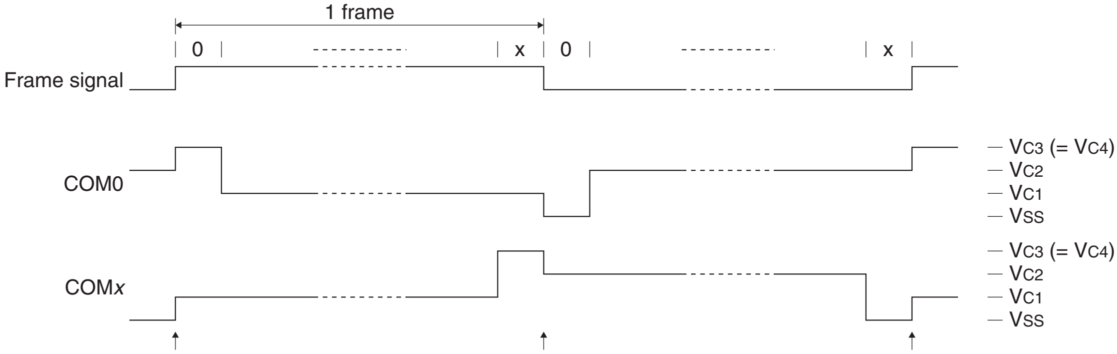


Figure 17.7.1 Frame Interrupt Timings (1/x duty, 1/3 bias)

17.8 Control Registers

LCD8B Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8CLK	15–9	–	0x00	–	R	–
	8	DBRUN	1	H0	R/W	
	7	–	0	–	R	
	6–4	CLKDIV[2:0]	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 LCD8B 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

Bit 7 Reserved

Bits 6–4 CLKDIV[2:0]

These bits select the division ratio of the LCD8B operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of the LCD8B.

Table 17.8.1 Clock Source and Division Ratio Settings

LCD8CLK.CLKDIV[2:0] bits	LCD8CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x7	Reserved	1/1	Reserved	1/1
0x6				
0x5	1/128		1/128	
0x4	1/64		1/64	
0x3	1/32		1/32	
0x2	1/16		1/16	
0x1	1/8		1/8	
0x0	1/4		1/4	

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

Note: The LCD8CLK register settings can be altered only when the LCD8CTL.MODEN bit = 0.

LCD8B Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8CTL	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 LCD8B operations.

1 (R/W): Enable LCD8B operations

0 (R/W): Disable LCD8B operations

Setting this bit to 1 starts supplying the operating clock to LCD8B and the SEG/COM pins are configured to output a low level. Setting this bit to 0 stops the operating clock and the SEG/COM pins are put into Hi-Z status.

Note: If the LCD8CTL.MODEN bit is altered from 1 to 0 while the LCD panel is displaying, the LCD display is automatically turned off and the LCD8DSP.DSPC[1:0] bits are also set to 0x0.

LCD8B Timing Control Register 1

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8TIM1	15–12	–	0x0	–	R	–
	11–8	FRMCNT[3:0]	0x3	H0	R/W	
	7–3	–	0x00	–	R	
	2–0	LDUTY[2:0]	0x7	H0	R/W	

Bits 15–12 Reserved

Bits 11–8 FRMCNT[3:0]

These bits set the frame frequency. For more information, refer to "Frame Frequency."

Bits 7–3 Reserved

Bits 2–0 LDUTY[2:0]

These bits set the drive duty. For more information, refer to "Drive Duty Switching."

LCD8B Timing Control Register 2

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8TIM2	15–10	–	0x00	–	R	–
	9–8	BSTC[1:0]	0x1	H0	R/W	
	7–3	–	0x00	–	R	
	2–0	NLINE[2:0]	0x0	H0	R/W	

Bits 15–10 Reserved

Bits 9–8 BSTC[1:0]

These bits select the booster clock frequency for the LCD voltage booster.

Table 17.8.2 Booster Clock Frequency

LCD8TIM2.BSTC[1:0] bits	Booster clock frequency [Hz]
0x3	$f_{CLK\_LCD8B}/64$
0x2	$f_{CLK\_LCD8B}/32$
0x1	$f_{CLK\_LCD8B}/16$
0x0	$f_{CLK\_LCD8B}/4$

f_{CLK_LCD8B}: LCD8B operating clock frequency [Hz]

Bits 7–3 Reserved

Bits 2–0 NLINE[2:0]

These bits enable the n-line inverse AC drive function and set the number of inverse lines. For more information, refer to "n-Segment-Line Inverse AC Drive."

LCD8B Power Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8PWR	15–13	–	0x0	–	R	–
	12–8	LC[4:0]	0x00	H0	R/W	
	7–5	–	0x0	–	R	
	4	BSTEN	0	H0	R/W	
	3	BISEL	1	H0	R/W	
	2	HVLD	0	H0	R/W	
	1	VCSEL	0	H0	R/W	
	0	VCEN	0	H0	R/W	

Bits 15–13 Reserved

Bits 12–8 LC[4:0]

These bits set the LCD panel contrast.

Table 17.8.3 LCD Contrast Adjustment

LCD8PWR.LC[4:0] bits	Contrast
0x1f	High (dark)
0x1e	↑
:	:
0x01	↓
0x00	Low (light)

Bits 7–5 Reserved

Bit 4 BSTEN

This bit turns the LCD voltage booster on and off.

1 (R/W): LCD voltage booster on
0 (R/W): LCD voltage booster off

For more information, refer to "LCD Power Supply."

Bit 3 BISEL

This bit selects the LCD drive bias.
1 (R/W): 1/4 bias
0 (R/W): 1/3 bias

Bit 2 HVLD

This bit sets the LCD voltage regulator into heavy load protection mode.
1 (R/W): Heavy load protection mode
0 (R/W): Normal mode

For more information, refer to "LCD Voltage Regulator Settings."

Bit 1 VCSEL

This bit sets the LCD voltage regulator output (reference voltage for boosting).
1 (R/W): V_{C2}
0 (R/W): V_{C1}

For more information, refer to "LCD Voltage Regulator Settings."

Note: The LCD8PWR.VCSEL bit must be set to 0 in an external voltage application mode.

Bit 0 VCEN

This bit turns the LCD voltage regulator on and off.
1 (R/W): LCD voltage regulator on
0 (R/W): LCD voltage regulator off

For more information, refer to "LCD Power Supply."

LCD8B Display Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8DSP	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	6	SEGREV	1	H0	R/W	
	5	COMREV	1	H0	R/W	
	4	DSPREV	1	H0	R/W	
	3	–	0	–	R	
	2	DSPAR	0	H0	R/W	
	1–0	DSPC[1:0]	0x0	H0	R/W	

Bits 15–7 Reserved

Bit 6 SEGREV

This bit selects the segment pin assignment direction.
1 (R/W): Normal assignment
0 (R/W): Inverse assignment

For more information, see Figures 17.6.3.1 to 17.6.3.4.

Bit 5 COMREV

This bit selects the common pin assignment direction.
1 (R/W): Normal assignment
0 (R/W): Inverse assignment

For more information, see Figures 17.6.3.1 to 17.6.3.4.

Note: Do not set the LCD8DSP.COMREV bit to 0 when the LCD8TIM1.LDUTY[2:0] bits = 0x4–0x6.

Bit 4 DSPREV

This bit controls black/white inversion on the LCD display.
1 (R/W): Normal display
0 (R/W): Inverted display

Bit 3 Reserved

Bit 2 DSPAR

This bit switches the display area in the display data RAM.

1 (R/W): Display area 1

0 (R/W): Display area 0

Bits 1–0 DSPC[1:0]

These bits control the LCD display on/off and select a display mode. For more information, refer to "Display On/Off."

LCD8B COM Pin Control Register 0

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8COMC0	15–8	–	0x00	–	R	–
	7	COM7DEN	1	H0	R/W	
	6	COM6DEN	1	H0	R/W	
	5	COM5DEN	1	H0	R/W	
	4	COM4DEN	1	H0	R/W	
	3	COM3DEN	1	H0	R/W	
	2	COM2DEN	1	H0	R/W	
	1	COM1DEN	1	H0	R/W	
	0	COM0DEN	1	H0	R/W	

Bits 15–8 Reserved

Bits 7–0 COM*x*DEN

These bits configure the partial drive of the COM*x* pins.

1 (R/W): Normal output

0 (R/W): Off waveform output

LCD8B Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8INTF	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	FRMIF	0	H0	R/W	Cleared by writing 1.

Bits 15–1 Reserved

Bit 0 FRMIF

This bit indicates the frame interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Clear flag

0 (W): Ineffective

LCD8B Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
LCD8INTE	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	FRMIE	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 FRMIE

This bit enables the frame interrupt.

1 (R/W): Enable interrupt

0 (R/W): Disable interrupt

18 R/F Converter (RFC)

18.1 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.
- Can generate reference oscillation completion, sensor (A and B) oscillation completion, measurement counter overflow error, and time base counter overflow error interrupts.

Figure 18.1.1 shows the RFC configuration.

Table 18.1.1 RFC Channel Configuration of S1C17W15

Item	S1C17W15
Number of channels	4 channels (Ch.0–Ch.3) ∗ Ch.1–Ch.3 can only be used in DC oscillation mode for resistive sensor measurements.

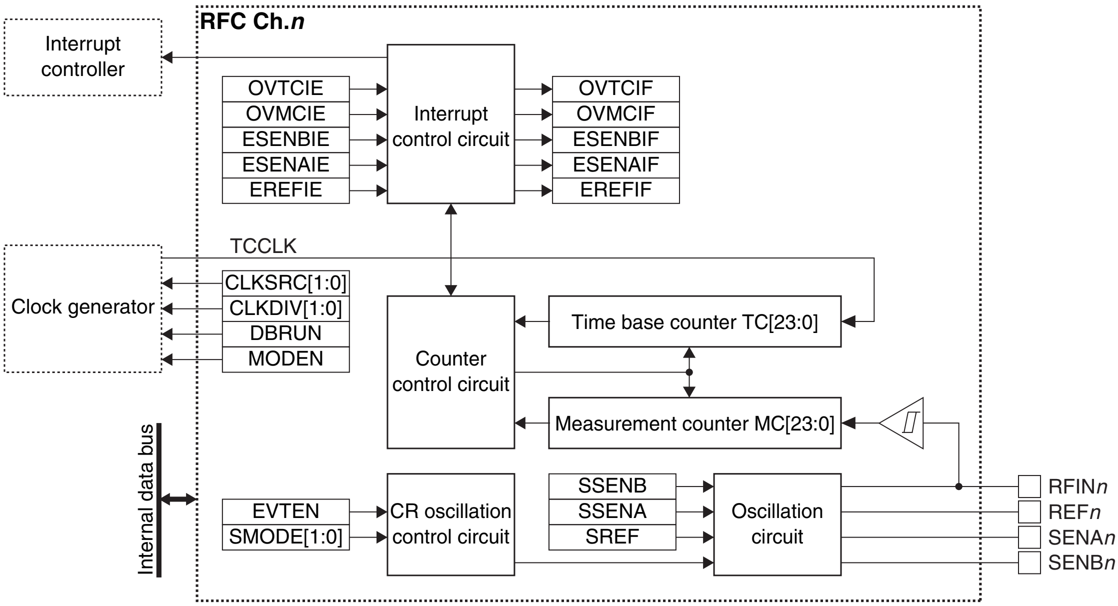


Figure 18.1.1 RFC Configuration

18.2 Input/Output Pins and External Connections

18.2.1 List of Input/Output Pins

Table 18.2.1.1 lists the RFC pins.

Table 18.2.1.1 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

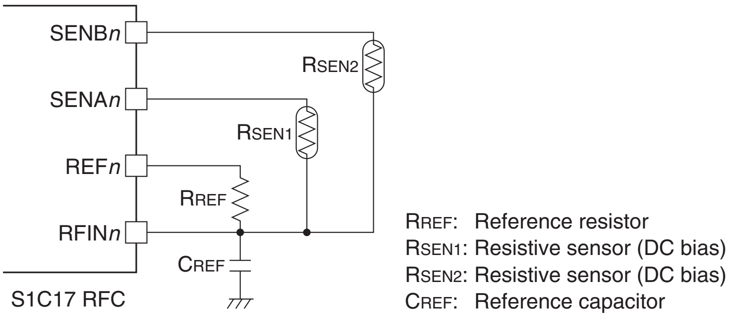
* Indicates the status when the pin is configured for the RFC.

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.

18.2.2 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."



* Leave the unused pin (SENA*n* or SENB*n*) open if one resistive sensor only is used.

Figure 18.2.2.1 Connection Example in Resistive Sensor DC Oscillation Mode

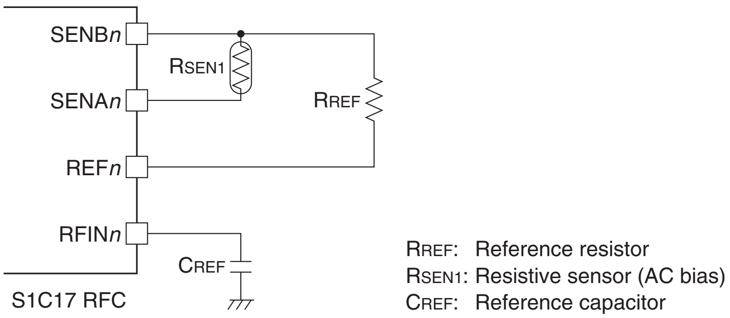
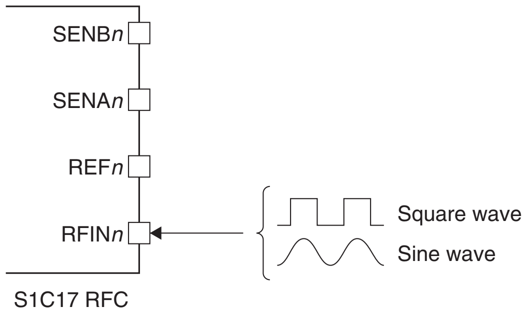


Figure 18.2.2.2 Connection Example in Resistive Sensor AC Oscillation Mode



∗ Leave the unused pins open.

Figure 18.2.2.3 External Clock Input in External Clock Input Mode

18.3 Clock Settings

18.3.1 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.

18.3.2 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.

18.3.3 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.

18.4 Operations

18.4.1 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)

18.4.2 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.

18.4.3 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.

18.4.4 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.

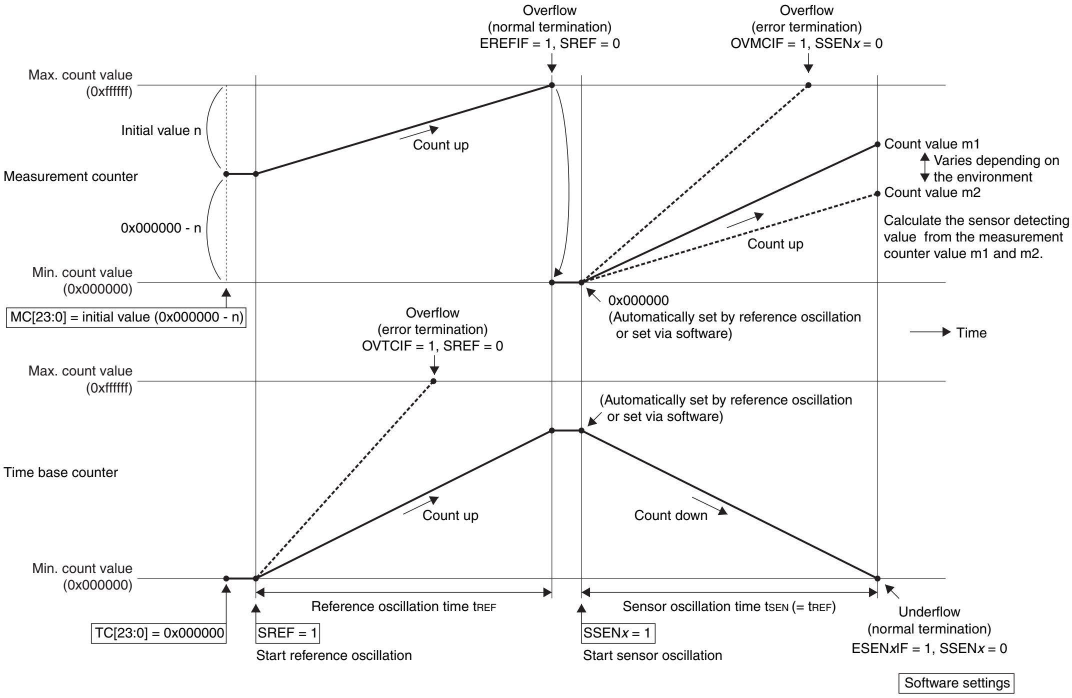


Figure 18.4.4.1 Counter Operations During Reference/Sensor Oscillation

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 18.4.4.1 lists the error factors. (n: measurement counter initial value, m: measurement counter value at the end of sensor oscillation)

Table 18.4.4.1 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

18.5 Interrupts

The RFC has a function to generate the interrupts shown in Table 18.5.1.

Table 18.5.1 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.

18.6 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 18.6.1 Clock Source and Division Ratio Settings

RFC*n*CLK. CLKDIV[1:0] bits	RFC*n*CLK.CLKSRC[1:0] bits			
	0x0 IOSC	0x1 OSC1	0x2 OSC3	0x3 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 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	15–8	–	0x00	–	R	–
	7	–	0	–	R	
	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 15–7 Reserved

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 18.6.2 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 RFC*n*MCH	31–24	–	0x00	–	R	–
	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 RFC*n*TCH	31–24	–	0x00	–	R	–
	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

19 Multiplier/Divider (COPRO2)

19.1 Overview

COPRO2 is the coprocessor that provides multiplier/divider functions. The features of COPRO2 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/unsigned 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.
  (32 bits ÷ 32 bits = 32 bits with 32-bit reminder)
  Can be executed in 17 to 20 cycles.
  Overflow detection and division by zero processing are not supported.

Figure 19.1.1 shows the COPRO2 configuration.

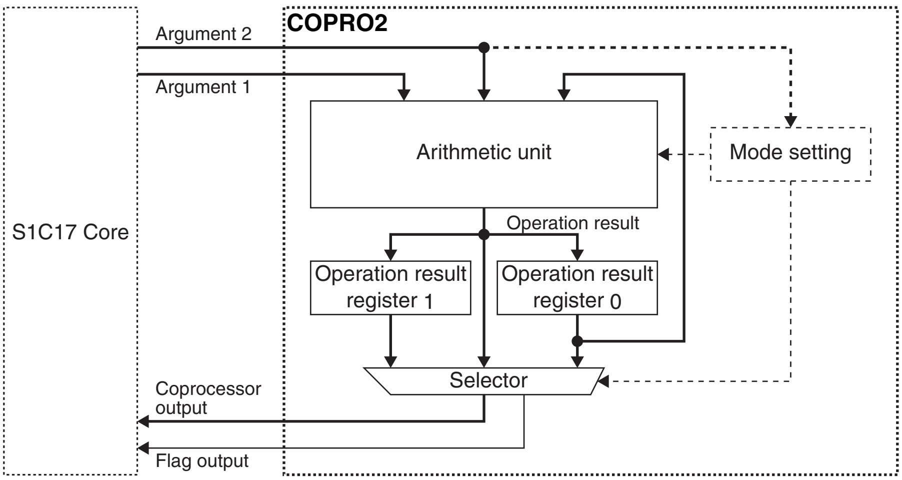


Figure 19.1.1 COPRO2 Configuration

19.2 Operation Mode and Output Mode

COPRO2 operates according to the operation mode specified by the application program. As listed in Table 19.2.1, COPRO2 supports 11 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 result register 0 or 1 to be read from COPRO2.

The operation and output modes can be specified with a 7-bit data by writing it to the mode setting register in COPRO2. 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 19.2.1 Mode Setting Register

Table 19.2.1 Mode Settings

Setting value (D[6:4])	Output mode	Setting value (D[3:0])	Operation mode
0x0	**16 low-order bits output mode 0** The low-order 16 bits of the operation result register 0 can be read as the coprocessor output.	0x0	**Initialize mode 0** Clears the operation result registers 0 and 1 to 0x0.
0x1	**16 high-order bits output mode 0** The high-order 16 bits of the operation result register 0 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 0.
0x2	**16 low-order bits output mode 1** The low-order 16 bits of the operation result register 1 can be read as the coprocessor output.	0x2	**Initialize mode 2** Loads the 32-bit data into the operation result register 0.
0x3	**16 high-order bits output mode 1** The high-order 16 bits of the operation result register 1 can be read as the coprocessor output.	0x3	**Operation result read mode** Outputs the data in the operation result registers 0 and 1 without computation.
0x4–0x7	Reserved	0x4	**Unsigned multiplication mode** Performs unsigned multiplication.
		0x5	**Signed multiplication mode** Performs signed multiplication.
		0x6	**Unsigned MAC mode** Performs unsigned MAC operation.
		0x7	**Signed MAC mode** Performs signed MAC operation.
		0x8	**Unsigned division mode** Performs unsigned division.
		0x9	**Signed division mode** Performs signed division.
		0xa	**Initialize mode 3** Loads the 32-bit data into the operation result register 1.
		0xb–0xf	Reserved

19.3 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 0) or 0x05 (signed multiplication, 16 low-order bits output mode 0).
2. Send the 16-bit multiplicand (B) and 16-bit multiplier (C) to COPRO2 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 0).
5. Read another one-half result (16 high-order bits = A[31:16]).

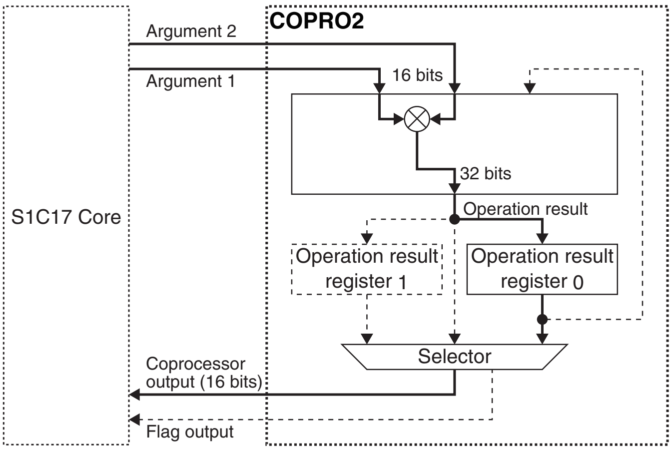


Figure 19.3.1 Data Path in Multiplication Mode

Table 19.3.1 Operation in Multiplication Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x04 or 0x05	`ld.ca %rd,%rs`	res0[31:0] ← %rd × %rs %rd ← res0[15:0]	psr (CVZN) ← 0b0000	The operation result register 0 keeps the operation result until it is rewritten by other operation.
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res0[31:0] ← %rd × *imm7/16* %rd ← res0[15:0]		
0x14 or 0x15	`ld.ca %rd,%rs`	res0[31:0] ← %rd × %rs %rd ← res0[31:16]		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res0[31:0] ← %rd × *imm7/16* %rd ← res0[31:16]		

res0: operation result register 0

Example:

```
ld.cw %r0,0x04 ; Sets the mode (unsigned multiplication mode and 16 low-order bits output mode 0).
ld.ca %r0,%r1  ; Performs "res0[31:0] = %r0[15:0] × %r1[15:0]" 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 0).
ld.ca %r1,%r0  ; Loads the 16 high-order bits of the result to %r1.
```

19.4 Division

The division function performs “A (32 bits) = B (32 bits) ÷ C (32 bits), D (32 bits) = remainder.”
The following shows a procedure to perform a division:

1. Set the mode to 0x02 (initialize mode 2).
2 Set the 32-bit dividend (B) to the operation result register 0 using a “`ld.cf`” instruction.
3. Set the mode to 0x08 (unsigned division, 16 low-order bits output mode 0) or 0x09 (signed division, 16 low-order bits output mode 0).
4. Send the 32-bit divisor (C) to COPRO2 using a “`ld.ca`” instruction.
5. Read the one-half result (16 low-order bits = A[15:0]) of the operation result register 0 (quotient) and the flag status.
6. Set the mode to 0x13 (operation result read, 16 high-order bits output mode 0).
7. Read another one-half result (16 high-order bits = A[31:16]) of the operation result register 0 (quotient).
8. Set the mode to 0x23 (operation result read, 16 low-order bits output mode 1).
9. Read the one-half result (16 low-order bits = D[15:0]) of the operation result register 1 (remainder).
10. Set the mode to 0x33 (operation result read, 16 high-order bits output mode 1).
11. Read another one-half result (16 high-order bits = D[31:16]) of the operation result register 1 (remainder).

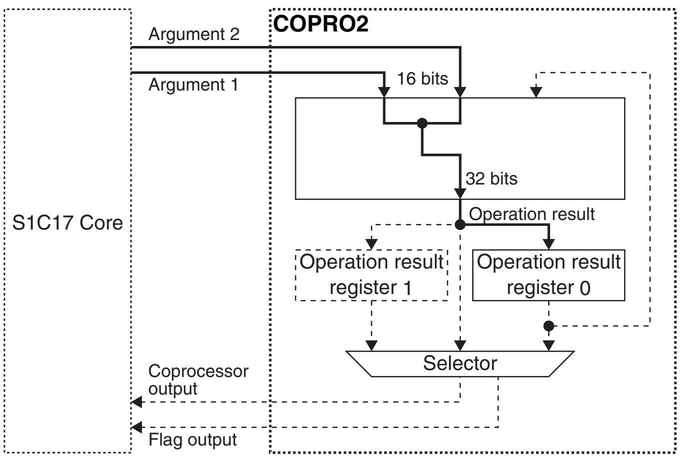


Figure 19.4.1 Data Path in Initialize Mode 2

Table 19.4.1 Initializing the Operation Result Register 0 (32 bits)

Mode setting value	Instruction	Operations	Remarks
0x02	`ld.cf %rd,%rs`	res0[31:16] ← %rd res0[15:0] ← %rs	
	(`ext` *`imm9`*) `ld.cf %rd,`*`imm7`*	res0[31:16] ← %rd res0[15:0] ← *imm7/16*	

res0: operation result register 0

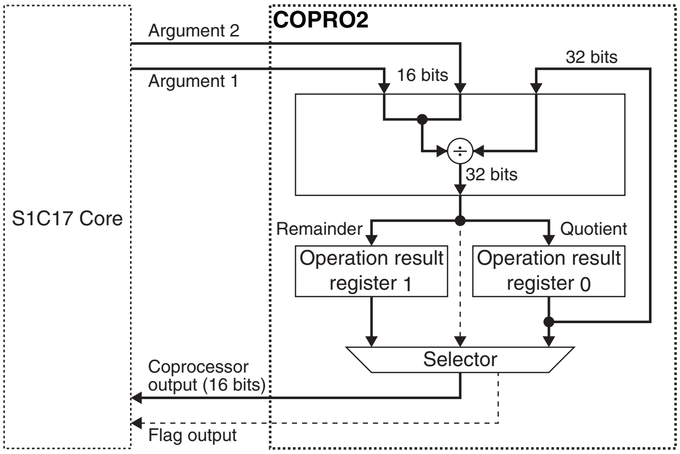


Figure 19.4.2 Data Path in Division Mode

Table 19.4.2 Operation in Division Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x08 or 0x09	`ld.ca %rd,%rs`	res0[31:0] ÷ {%rd, %rs} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res0[15:0] (Quotient)	psr (CVZN) ← 0b0000	The operation result registers 0 and 1 keep the operation results until they are rewritten by other operation. COPRO2 does not support 0 ÷ 0 division.
	(`ext` *`imm9`*) `ld.ca %rd,`*`imm7`*	res0[31:0] ÷ {%rd, *imm7/16*} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res0[15:0] (Quotient)		
0x18 or 0x19	`ld.ca %rd,%rs`	res0[31:0] ÷ {%rd, %rs} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res0[31:16] (Quotient)		
	(`ext` *`imm9`*) `ld.ca %rd,`*`imm7`*	res0[31:0] ÷ {%rd, *imm7/16*} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res0[31:16] (Quotient)		
0x28 or 0x29	`ld.ca %rd,%rs`	res0[31:0] ÷ {%rd, %rs} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res1[15:0] (Remainder)		
	(`ext` *`imm9`*) `ld.ca %rd,`*`imm7`*	res0[31:0] ÷ {%rd, *imm7/16*} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res1[15:0] (Remainder)		
0x38 or 0x39	`ld.ca %rd,%rs`	res0[31:0] ÷ {%rd, %rs} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res1[31:16] (Remainder)		
	(`ext` *`imm9`*) `ld.ca %rd,`*`imm7`*	res0[31:0] ÷ {%rd, *imm7/16*} res0[31:0] ← Quotient res1[31:0] ← Remainder %rd ← res1[31:16] (Remainder)		

res0: operation result register 0, res1: operation result register 1

Example:

```
ld.cw %r0,0x02 ; Sets the mode (initialize mode 2).
ld.cf %r0,%r1  ; Set the dividend {%r0, %r1} to the operation result register 0.
ld.cw %r0,0x08 ; Sets the mode (unsigned division mode and 16 low-order bits output mode 0).
ld.ca %r0,%r1  ; Performs "res0[31:0] (quotient), res1[31:0] (remainder) = res0[31:0] ÷ {%r0[15:0],
               ; %r1[15:0]}" and loads the 16 low-order bits of the result (quotient) to %r0.
ld.ca %r1,%r0  ; Loads the 16 low-order bits of the result (quotient) to %r1.
ld.cw %r0,0x13 ; Sets the mode (operation result read mode and 16 high-order bits output mode 0).
ld.ca %r2,%r0  ; Loads the 16 high-order bits of the result (quotient) to %r2.
ld.cw %r0,0x23 ; Sets the mode (operation result read mode and 16 low-order bits output mode 1).
ld.ca %r3,%r0  ; Loads the 16 low-order bits of the result (remainder) to %r3.
ld.cw %r0,0x33 ; Sets the mode (operation result read mode and 16 high-order bits output mode 1).
ld.ca %r4,%r0  ; Loads the 16 high-order bits of the result (remainder) to %r4.
```

19.5 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 0.
   - To clear the operation result registers (A = 0):
     Set the mode to 0x00 (initialize mode 0). (It is not necessary to send 0x00 to COPRO2 with another instruction.)
   - To load a 16-bit value to the operation result register 0:
     Set the operation mode to 0x01 (initialize mode 1) and then send the initial value (16 bits) to COPRO2 using a "`ld.cf`" instruction.
   - To load a 32-bit value to the operation result register 0:
     Set the operation mode to 0x02 (initialize mode 2) and then send the initial value (32 bits) to COPRO2 using a "`ld.cf`" instruction.
2. Set the mode to 0x06 (unsigned MAC, 16 low-order bits output mode 0) or 0x07 (signed MAC, 16 low-order bits output mode 0).
3. Repeat sending the 16-bit multiplicand (B) and 16-bit multiplier (C) to COPRO2 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]).

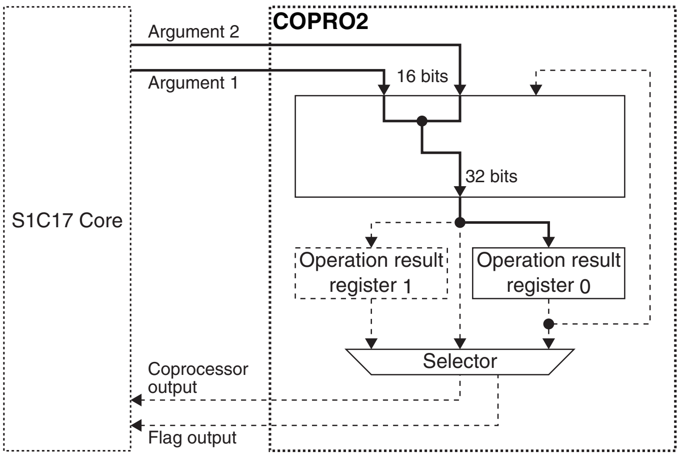


Figure 19.5.1 Data Path in Initialize Mode

Table 19.5.1 Initializing the Operation Result Register 0

Mode setting value	Instruction	Operations	Remarks
0x00	–	res0[31:0] ← 0x0 res1[31:0] ← 0x0	Setting the operating mode executes the initialization without sending data.
0x01	`ld.cf %rd,%rs`	res0[31:16] ← 0x0 res0[15:0] ← %rs	
	(`ext` *imm9*) `ld.cf %rd,`*imm7*	res0[31:16] ← 0x0 res0[15:0] ← *imm7/16*	
0x02	`ld.cf %rd,%rs`	res0[31:16] ← %rd res0[15:0] ← %rs	
	(`ext` *imm9*) `ld.cf %rd,`*imm7*	res0[31:16] ← %rd res0[15:0] ← *imm7/16*	

res0: operation result register 0, res1: operation result register 1

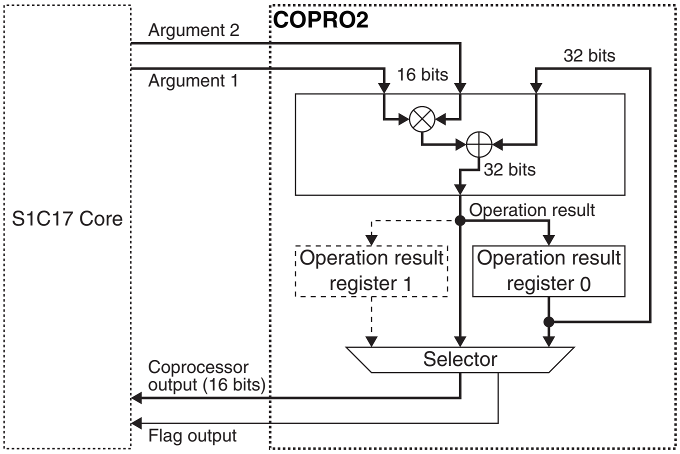


Figure 19.5.2 Data Path in MAC Mode

Table 19.5.2 Operation in MAC Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x06 or 0x07	`ld.ca %rd,%rs`	res0[31:0] ← %rd × %rs + res0[31:0] %rd ← res0[15:0]	psr (CVZN) ← 0b0100 if an overflow has occurred Otherwise psr (CVZN) ← 0b0000	The operation result register 0 keeps the operation result until it is rewritten by other operation. Overflow can be detected only in signed MAC mode (it does not occur in unsigned MAC mode).
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res0[31:0] ← %rd × *imm7/16* + res0[31:0] %rd ← res0[15:0]		
0x16 or 0x17	`ld.ca %rd,%rs`	res0[31:0] ← %rd × %rs + res0[31:0] %rd ← res0[31:16]		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res0[31:0] ← %rd × *imm7/16* + res0[31:0] %rd ← res0[31:16]		

res0: operation result register 0

Example:

```
ld.cw %r0,0x00 ; Sets the mode (initialize mode 0) to clear the operation result register 0 to 0x0000.
ld.cw %r0,0x07 ; Sets the mode (signed MAC mode and 16 low-order bits output mode 0).
ld.ca %r0,%r1  ; Performs "res0[31:0] = %r0[15:0] × %r1[15:0] + res0[31:0]" 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 0).
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 signed 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 19.5.3 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.

19.6 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 COPRO2 into operation result read mode. The operation result register keeps the loaded operation result until it is rewritten by other operation.

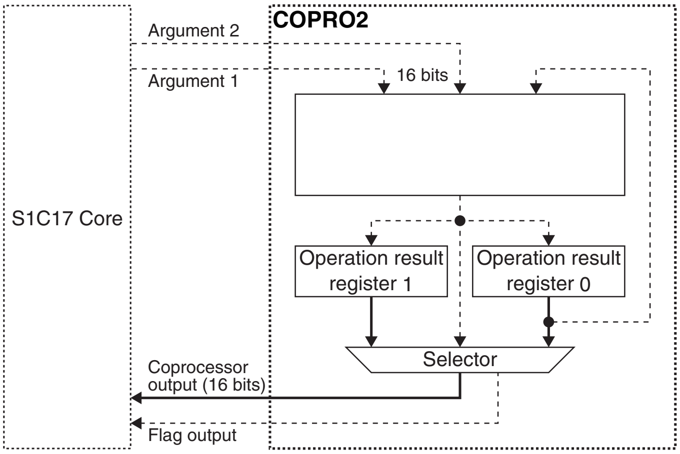


Figure 19.6.1 Data Path in Operation Result Read Mode

Table 19.6.1 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 registers 0 and 1.
	`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]		
0x23	`ld.ca %rd,%rs`	%rd ← res1[15:0]		
	`ld.ca %rd,`*imm7*	%rd ← res1[15:0]		
0x33	`ld.ca %rd,%rs`	%rd ← res1[31:16]		
	`ld.ca %rd,`*imm7*	%rd ← res1[31:16]		

res0: operation result register 0, res1: operation result register 1

20 Electrical Characteristics

20.1 Absolute Maximum Ratings

(V_{SS} = 0 V)

Item	Symbol	Condition		Rated value	Unit
Power supply voltage	V_{DD}			-0.3 to 4.0	V
Flash programming voltage	V_{PP}			-0.3 to 8.0	V
LCD power supply voltage	V_{C1}			-0.3 to 7.0	V
	V_{C2}			-0.3 to 7.0	V
	V_{C3}			-0.3 to 7.0	V
	V_{C4}			-0.3 to 7.0	V
Input voltage	V_I	P00–03, PD3–D4, #RESET		-0.3 to V_{DD} + 0.5	V
		P04–06, P10–17, P20–27, P30–37, PD0–D1		-0.3 to 4.0	V
Output voltage	V_O	P00–P06, P10–17, P20–27, P30–37, PD0–D4		-0.3 to V_{DD} + 0.5	V
High level output current	I_{OH}	1 pin	P00–P06, P10–17, P20–27, P30–37, PD0–D4	-10	mA
		Total of all pins		-20	mA
Low level output current	I_{OL}	1 pin	P00–P06, P10–17, P20–27, P30–37, PD0–D4	10	mA
		Total of all pins		20	mA
Operating temperature	Ta			-40 to 85	°C
Storage temperature	Tstg			-65 to 125	°C

20.2 Recommended Operating Conditions

(V_{SS} = 0 V) ∗1

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
Power supply voltage	V_{DD}	For normal operation		1.2	–	3.6	V
		For Flash programming		1.8	–	3.6	V
		For super economy mode		2.5	–	3.6	V
Flash programming voltage	V_{PP}			7.3	7.5	7.7	V
LCD power supply voltage (1/3 bias)	V_{C1}	When an external voltage is applied $V_{C1} \leq V_{C2} \leq V_{C3}$ (= V_{C4}), $V_{C1} \leq V_{DD}$		–	1.0	1.8	V
	V_{C2}			–	2.0	3.6	V
	V_{C3}/V_{C4}			–	3.0	5.4	V
LCD power supply voltage (1/4 bias)	V_{C1}	When an external voltage is applied $V_{C1} \leq V_{C2} \leq V_{C3} \leq V_{C4}$, $V_{C1} \leq V_{DD}$		–	1.0	1.4	V
	V_{C2}			–	2.0	2.8	V
	V_{C3}			–	3.0	4.2	V
	V_{C4}			–	4.0	5.6	V
OSC1 oscillator oscillation frequency	f_{OSC1}	Crystal oscillator		–	32.768	–	kHz
OSC3 oscillator oscillation frequency	f_{OSC3}	Internal oscillator or crystal/ ceramic oscillator	V_{DD} = 1.2 to 1.6 V	0.5	–	1.1	MHz
			V_{DD} = 1.6 to 3.6 V	0.5	–	4.2	MHz
		CR oscillator	V_{DD} = 1.6 to 3.6 V	0.1	–	2.1	MHz
EXOSC external clock frequency	f_{EXOSC}	When supplied from an external oscillator	V_{DD} = 1.2 to 1.6 V	0.016	–	1.1	MHz
			V_{DD} = 1.6 to 3.6 V	0.016	–	4.2	MHz
Bypass capacitor between V_{SS} and V_{DD}	C_{PW1}			–	3.3	–	μF
Capacitors between V_{SS} and V_{D1-2}	C_{PW2-3}			–	1	–	μF
Capacitor between C_{V1} and C_{V2}	C_{CV}	∗2		–	1	10	μF
Capacitors between V_{SS} and V_{C1-3}	C_{LCD1-3}	∗3		–	1	–	μF
Capacitor between V_{SS} and V_{C4}	C_{LCD4}	∗3, ∗4		–	1	–	μF
Capacitor between C_{P1} and C_{P2}	C_{LCD5}	∗3		–	1	–	μF
Capacitor between C_{P3} and C_{P4}	C_{LCD6}	∗3, ∗4		–	1	–	μF
Gate capacitor for OSC1 oscillator	C_{G1}	∗5		0	–	25	pF
Drain capacitor for OSC1 oscillator	C_{D1}	∗5		–	0	–	pF
Gate capacitor for OSC3 oscillator	C_{G3}	When the crystal/ceramic oscillator is used ∗5		0	–	100	pF
Drain capacitor for OSC3 oscillator	C_{D3}	When the crystal/ceramic oscillator is used ∗5		0	–	100	pF
Oscillation resistor for OSC3 oscillator	R_{CR3}	When the CR oscillator is used		10	–	1,000	kΩ
DSIO pull-up resistor	R_{DBG}	∗6		–	10	–	kΩ
Capacitor between V_{SS} and V_{PP}	C_{VPP}			–	0.1	–	μF

∗1 The potential variation of the V_{SS} voltage should be suppressed to within ±0.3 V on the basis of the ground potential of the MCU mounting board while the Flash is being programmed, as it affects the Flash memory characteristics (programming count).

∗2 The C_{V1}–C_{V2} pins can be left open when super economy mode is not used.

∗3 The V_{C1}–V_{C4} and C_{P1}–C_{P4} pins can be left open when the LCD driver is not used.

∗4 Connect between the V_{C3} and V_{C4} pins when the LCD power supply circuit is configured for 1/3 bias. Also the C_{P3}–C_{P4} pins can be left open.

∗5 The component values should be determined after performing matching evaluation of the resonator mounted on the printed circuit board actually used.

∗6 R_{DBG} is not required when using the DSIO pin as a general-purpose I/O port.

∗7 The component values should be determined after evaluating operations using an actual mounting board.

20.3 Current Consumption

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C, EXOSC = OFF, PWGCTL.PWGMOD[2:0] bits = 0x0 (automatic mode), PWGTIM.DCCCLK[1:0] bits = 0x0 (OSC1/32), FLASHCWAIT.RDWAIT[1:0] bits = 0x1 (2 cycles)

Item	Symbol	Condition	V_{DD} or Ta	Min.	Typ.	Max.	Unit
Current consumption in SLEEP mode	I_{SLP}	OSC1 = OFF, IOSC = OFF, OSC3 = OFF	25 °C	–	0.15	0.50	µA
			85 °C	–	1.6	9.0	µA
Current consumption in HALT mode	I_{HALT1}	IOSC = ON, OSC1 = 32 kHz*1, OSC3 = OFF		–	30	50	µA
	I_{HALT2}	IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = OFF		–	0.5	1.1	µA
		IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = OFF, PWGCTL.PWGMOD[2:0] bits = 0x5 (super economy mode)	2.5 to 3.6 V	–	0.3	0.65	µA
	I_{HALT3}	IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = 1 MHz (ceramic oscillator)*2		–	30	45	µA
Current consumption in RUN mode	I_{RUN10}*5	IOSC = ON, OSC1 = 32 kHz*1, OSC3 = OFF, SYSCLK = IOSC		–	180	220	µA
		IOSC = ON, OSC1 = 32 kHz*1, OSC3 = OFF, SYSCLK = IOSC FLASHCWAIT.RDWAIT[1:0] bits = 0x0 (1 cycle)		–	280	350	µA
	I_{RUN20}*5	IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = OFF, SYSCLK = OSC1		–	8	12	µA
		IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = OFF, SYSCLK = OSC1 PWGCTL.PWGMOD[2:0] bits = 0x2 (normal mode)		–	17	22	µA
		IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = OFF, SYSCLK = OSC1, PWGCTL.PWGMOD[2:0] bits = 0x5 (super economy mode)	2.5 to 3.6 V	–	4	6	µA
	I_{RUN30}*5	IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = 1 MHz (ceramic oscillator)*2, SYSCLK = OSC3	1.6 to 3.6 V	–	250	300	µA
		IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = 1 MHz (CR oscillator)*3, SYSCLK = OSC3		–	250	350	µA
		IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = 4 MHz (internal oscillator)*4, SYSCLK = OSC3		–	930	1,100	µA
	I_{RUN11}*6	IOSC = ON, OSC1 = 32 kHz*1, SYSCLK = IOSC, running in the RAM		–	110	150	µA
	I_{RUN21}*6	IOSC = OFF, OSC1 = 32 kHz*1, SYSCLK = OSC1, running in the RAM		–	4.8	7	µA
	I_{RUN31}*6	IOSC = OFF, OSC1 = 32 kHz*1, OSC3 = 1 MHz (ceramic oscillator)*2, SYSCLK = OSC3, running in the RAM	1.6 to 3.6 V	–	150	200	µA

*1 OSC1 oscillator: CLGOSC1.INV1N[1:0] bits = 0x0, CLGOSC1.CGI1[2:0] bits = 0x0, CLGOSC1.OSDEN bit = 0, C_{G1} = C_{D1} = 0 pF, Crystal resonator = C-002RX (manufactured by Seiko Epson Corporation, R_1 = 50 kΩ (Max.), C_L = 7 pF)

*2 OSC3 oscillator: CLGOSC3.OSC3MD[1:0] bits = 0x2, CLGOSC3.OSC3INV[1:0] bits = 0x0, C_{G3} = C_{D3} = 100 pF, ceramic resonator = CSBLA_J (manufactured by Murata Manufacturing Co., Ltd., 1 MHz)

*3 OSC3 oscillator: CLGOSC3.OSC3MD[1:0] bits = 0x1, R_{CR3} = 68 kΩ

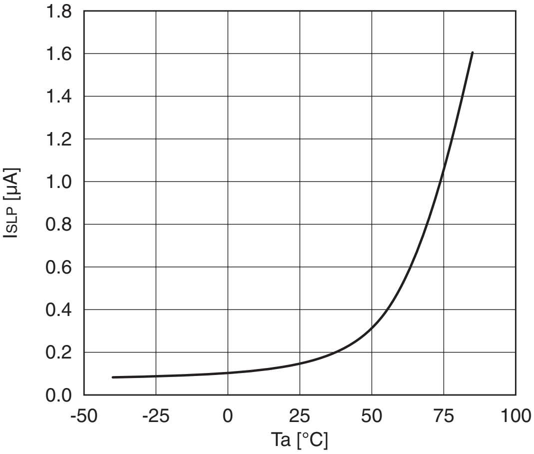
*4 OSC3 oscillator: CLGOSC3.OSC3MD[1:0] bits = 0x0, CLGOSC3.OSC3FQ[1:0] bits = 0x3

*5 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.

*6 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.

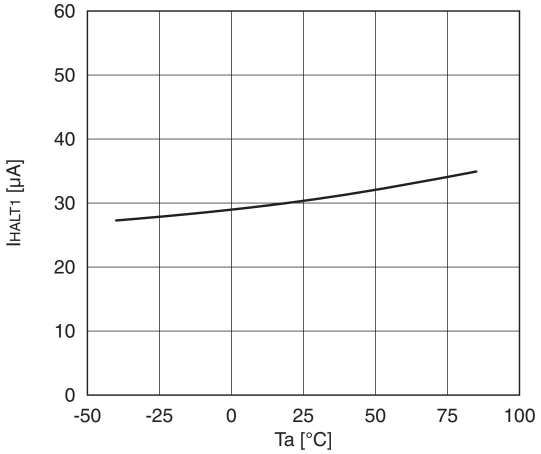
Current consumption-temperature characteristic in SLEEP mode

IOSC = OFF, OSC1 = OFF, OSC3 = OFF, Typ. value



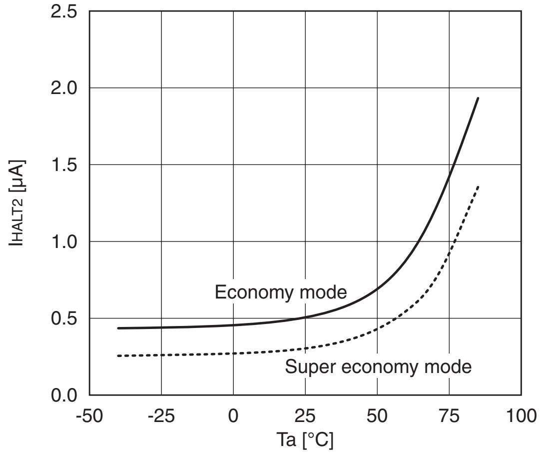
Current consumption-temperature characteristic in HALT mode (IOSC operation)

IOSC = ON, OSC1 = 32 kHz, OSC3 = OFF, Typ. value



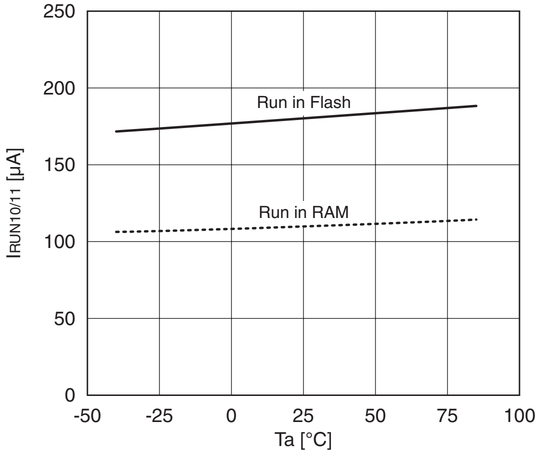
Current consumption-temperature characteristic in HALT mode (OSC1 operation)

IOSC = OFF, OSC1 = 32 kHz, OSC3 = OFF, Typ. value



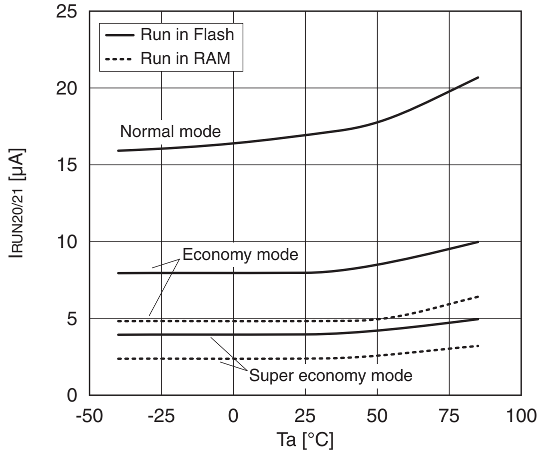
Current consumption-temperature characteristic in RUN mode (IOSC operation)

IOSC = ON, OSC1 = 32 kHz, OSC3 = OFF, Typ. value



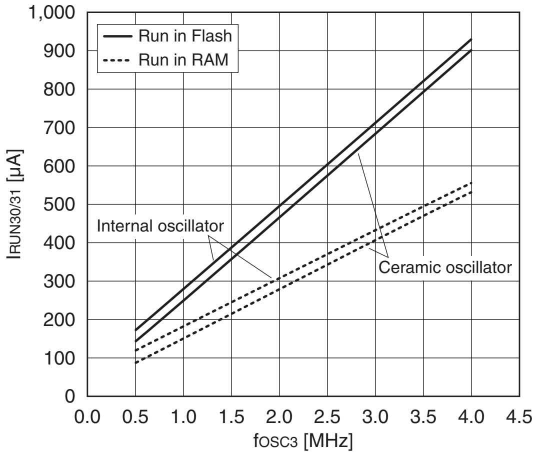
Current consumption-temperature characteristic in RUN mode (OSC1 operation)

IOSC = OFF, OSC1 = 32 kHz, OSC3 = OFF, Typ. value



Current consumption-frequency characteristic in RUN mode (OSC3 operation)

IOSC = OFF, OSC1 = 32 kHz, OSC3 = ON, Ta = 25 °C, Typ. value

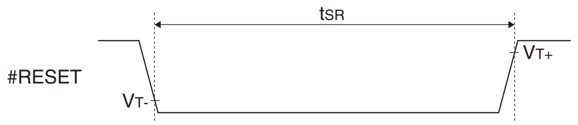


20.4 System Reset Controller (SRC) Characteristics

#RESET pin characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

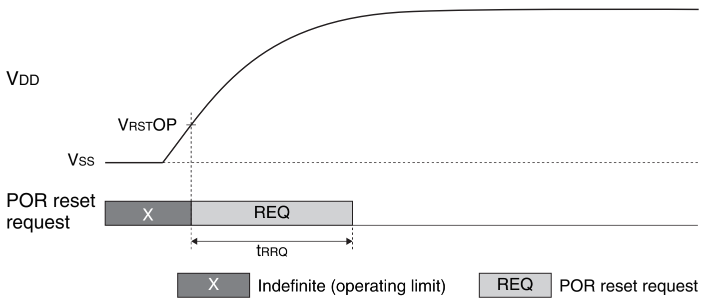
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
High level Schmitt input threshold voltage	V_{T+}		0.5 × V_{DD}	–	0.8 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}		0.2 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T		20	–	–	mV
Input pull-up resistance	R_{IN}		100	270	500	kΩ
Pin capacitance	C_{IN}		–	–	15	pF
Reset Low pulse width	t_{SR}		5	–	–	µs



POR characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
POR operating limit voltage	V_{RSTOP}		–	0.5	0.95	V
POR 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 V_{RSTOP} or less.

Reset hold circuit characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Reset hold time*1	t_{RSTR}		0.5	–	0.9	ms

*1 Time until the internal reset signal is negated after the reset request is canceled.

20.5 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.

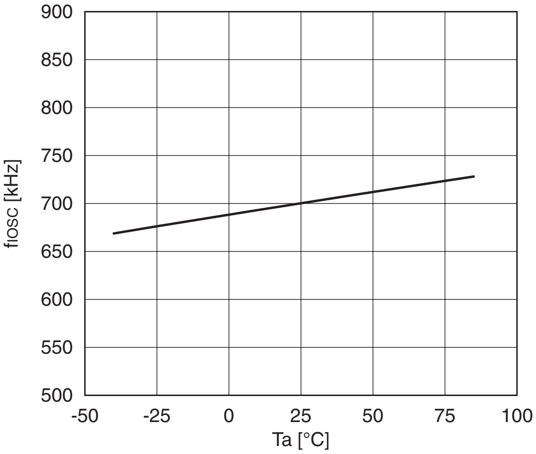
IOSC oscillator circuit characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	V_{DD}	Ta	Min.	Typ.	Max.	Unit
Oscillation start time	t_{staI}				–	–	3	µs
Oscillation frequency	f_{IOSC}		1.6 to 3.6 V	25 °C	679	700	721	kHz
			1.2 to 1.6 V		665	700	735	kHz
			1.6 to 3.6 V	-40 to 85 °C	651	700	749	kHz
			1.2 to 1.6 V		630	700	770	kHz

IOSC oscillation frequency-temperature characteristic

VDD = 1.6 to 3.6 V, Typ. value



OSC1 oscillator circuit characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time*1	t_{sta1}	CLGOSC1.INV1N[1:0] bits = 0x1, CLGOSC1.INV1B[1:0] bits = 0x2, CLGOSC1.OSC1BUP bit = 1	–	–	3	s
Internal gate capacitance	C_{GI1}	CLGOSC1.CGI1[2:0] bits = 0x0	–	12	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x1	–	14	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x2	–	16	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x3	–	18	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x4	–	19	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x5	–	21	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x6	–	23	–	pF
		CLGOSC1.CGI1[2:0] bits = 0x7	–	24	–	pF
Internal drain capacitance	C_{DI1}		–	6	–	pF
Oscillator circuit current - oscillation inverter drivability ratio *1	I_{OSC1}	CLGOSC1.INV1N/INV1B[1:0] bits = 0x0	–	70	–	%
		CLGOSC1.INV1N/INV1B[1:0] bits = 0x1 (reference)	–	100	–	%
		CLGOSC1.INV1N/INV1B[1:0] bits = 0x2	–	130	–	%
		CLGOSC1.INV1N/INV1B[1:0] bits = 0x3	–	300	–	%
Oscillation stop detector current	I_{OSD1}	CLGOSC1.OSDEN bit = 1	–	0.025	0.1	µA

*1 CLGOSC1.CGI1[2:0] bits = 0x0, Crystal resonator = C-002RX (manufactured by Seiko Epson Corporation, R_1 = 50 kΩ (Max.), C_L = 7 pF)

OSC3 oscillator circuit characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C

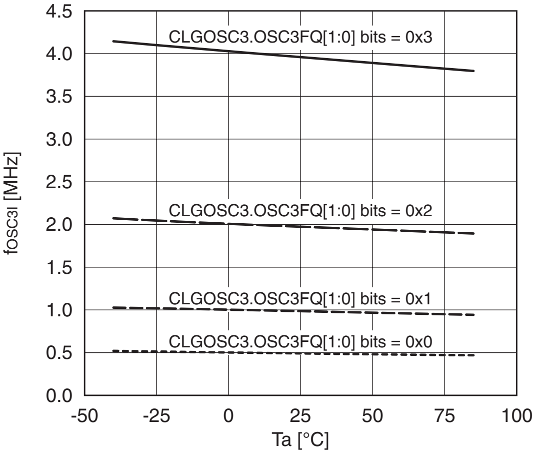
Item	Symbol	Condition	VDD	Min.	Typ.	Max.	Unit
Internal oscillator oscillation start time	t_{sta3I}	CLGOSC3.OSC3MD[1:0] bits = 0x0		–	–	3	µs
Internal oscillator oscillation frequency	f_{OSC3I}	CLGOSC3.OSC3MD[1:0] bits = 0x0, CLGOSC3.OSC3FQ[1:0] bits = 0x3	1.6 to 3.6 V	3.42	4.00	4.18	MHz
		CLGOSC3.OSC3MD[1:0] bits = 0x0, CLGOSC3.OSC3FQ[1:0] bits = 0x2	1.6 to 3.6 V	1.71	2.00	2.09	MHz
		CLGOSC3.OSC3MD[1:0] bits = 0x0, CLGOSC3.OSC3FQ[1:0] bits = 0x1	1.6 to 3.6 V	0.86	1.00	1.05	MHz
			1.2 to 1.6 V	–	1.00	–	MHz
		CLGOSC3.OSC3MD[1:0] bits = 0x0, CLGOSC3.OSC3FQ[1:0] bits = 0x0	1.6 to 3.6 V	0.43	0.50	0.53	MHz
			1.2 to 1.6 V	–	0.50	–	MHz
CR oscillator oscillation start time	t_{sta3R}	CLGOSC3.OSC3MD[1:0] bits = 0x1		–	–	3	µs
CR oscillator frequency/IC deviation	$\Delta f_{OSC3R}/\Delta IC$	CLGOSC3.OSC3MD[1:0] bits = 0x1	1.6 to 3.6 V	-30	–	30	%
Crystal/ceramic oscillator oscillation start time*1	t_{sta3C}	CLGOSC3.OSC3MD[1:0] bits = 0x2, CLGOSC3.OSC3INV[1:0] bits = 0x0		–	–	10	ms
Crystal/ceramic oscillator internal gate capacitance	C_{GI3C}	CLGOSC3.OSC3MD[1:0] bits = 0x2		–	8	–	pF
Crystal/ceramic oscillator internal drain capacitance	C_{DI3C}	CLGOSC3.OSC3MD[1:0] bits = 0x2		–	8	–	pF

Item	Symbol	Condition	VDD	Min.	Typ.	Max.	Unit
Crystal/ceramic oscillator circuit current - oscillation inverter drivability ratio	IOSC3C	CLGOSC3.OSC3MD[1:0] bits = 0x2, CLGOSC3.OSC3INV[1:0] bits = 0x0		–	50	–	%
		CLGOSC3.OSC3MD[1:0] bits = 0x2, CLGOSC3.OSC3INV[1:0] bits = 0x1 (reference)		–	100	–	%
		CLGOSC3.OSC3MD[1:0] bits = 0x2, CLGOSC3.OSC3INV[1:0] bits = 0x2		–	120	–	%
		CLGOSC3.OSC3MD[1:0] bits = 0x2, CLGOSC3.OSC3INV[1:0] bits = 0x3		–	190	–	%

*2 Ceramic resonator = CSBLA_J (manufactured by Murata Manufacturing Co., Ltd., 1 MHz), CG3 = CD3 = 100 pF

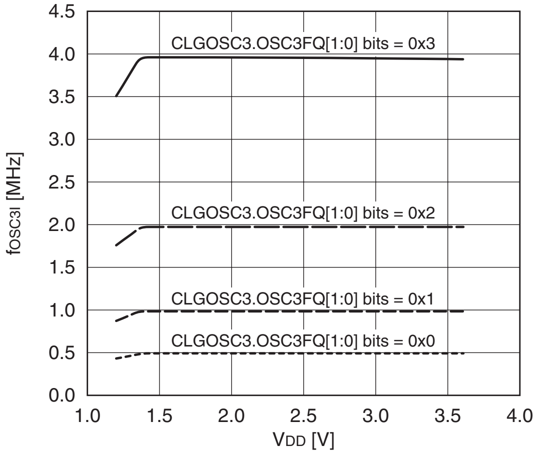
OSC3 internal oscillation frequency-temperature characteristic

VDD = 1.6 to 3.6 V, Typ. value



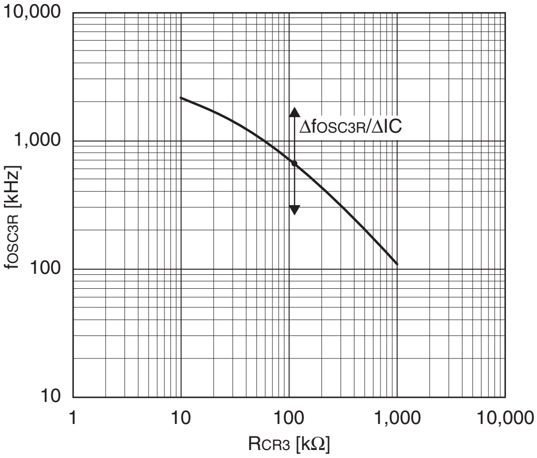
OSC3 internal oscillation frequency-power supply voltage characteristic

Ta = 25 °C, Typ. value



OSC3 CR oscillation frequency-resistance characteristic

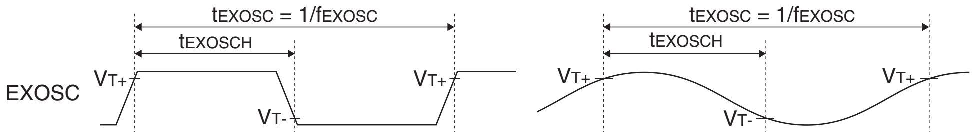
VDD = 1.6 to 3.6 V, Ta = 25 °C, Typ. value



EXOSC external clock input characteristics

Unless otherwise specified: VDD = 1.2 to 3.6 V, VSS = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
EXOSC external clock duty ratio	tEXOSCD	tEXOSCD = tEXOSCH/tEXOSC	46	–	54	%
High level Schmitt input threshold voltage	VT+		0.5 × VDD	–	0.8 × VDD	V
Low level Schmitt input threshold voltage	VT-		0.2 × VDD	–	0.5 × VDD	V
Schmitt input hysteresis voltage	ΔVT		120	–	–	mV



20.6 Flash Memory Characteristics

Unless otherwise specified: V_{DD} = 1.8 to 3.6 V, V_{SS} = 0 V *1, Ta = -40 to 85 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Programming count *2	C_{FEP}	Programmed data is guaranteed to be retained for 10 years.	50	–	–	times

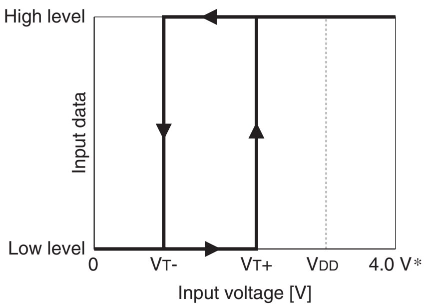
*1 The potential variation of the V_{SS} voltage should be suppressed to within ±0.3 V on the basis of the ground potential of the MCU mounting board while the Flash is being programmed, as it affects the Flash memory characteristics (programming count).

*2 Assumed that Erasing + Programming as count of 1. The count includes programming in the factory for shipment with ROM data programmed.

20.7 Input/Output Port (PPORT) Characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

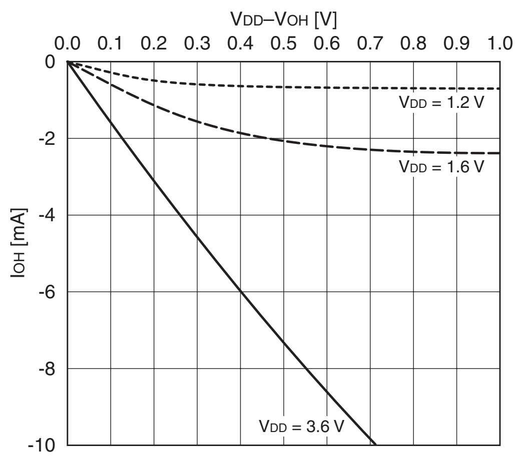
Item	Symbol	Condition	V_{DD}	Min.	Typ.	Max.	Unit
High level Schmitt input threshold voltage	V_{T+}	P00–06, P10–17, P20–27, P30–37, PD0–D1, PD3–D4		0.5 × V_{DD}	–	0.8 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}	P00–06, P10–17, P20–27, P30–37, PD0–D1, PD3–D4		0.2 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T	P00–06, P10–17, P20–27, P30–37, PD0–D1, PD3–D4		120	–	–	mV
High level output current	I_{OH}	P00–06, P10–17, P20–27, P30–37, PD0–D4, V_{OH} = 0.9 × V_{DD}	1.2 to 1.6 V	–	–	-0.3	mA
			1.6 to 3.6 V	–	–	-0.5	mA
Low level output current	I_{OL}	P00–06, P10–17, P20–27, P30–37, PD0–D4, V_{OL} = 0.1 × V_{DD}	1.2 to 1.6 V	0.3	–	–	mA
			1.6 to 3.6 V	0.5	–	–	mA
Leakage current	I_{LEAK}	P00–06, P10–17, P20–27, P30–37, PD0–D4		-150	–	150	nA
Input pull-up resistance	R_{INU}	P00–06, P10–17, P20–27, P30–37, PD0–D1, PD3–D4		75	150	300	kΩ
Input pull-down resistance	R_{IND}	P00–06, P10–17, P20–27, P30–37, PD0–D1, PD3–D4		75	150	300	kΩ
Pin capacitance	C_{IN}	P00–06, P10–17, P20–27, P30–37, PD0–D1, PD3–D4		–	–	15	pF



(* For over voltage tolerant fail-safe type port)

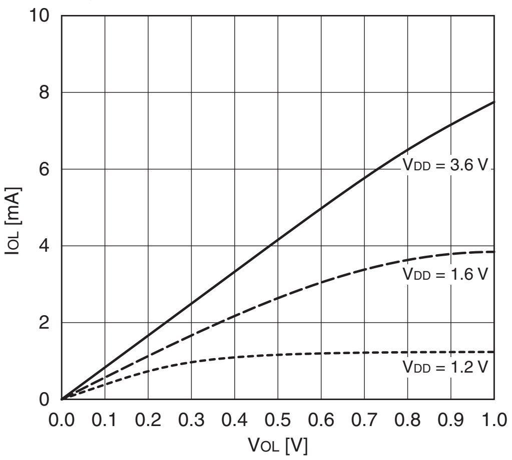
High-level output current characteristic

Ta = 85 °C, Max. value



Low-level output current characteristic

Ta = 85 °C, Min. value



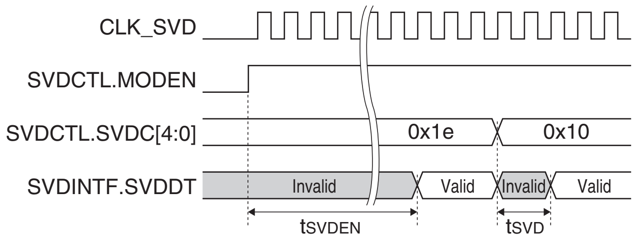
20.8 Supply Voltage Detector (SVD) Characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
EXSVD pin input voltage range	V_{EXSVD}		1.17	–	4.0	V
EXSVD input impedance	R_{EXSVD}	SVDCTL.SVDC[4:0] bits = 0x01	208	297	386	kΩ
		SVDCTL.SVDC[4:0] bits = 0x02	216	309	402	kΩ
		SVDCTL.SVDC[4:0] bits = 0x03	225	321	417	kΩ
		SVDCTL.SVDC[4:0] bits = 0x04	233	333	433	kΩ
		SVDCTL.SVDC[4:0] bits = 0x05	242	345	449	kΩ
		SVDCTL.SVDC[4:0] bits = 0x06	250	357	464	kΩ
		SVDCTL.SVDC[4:0] bits = 0x07	258	369	480	kΩ
		SVDCTL.SVDC[4:0] bits = 0x08	267	381	495	kΩ
		SVDCTL.SVDC[4:0] bits = 0x09	275	393	511	kΩ
		SVDCTL.SVDC[4:0] bits = 0x0a	284	405	527	kΩ
		SVDCTL.SVDC[4:0] bits = 0x0b	292	417	542	kΩ
		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 = 0x01	1.17	1.20	1.23	V
		SVDCTL.SVDC[4:0] bits = 0x02	1.22	1.25	1.28	V
		SVDCTL.SVDC[4:0] bits = 0x03	1.27	1.30	1.33	V
		SVDCTL.SVDC[4:0] bits = 0x04	1.32	1.35	1.38	V
		SVDCTL.SVDC[4:0] bits = 0x05	1.37	1.40	1.44	V
		SVDCTL.SVDC[4:0] bits = 0x06	1.41	1.45	1.49	V
		SVDCTL.SVDC[4:0] bits = 0x07	1.46	1.50	1.54	V
		SVDCTL.SVDC[4:0] bits = 0x08	1.51	1.55	1.59	V
		SVDCTL.SVDC[4:0] bits = 0x09	1.56	1.60	1.64	V
		SVDCTL.SVDC[4:0] bits = 0x0a	1.61	1.65	1.69	V
		SVDCTL.SVDC[4:0] bits = 0x0b	1.66	1.70	1.74	V
		SVDCTL.SVDC[4:0] bits = 0x0c	1.76	1.80	1.85	V
		SVDCTL.SVDC[4:0] bits = 0x0d	1.85	1.90	1.95	V
		SVDCTL.SVDC[4:0] bits = 0x0e	1.95	2.00	2.05	V
		SVDCTL.SVDC[4:0] bits = 0x0f	2.05	2.10	2.15	V
		SVDCTL.SVDC[4:0] bits = 0x10	2.15	2.20	2.26	V
		SVDCTL.SVDC[4:0] bits = 0x11	2.24	2.30	2.36	V
		SVDCTL.SVDC[4:0] bits = 0x12	2.34	2.40	2.46	V
		SVDCTL.SVDC[4:0] bits = 0x13	2.44	2.50	2.56	V
		SVDCTL.SVDC[4:0] bits = 0x14	2.54	2.60	2.67	V
		SVDCTL.SVDC[4:0] bits = 0x15	2.63	2.70	2.77	V
		SVDCTL.SVDC[4:0] bits = 0x16	2.73	2.80	2.87	V
		SVDCTL.SVDC[4:0] bits = 0x17	2.83	2.90	2.97	V
		SVDCTL.SVDC[4:0] bits = 0x18	2.93	3.00	3.08	V
		SVDCTL.SVDC[4:0] bits = 0x19	3.02	3.10	3.18	V
		SVDCTL.SVDC[4:0] bits = 0x1a	3.12	3.20	3.28	V
		SVDCTL.SVDC[4:0] bits = 0x1b	3.22	3.30	3.38	V
		SVDCTL.SVDC[4:0] bits = 0x1c	3.32	3.40	3.49	V
		SVDCTL.SVDC[4:0] bits = 0x1d	3.41	3.50	3.59	V
		SVDCTL.SVDC[4:0] bits = 0x1e	3.51	3.60	3.69	V
SVD circuit enable response time	t_{SVDEN}	*1	–	–	500	µs
SVD circuit response time	t_{SVD}		–	–	60	µs

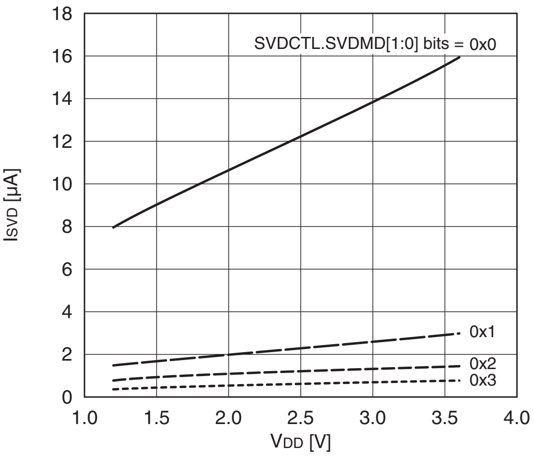
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
SVD circuit current	I_{SVD}	SVDCTL.SVDMD[1:0] bits = 0x0, SVDCTL.SVDC[4:0] bits = 0x01, CLK_SVD = 32 kHz, Ta = 25 °C	–	16	25	μA
		SVDCTL.SVDMD[1:0] bits = 0x1, SVDCTL.SVDC[4:0] bits = 0x01, CLK_SVD = 32 kHz, Ta = 25 °C	–	3	4.5	μA
		SVDCTL.SVDMD[1:0] bits = 0x2, SVDCTL.SVDC[4:0] bits = 0x01, CLK_SVD = 32 kHz, Ta = 25 °C	–	1.5	2.3	μA
		SVDCTL.SVDMD[1:0] bits = 0x3, SVDCTL.SVDC[4:0] bits = 0x01, CLK_SVD = 32 kHz, Ta = 25 °C	–	0.8	1.2	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 = 0x01, CLK_SVD = 32 kHz, Typ. value



20.9 UART (UART) Characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	V_{DD}	Min.	Typ.	Max.	Unit
Transfer baud rate	U_{BRT1}	Normal mode	1.6 to 3.6 V	150	–	230,400	bps
			1.2 to 1.6 V	150	–	57,600	bps
	U_{BRT2}	IrDA mode	1.6 to 3.6 V	150	–	115,200	bps
			1.2 to 1.6 V	150	–	57,600	bps

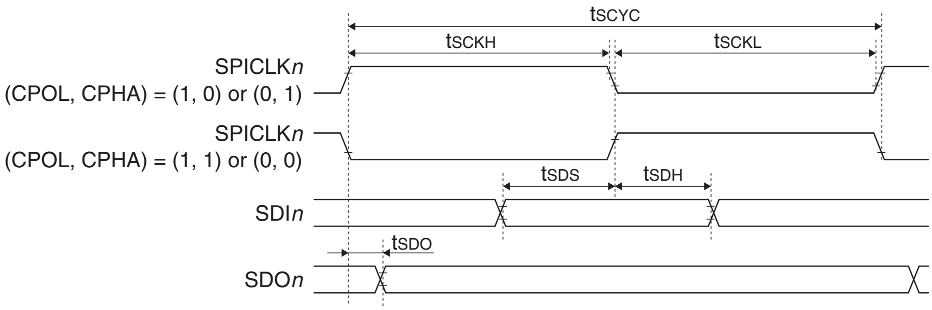
20.10 Synchronous Serial Interface (SPIA) Characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

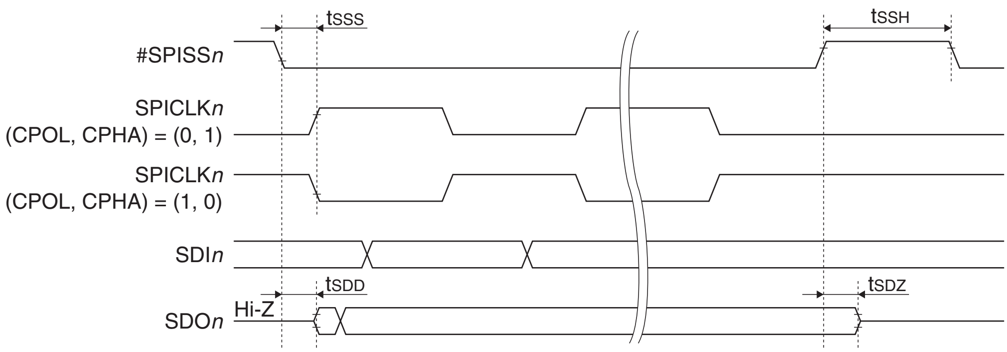
Item	Symbol	Condition	V_{DD}	Min.	Typ.	Max.	Unit
SPICLK*n* cycle time	t_{SCYC}		1.6 to 3.6 V	500	–	–	ns
			1.2 to 1.6 V	1,000	–	–	ns
SPICLK*n* High pulse width	t_{SCKH}		1.6 to 3.6 V	200	–	–	ns
			1.2 to 1.6 V	400	–	–	ns
SPICLK*n* Low pulse width	t_{SCKL}		1.6 to 3.6 V	200	–	–	ns
			1.2 to 1.6 V	400	–	–	ns
SDI*n* setup time	t_{SDS}		1.6 to 3.6 V	125	–	–	ns
			1.2 to 1.6 V	250	–	–	ns
SDI*n* hold time	t_{SDH}		1.6 to 3.6 V	70	–	–	ns
			1.2 to 1.6 V	140	–	–	ns
SDO*n* output delay time	t_{SDO}	C_L = 30 pF *1	1.6 to 3.6 V	–	–	100	ns
			1.2 to 1.6 V	–	–	200	ns
#SPISS*n* setup time	t_{SSS}		1.6 to 3.6 V	125	–	–	ns
			1.2 to 1.6 V	250	–	–	ns
#SPISS*n* High pulse width	t_{SSH}		1.6 to 3.6 V	80	–	–	ns
			1.2 to 1.6 V	160	–	–	ns
SDO*n* output start time	t_{SDD}	C_L = 30 pF *1	1.6 to 3.6 V	–	–	100	ns
			1.2 to 1.6 V	–	–	200	ns
SDO*n* output stop time	t_{SDZ}	C_L = 30 pF *1	1.6 to 3.6 V	–	–	80	ns
			1.2 to 1.6 V	–	–	160	ns

*1 C_L = Pin load

Master and slave modes



Slave mode

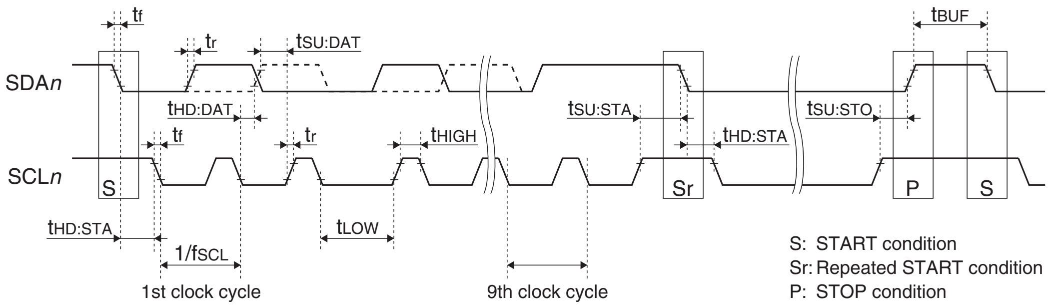


20.11 I²C (I2C) Characteristics

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	Standard mode V_{DD} = 1.2 to 3.6 V			Fast mode V_{DD} = 1.6 to 3.6 V			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.



20.12 LCD Driver (LCD8B) Characteristics

The LCD driver characteristics varies depending on the panel load (panel size, drive duty, number of display pixels and display contents), so evaluate them by connecting to the actually used LCD panel.

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = 25 °C, LCD8TIM2.BSTC[1:0] bits = 0x1 (Voltage booster clock = 2 kHz), No panel load

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
LCD drive voltage (1/3 bias, V_{C2} reference voltage) V_{DD} = 2.7 to 3.6 V LCD8PWR.BISEL bit = 0 LCD8PWR.VCSEL bit = 1	V_{C1}	Connect 1 MΩ load resistor between V_{SS} and V_{C1}		0.318 × V_{C3} (Typ.)	–	0.352 × V_{C3} (Typ.)	V
	V_{C2}	Connect 1 MΩ load resistor between V_{SS} and V_{C2}		0.660 × V_{C3} (Typ.)	–	0.694 × V_{C3} (Typ.)	V
	V_{C3}	Connect 1 MΩ load resistor between V_{SS} and V_{C3}	LCD8PWR.LC[4:0] bits = 0x00	2.43	2.51	2.59	V
			LCD8PWR.LC[4:0] bits = 0x01	2.48	2.56	2.64	V
			LCD8PWR.LC[4:0] bits = 0x02	2.52	2.60	2.68	V
			LCD8PWR.LC[4:0] bits = 0x03	2.56	2.64	2.72	V
			LCD8PWR.LC[4:0] bits = 0x04	2.61	2.69	2.77	V
			LCD8PWR.LC[4:0] bits = 0x05	2.65	2.73	2.81	V
			LCD8PWR.LC[4:0] bits = 0x06	2.70	2.78	2.86	V
			LCD8PWR.LC[4:0] bits = 0x07	2.74	2.82	2.90	V
			LCD8PWR.LC[4:0] bits = 0x08	2.78	2.87	2.96	V
			LCD8PWR.LC[4:0] bits = 0x09	2.82	2.91	3.00	V
			LCD8PWR.LC[4:0] bits = 0x0a	2.87	2.96	3.05	V
			LCD8PWR.LC[4:0] bits = 0x0b	2.91	3.00	3.09	V
			LCD8PWR.LC[4:0] bits = 0x0c	2.95	3.04	3.13	V
			LCD8PWR.LC[4:0] bits = 0x0d	3.00	3.09	3.18	V
			LCD8PWR.LC[4:0] bits = 0x0e	3.04	3.13	3.22	V
			LCD8PWR.LC[4:0] bits = 0x0f	3.08	3.18	3.28	V
			LCD8PWR.LC[4:0] bits = 0x10	3.12	3.22	3.32	V
			LCD8PWR.LC[4:0] bits = 0x11	3.17	3.27	3.37	V

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
LCD drive voltage (1/3 bias, V_{C2} reference voltage) V_{DD} = 2.7 to 3.6 V LCD8PWR.BISEL bit = 0 LCD8PWR.VCSEL bit = 1	V_{C3}	Connect 1 MΩ load resistor between V_{SS} and V_{C3}	LCD8PWR.LC[4:0] bits = 0x12	3.21	3.31	3.41	V
			LCD8PWR.LC[4:0] bits = 0x13	3.25	3.35	3.45	V
			LCD8PWR.LC[4:0] bits = 0x14	3.30	3.40	3.50	V
			LCD8PWR.LC[4:0] bits = 0x15	3.34	3.44	3.54	V
			LCD8PWR.LC[4:0] bits = 0x16	3.39	3.49	3.59	V
			LCD8PWR.LC[4:0] bits = 0x17	3.42	3.53	3.64	V
			LCD8PWR.LC[4:0] bits = 0x18	3.47	3.58	3.69	V
			LCD8PWR.LC[4:0] bits = 0x19	3.51	3.62	3.73	V
			LCD8PWR.LC[4:0] bits = 0x1a	3.55	3.66	3.77	V
			LCD8PWR.LC[4:0] bits = 0x1b	3.60	3.71	3.82	V
			LCD8PWR.LC[4:0] bits = 0x1c	3.64	3.75	3.86	V
			LCD8PWR.LC[4:0] bits = 0x1d	3.69	3.80	3.91	V
			LCD8PWR.LC[4:0] bits = 0x1e	3.72	3.84	3.96	V
			LCD8PWR.LC[4:0] bits = 0x1f	3.77	3.89	4.01	V
LCD drive voltage (1/3 bias, V_{C1} reference voltage) V_{DD} = 1.2 to 3.6 V *3 LCD8PWR.BISEL bit = 0 LCD8PWR.VCSEL bit = 0	V_{C1}	Connect 1 MΩ load resistor between V_{SS} and V_{C1}		0.321 × V_{C3} (Typ.)	–	0.355 × V_{C3} (Typ.)	V
	V_{C2}	Connect 1 MΩ load resistor between V_{SS} and V_{C2}		0.656 × V_{C3} (Typ.)	–	0.703 × V_{C3} (Typ.)	V
	V_{C3}	Connect 1 MΩ load resistor between V_{SS} and V_{C3}	LCD8PWR.LC[4:0] bits = 0x00	2.40	2.47	2.54	V
			LCD8PWR.LC[4:0] bits = 0x01	2.43	2.51	2.59	V
			LCD8PWR.LC[4:0] bits = 0x02	2.47	2.55	2.63	V
			LCD8PWR.LC[4:0] bits = 0x03	2.52	2.60	2.68	V
			LCD8PWR.LC[4:0] bits = 0x04	2.56	2.64	2.72	V
			LCD8PWR.LC[4:0] bits = 0x05	2.61	2.69	2.77	V
			LCD8PWR.LC[4:0] bits = 0x06	2.65	2.73	2.81	V
			LCD8PWR.LC[4:0] bits = 0x07	2.70	2.78	2.86	V
			LCD8PWR.LC[4:0] bits = 0x08	2.74	2.82	2.90	V
			LCD8PWR.LC[4:0] bits = 0x09	2.78	2.87	2.96	V
			LCD8PWR.LC[4:0] bits = 0x0a	2.82	2.91	3.00	V
			LCD8PWR.LC[4:0] bits = 0x0b	2.87	2.96	3.05	V
			LCD8PWR.LC[4:0] bits = 0x0c	2.91	3.00	3.09	V
			LCD8PWR.LC[4:0] bits = 0x0d	2.95	3.04	3.13	V
			LCD8PWR.LC[4:0] bits = 0x0e	3.00	3.09	3.18	V
			LCD8PWR.LC[4:0] bits = 0x0f	3.04	3.13	3.22	V
			LCD8PWR.LC[4:0] bits = 0x10	3.08	3.18	3.28	V
			LCD8PWR.LC[4:0] bits = 0x11	3.12	3.22	3.32	V
			LCD8PWR.LC[4:0] bits = 0x12	3.17	3.27	3.37	V
			LCD8PWR.LC[4:0] bits = 0x13	3.21	3.31	3.41	V
			LCD8PWR.LC[4:0] bits = 0x14	3.26	3.36	3.46	V
			LCD8PWR.LC[4:0] bits = 0x15	3.30	3.40	3.50	V
			LCD8PWR.LC[4:0] bits = 0x16	3.35	3.45	3.55	V
			LCD8PWR.LC[4:0] bits = 0x17	3.39	3.49	3.59	V
			LCD8PWR.LC[4:0] bits = 0x18	3.42	3.53	3.64	V
			LCD8PWR.LC[4:0] bits = 0x19	3.47	3.58	3.69	V
			LCD8PWR.LC[4:0] bits = 0x1a	3.51	3.62	3.73	V
			LCD8PWR.LC[4:0] bits = 0x1b	3.56	3.67	3.78	V
			LCD8PWR.LC[4:0] bits = 0x1c	3.60	3.71	3.82	V
			LCD8PWR.LC[4:0] bits = 0x1d	3.65	3.76	3.87	V
			LCD8PWR.LC[4:0] bits = 0x1e	3.69	3.80	3.91	V
			LCD8PWR.LC[4:0] bits = 0x1f	3.73	3.85	3.97	V
LCD drive voltage (1/4 bias, V_{C2} reference voltage) V_{DD} = 2.7 to 3.6 V LCD8PWR.BISEL bit = 1 LCD8PWR.VCSEL bit = 1	V_{C1}	Connect 1 MΩ load resistor between V_{SS} and V_{C1}		0.240 × V_{C4} (Typ.)	–	0.265 × V_{C4} (Typ.)	V
	V_{C2}	Connect 1 MΩ load resistor between V_{SS} and V_{C2}		0.497 × V_{C4} (Typ.)	–	0.523 × V_{C4} (Typ.)	V
	V_{C3}	Connect 1 MΩ load resistor between V_{SS} and V_{C3}		0.727 × V_{C4} (Typ.)	–	0.781 × V_{C4} (Typ.)	V
	V_{C4}	Connect 1 MΩ load resistor between V_{SS} and V_{C4}	LCD8PWR.LC[4:0] bits = 0x00	3.20	3.33	3.46	V
			LCD8PWR.LC[4:0] bits = 0x01	3.25	3.39	3.53	V
			LCD8PWR.LC[4:0] bits = 0x02	3.31	3.45	3.59	V
			LCD8PWR.LC[4:0] bits = 0x03	3.37	3.51	3.65	V
			LCD8PWR.LC[4:0] bits = 0x04	3.43	3.57	3.71	V
			LCD8PWR.LC[4:0] bits = 0x05	3.48	3.63	3.78	V
			LCD8PWR.LC[4:0] bits = 0x06	3.53	3.68	3.83	V
			LCD8PWR.LC[4:0] bits = 0x07	3.59	3.74	3.89	V
			LCD8PWR.LC[4:0] bits = 0x08	3.65	3.80	3.95	V

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
LCD drive voltage (1/4 bias, V_{C2} reference voltage) V_{DD} = 2.7 to 3.6 V LCD8PWR.BISEL bit = 1 LCD8PWR.VCSEL bit = 1	V_{C4}	Connect 1 MΩ load resistor between V_{SS} and V_{C4}	LCD8PWR.LC[4:0] bits = 0x09	3.71	3.86	4.01	V
			LCD8PWR.LC[4:0] bits = 0x0a	3.76	3.92	4.08	V
			LCD8PWR.LC[4:0] bits = 0x0b	3.82	3.98	4.14	V
			LCD8PWR.LC[4:0] bits = 0x0c	3.88	4.04	4.20	V
			LCD8PWR.LC[4:0] bits = 0x0d	3.94	4.10	4.26	V
			LCD8PWR.LC[4:0] bits = 0x0e	3.99	4.16	4.33	V
			LCD8PWR.LC[4:0] bits = 0x0f	4.04	4.21	4.38	V
			LCD8PWR.LC[4:0] bits = 0x10	4.10	4.27	4.44	V
			LCD8PWR.LC[4:0] bits = 0x11	4.16	4.33	4.50	V
			LCD8PWR.LC[4:0] bits = 0x12	4.21	4.39	4.57	V
			LCD8PWR.LC[4:0] bits = 0x13	4.27	4.45	4.63	V
			LCD8PWR.LC[4:0] bits = 0x14	4.33	4.51	4.69	V
			LCD8PWR.LC[4:0] bits = 0x15	4.39	4.57	4.75	V
			LCD8PWR.LC[4:0] bits = 0x16	4.44	4.63	4.82	V
			LCD8PWR.LC[4:0] bits = 0x17	4.49	4.68	4.87	V
			LCD8PWR.LC[4:0] bits = 0x18	4.55	4.74	4.93	V
			LCD8PWR.LC[4:0] bits = 0x19	4.61	4.80	4.99	V
			LCD8PWR.LC[4:0] bits = 0x1a	4.67	4.86	5.05	V
			LCD8PWR.LC[4:0] bits = 0x1b	4.72	4.92	5.12	V
			LCD8PWR.LC[4:0] bits = 0x1c	4.78	4.98	5.18	V
			LCD8PWR.LC[4:0] bits = 0x1d	4.84	5.04	5.24	V
			LCD8PWR.LC[4:0] bits = 0x1e	4.90	5.10	5.30	V
			LCD8PWR.LC[4:0] bits = 0x1f	4.94	5.15	5.36	V
LCD drive voltage (1/4 bias, V_{C1} reference voltage) V_{DD} = 1.2 to 3.6 V *3 LCD8PWR.BISEL bit = 1 LCD8PWR.VCSEL bit = 0	V_{C1}	Connect 1 MΩ load resistor between V_{SS} and V_{C1}		0.245 × V_{C4} (Typ.)	–	0.271 × V_{C4} (Typ.)	V
	V_{C2}	Connect 1 MΩ load resistor between V_{SS} and V_{C2}		0.495 × V_{C4} (Typ.)	–	0.531 × V_{C4} (Typ.)	V
	V_{C3}	Connect 1 MΩ load resistor between V_{SS} and V_{C3}		0.735 × V_{C4} (Typ.)	–	0.789 × V_{C4} (Typ.)	V
	V_{C4}	Connect 1 MΩ load resistor between V_{SS} and V_{C4}	LCD8PWR.LC[4:0] bits = 0x00	3.09	3.22	3.35	V
			LCD8PWR.LC[4:0] bits = 0x01	3.15	3.28	3.41	V
			LCD8PWR.LC[4:0] bits = 0x02	3.21	3.34	3.47	V
			LCD8PWR.LC[4:0] bits = 0x03	3.26	3.40	3.54	V
			LCD8PWR.LC[4:0] bits = 0x04	3.31	3.45	3.59	V
			LCD8PWR.LC[4:0] bits = 0x05	3.37	3.51	3.65	V
			LCD8PWR.LC[4:0] bits = 0x06	3.43	3.57	3.71	V
			LCD8PWR.LC[4:0] bits = 0x07	3.48	3.63	3.78	V
			LCD8PWR.LC[4:0] bits = 0x08	3.54	3.69	3.84	V
			LCD8PWR.LC[4:0] bits = 0x09	3.59	3.74	3.89	V
			LCD8PWR.LC[4:0] bits = 0x0a	3.65	3.80	3.95	V
			LCD8PWR.LC[4:0] bits = 0x0b	3.71	3.86	4.01	V
			LCD8PWR.LC[4:0] bits = 0x0c	3.76	3.92	4.08	V
			LCD8PWR.LC[4:0] bits = 0x0d	3.82	3.98	4.14	V
			LCD8PWR.LC[4:0] bits = 0x0e	3.88	4.04	4.20	V
			LCD8PWR.LC[4:0] bits = 0x0f	3.93	4.09	4.25	V
			LCD8PWR.LC[4:0] bits = 0x10	3.98	4.15	4.32	V
			LCD8PWR.LC[4:0] bits = 0x11	4.04	4.21	4.38	V
			LCD8PWR.LC[4:0] bits = 0x12	4.10	4.27	4.44	V
			LCD8PWR.LC[4:0] bits = 0x13	4.16	4.33	4.50	V
			LCD8PWR.LC[4:0] bits = 0x14	4.20	4.38	4.56	V
			LCD8PWR.LC[4:0] bits = 0x15	4.26	4.44	4.62	V
			LCD8PWR.LC[4:0] bits = 0x16	4.32	4.50	4.68	V
			LCD8PWR.LC[4:0] bits = 0x17	4.38	4.56	4.74	V
			LCD8PWR.LC[4:0] bits = 0x18	4.44	4.62	4.80	V
			LCD8PWR.LC[4:0] bits = 0x19	4.49	4.68	4.87	V
			LCD8PWR.LC[4:0] bits = 0x1a	4.54	4.73	4.92	V
			LCD8PWR.LC[4:0] bits = 0x1b	4.60	4.79	4.98	V
			LCD8PWR.LC[4:0] bits = 0x1c	4.66	4.85	5.04	V
			LCD8PWR.LC[4:0] bits = 0x1d	4.71	4.91	5.11	V
			LCD8PWR.LC[4:0] bits = 0x1e	4.77	4.97	5.17	V
			LCD8PWR.LC[4:0] bits = 0x1f	4.82	5.02	5.22	V
Segment/Common output current	I_{SEGH}	SEG*xx*, COM*y* V_{SEGH} = V_{C4}/V_{C3}/V_{C2}/V_{C1} - 0.1 V, Ta = -40 to 85 °C		–	–	-10	µA
	I_{SEGL}	SEG*xx*, COM*y* V_{SEGL} = V_{C4}/V_{SS}/V_{C2}/V_{C1} + 0.1 V, Ta = -40 to 85 °C		10	–	–	µA

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
LCD circuit current (1/3 bias, V_{C2} reference voltage)	I_{LCD2}	LCD8DSP.DSPC[1:0] bits = 0x1 (checker pattern), LCD8PWR.BISEL bit = 0, LCD8PWR.VCSEL bit = 1 *1 *2	–	3.5	7.5	µA
		LCD8DSP.DSPC[1:0] bits = 0x2 (all on), LCD8PWR.BISEL bit = 0, LCD8PWR.VCSEL bit = 1 *1 *2	–	0.9	3	µA
LCD circuit current (1/3 bias, V_{C1} reference voltage)	I_{LCD1}	LCD8DSP.DSPC[1:0] bits = 0x1 (checker pattern), LCD8PWR.BISEL bit = 0, LCD8PWR.VCSEL bit = 0 *1 *2	–	6.5	15	µA
		LCD8DSP.DSPC[1:0] bits = 0x2 (all on), LCD8PWR.BISEL bit = 0, LCD8PWR.VCSEL bit = 0 *1 *2	–	1.4	4.5	µA
LCD circuit current (1/4 bias, V_{C2} reference voltage)	I_{LCD2}	LCD8DSP.DSPC[1:0] bits = 0x1 (checker pattern), LCD8PWR.BISEL bit = 1, LCD8PWR.VCSEL bit = 1 *1 *2	–	4.5	10	µA
		LCD8DSP.DSPC[1:0] bits = 0x2 (all on), LCD8PWR.BISEL bit = 1, LCD8PWR.VCSEL bit = 1 *1 *2	–	1.7	5	µA
LCD circuit current (1/4 bias, V_{C1} reference voltage)	I_{LCD1}	LCD8DSP.DSPC[1:0] bits = 0x1 (checker pattern), LCD8PWR.BISEL bit = 1, LCD8PWR.VCSEL bit = 0 *1 *2	–	8.5	20	µA
		LCD8DSP.DSPC[1:0] bits = 0x2 (all on), LCD8PWR.BISEL bit = 1, LCD8PWR.VCSEL bit = 0 *1 *2	–	3	9	µA
LCD circuit current in heavy load protection mode (1/3 bias, V_{C2} reference voltage)	I_{LCD2H}	LCD8DSP.DSPC[1:0] bits = 0x2 (all on), LCD8PWR.BISEL bit = 0, LCD8PWR.VCSEL bit = 1, LCD8PWR.HVLD bit = 1 *1 *2	–	13	30	µA
LCD circuit current in heavy load protection mode (1/4 bias, V_{C2} reference voltage)	I_{LCD2H}	LCD8DSP.DSPC[1:0] bits = 0x2 (all on), LCD8PWR.BISEL bit = 1, LCD8PWR.VCSEL bit = 1, LCD8PWR.HVLD bit = 1 *1 *2	–	14	31	µA

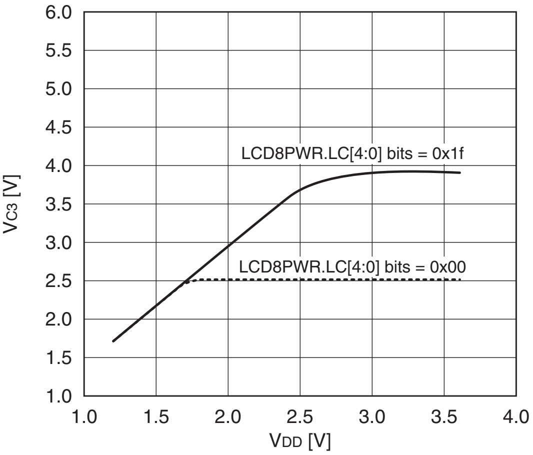
*1 Other LCD driver settings: LCD8PWR.LC[4:0] bits = 0x1f, CLK_LCD8B = 32 kHz, LCD8TIM1.FRMCNT[3:0] bits = 0x3 (frame frequency = 64 Hz)

*2 The value is added to the current consumption in HALT/RUN mode. Current consumption increases according to the display contents and panel load.

*3 The LCD drive voltage is lower than the LCD8PWR.LC[4:0] bit settings when V_{DD} = 1.2 to 1.6 V. See the LCD drive voltage-supply voltage characteristic graph shown below.

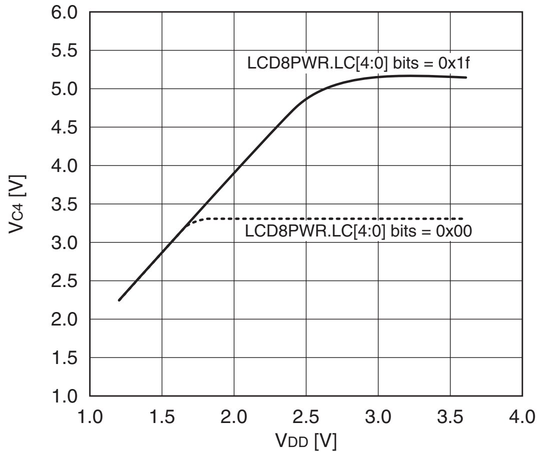
LCD drive voltage-supply voltage characteristic (1/3 bias, V_{C2} reference voltage)

Ta = 25 °C, Typ. value, when a 1 MΩ load resistor is connected between V_{SS} and V_{C3} (no panel load)



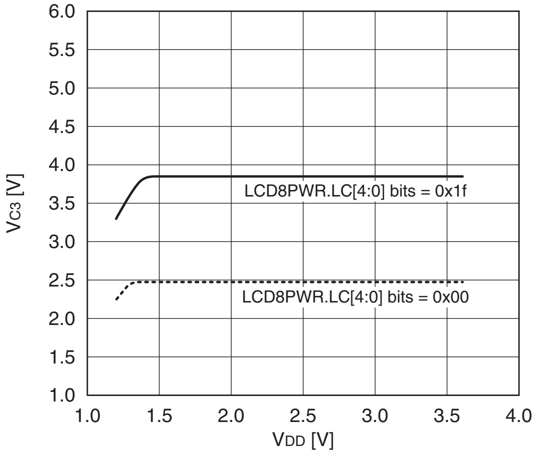
LCD drive voltage-supply voltage characteristic (1/4 bias, V_{C2} reference voltage)

Ta = 25 °C, Typ. value, when a 1 MΩ load resistor is connected between V_{SS} and V_{C4} (no panel load)



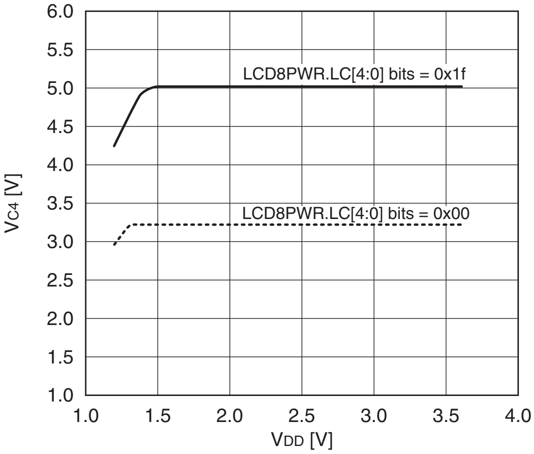
LCD drive voltage-supply voltage characteristic (1/3 bias, V_{C1} reference voltage)

Ta = 25 °C, Typ. value, when a 1 MΩ load resistor is connected between V_{SS} and V_{C3} (no panel load)



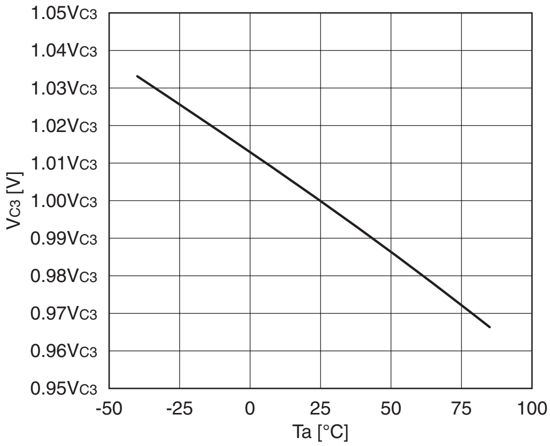
LCD drive voltage-supply voltage characteristic (1/4 bias, V_{C1} reference voltage)

Ta = 25 °C, Typ. value, when a 1 MΩ load resistor is connected between V_{SS} and V_{C4} (no panel load)



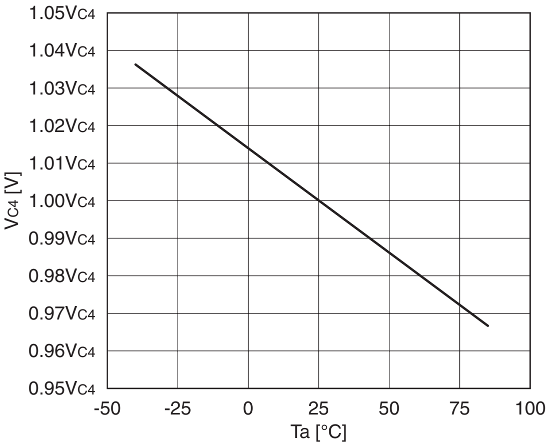
LCD drive voltage-temperature characteristic (1/3 bias, V_{C1}/V_{C2} reference voltage)

Typ. value, when a load is connected to the V_{C3} pin only



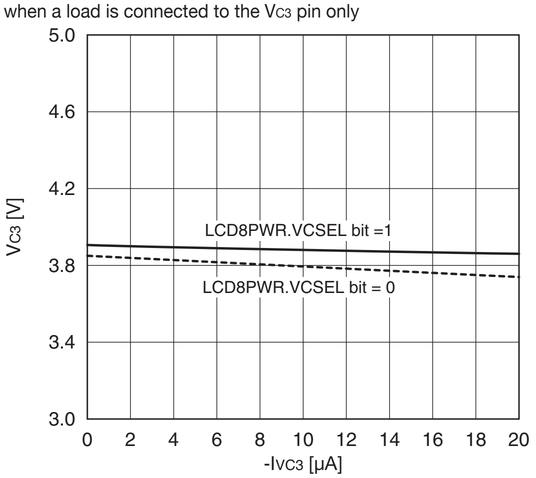
LCD drive voltage-temperature characteristic (1/4 bias, V_{C1}/V_{C2} reference voltage)

Typ. value, when a load is connected to the V_{C4} pin only



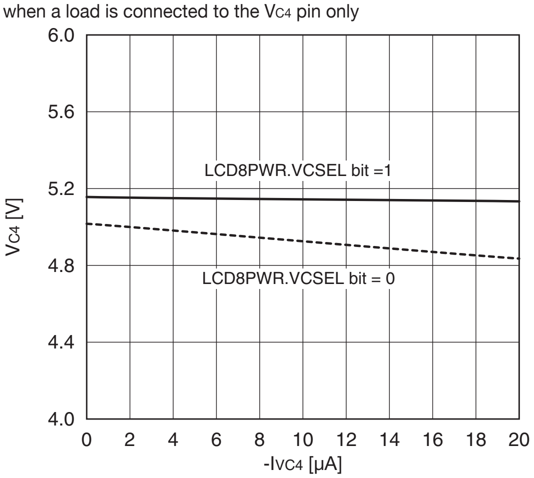
LCD drive voltage-load characteristic (1/3 bias)

Ta = 25 °C, Typ. value, LCD8PWR.LC[4:0] bits = 0x1f, when a load is connected to the V_{C3} pin only



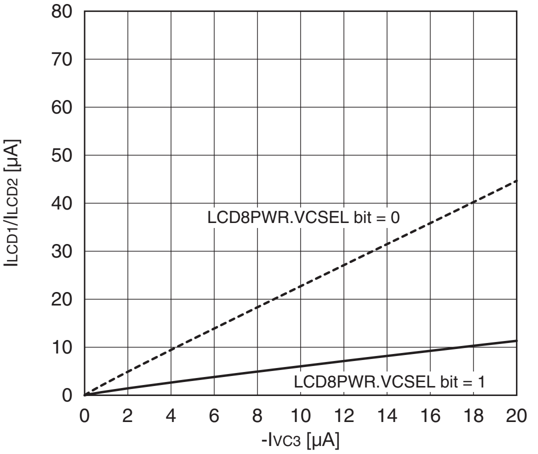
LCD drive voltage-load characteristic (1/4 bias)

Ta = 25 °C, Typ. value, LCD8PWR.LC[4:0] bits = 0x1f, when a load is connected to the V_{C4} pin only



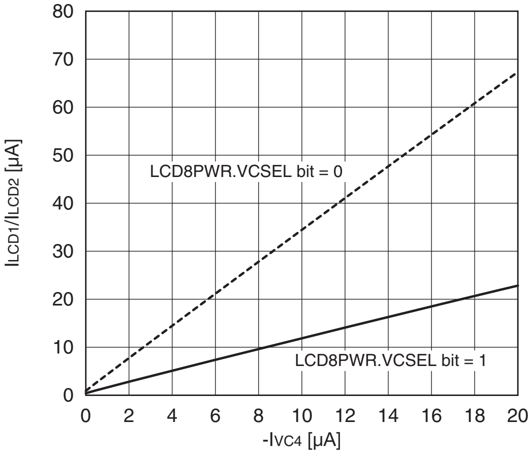
LCD circuit current-load characteristic (1/3 bias)

Ta = 25 °C, Typ. value, LCD8PWR.LC[4:0] bits = 0x1f,
when a load is connected to the V_{C3} pin only



LCD circuit current-load characteristic (1/4 bias)

Ta = 25 °C, Typ. value, LCD8PWR.LC[4:0] bits = 0x1f,
when a load is connected to the V_{C4} pin only



20.13 R/F Converter (RFC) Characteristics

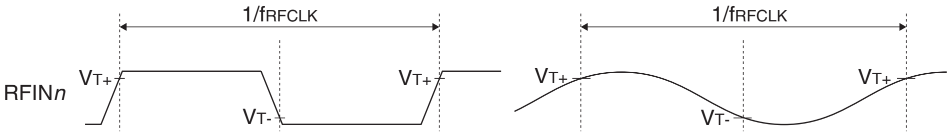
R/F converter characteristics change depending on conditions (board pattern, components used, etc.). Use these characteristic values as a reference and perform evaluation using the actual printed circuit board.

Unless otherwise specified: V_{DD} = 1.2 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	V_{DD}	Min.	Typ.	Max.	Unit
Reference/sensor oscillation frequency	f_{RFCLK}			1	–	1,000	kHz
Reference/sensor oscillation frequency IC deviation	$\Delta f_{RFCLK}/\Delta IC$	Ta = 25 °C *1	1.6 to 3.6 V	-30	–	30	%
			1.2 to 1.6 V	-50	–	50	%
Reference resistor/resistive sensor resistance	R_{REF}, R_{SEN}			1	–	–	kΩ
Reference capacitance	C_{REF}		1.6 to 3.6 V	100	–	–	pF
Time base counter clock frequency	f_{TCCLK}		1.6 to 3.6 V	–	–	4.2	MHz
			1.2 to 1.6 V	–	–	1.1	MHz
High level Schmitt input threshold voltage	V_{T+}			0.5 × V_{DD}	–	0.8 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}			0.2 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T			120	–	–	mV
R/F converter operating current	I_{RFC}	C_{REF} = 1,000 pF, R_{REF}/R_{SEN} = 100 kΩ, Ta = 25 °C, V_{DD} = 3.6 V		–	200	350	μA

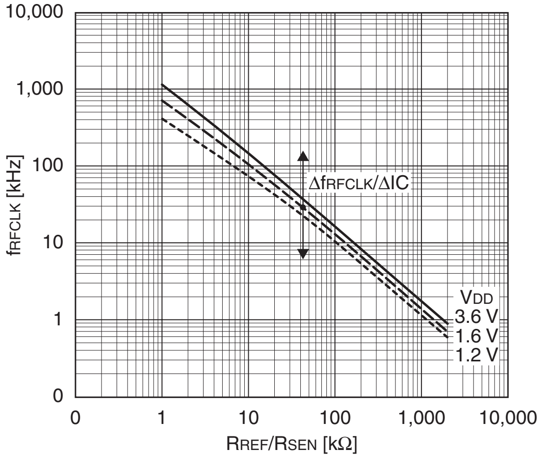
*1 In this characteristic, unevenness between production lots, and variations in measurement board, resistances and capacitances are taken into account.

Waveforms for external clock input mode



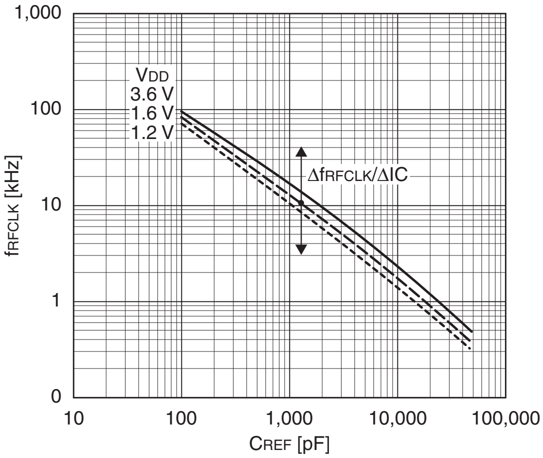
RFC reference/sensor oscillation frequency-resistance characteristic

CREF = 1,000 pF, Ta = 25 °C, Typ. value



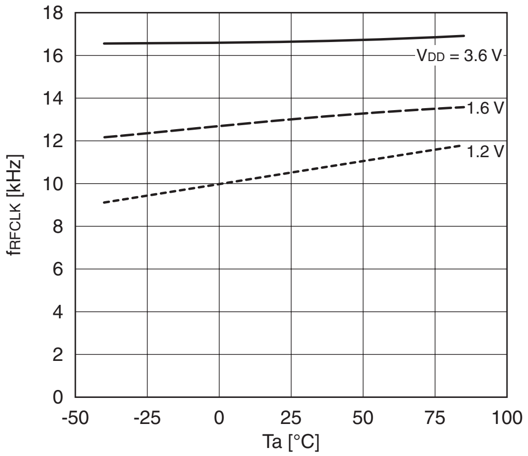
RFC reference/sensor oscillation frequency-capacitance characteristic

RREF/RSEN = 100 kΩ, Ta = 25 °C, Typ. value



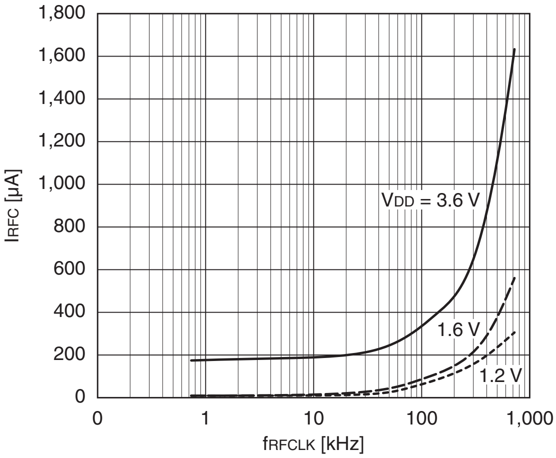
RFC reference/sensor oscillation current consumption-temperature characteristic

RREF/RSEN = 100 kΩ, CREF = 1,000 pF, Typ. value

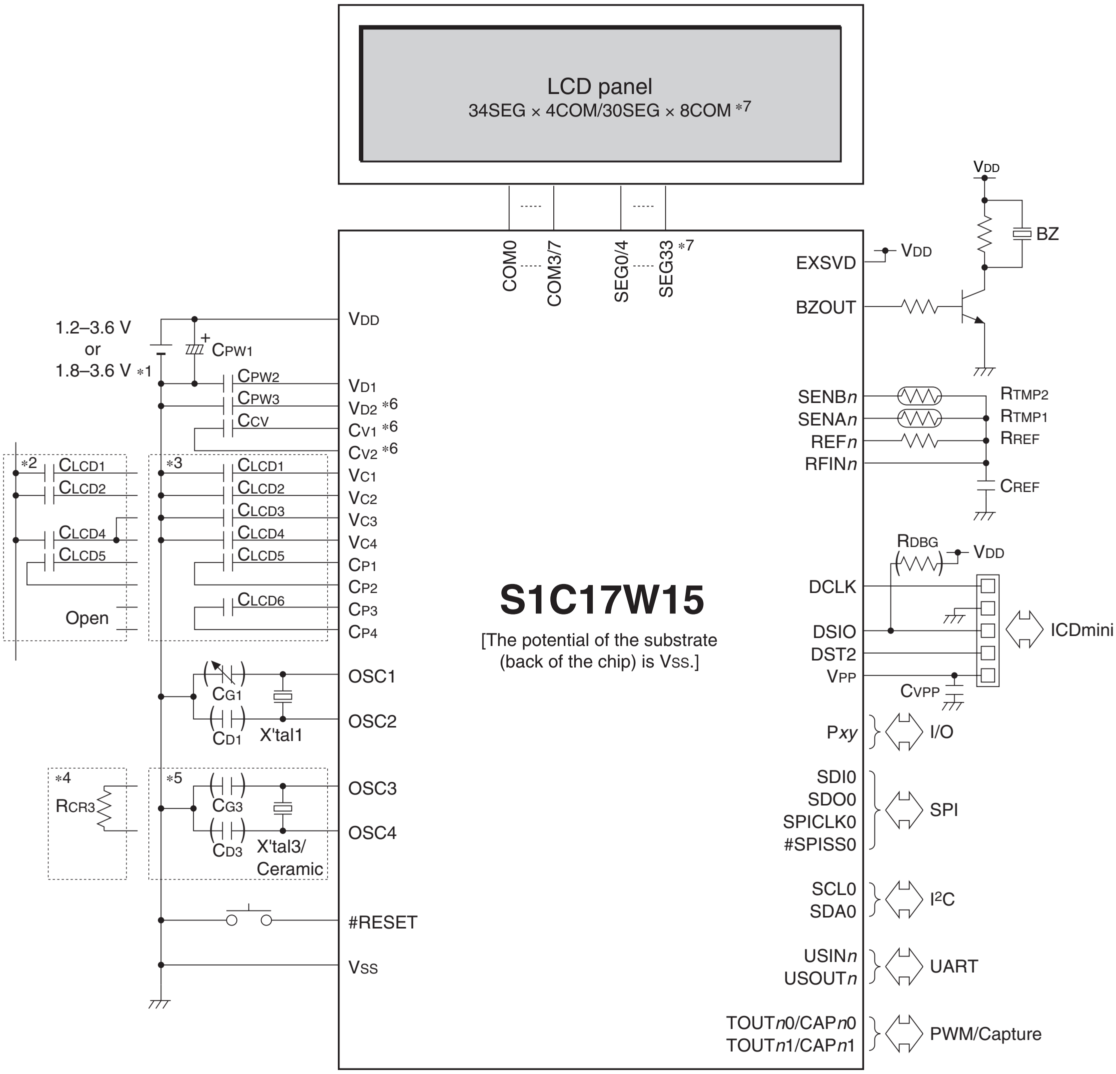


RFC reference/sensor oscillation current consumption-frequency characteristic

CREF = 1,000 pF, Ta = 25 °C, Typ. value



21 Basic External Connection Diagram



*1: For Flash programming
*2: When 1/3 bias is selected
*3: When 1/4 bias is selected
*4: When OSC3 CR oscillator is selected
*5: When OSC3 crystal/ceramic oscillator is selected
*6: These pins do not exist in the 64-pin package.
*7: The SEG24–SEG33 pins do not exist in the 64-pin package. (24SEG × 4COM/20SEG × 8COM)
The SEG28–SEG29 pins do not exist in the 80-pin package. (32SEG × 4COM/28SEG × 8COM)
( ): Do not mount components if unnecessary.

Sample external components

Symbol	Name	Recommended components
X'tal1	32 kHz crystal resonator	C-002RX (R_1 = 50 kΩ (Max.), C_L = 7 pF) manufactured by Seiko Epson Corporation
C_{G1}	OSC1 gate capacitor	Trimmer capacitor or ceramic capacitor
C_{D1}	OSC1 drain capacitor	Ceramic capacitor
X'tal3	Crystal resonator	CA-301 (4 MHz) manufactured by Seiko Epson Corporation
Ceramic	Ceramic resonator	CSBLA_J (1 MHz) manufactured by Murata Manufacturing Co., Ltd.
C_{G3}	OSC3 gate capacitor	Ceramic capacitor
C_{D3}	OSC3 drain capacitor	Ceramic capacitor
R_{CR3}	OSC3 oscillating resistor	Thick film chip resistor
C_{PW1}	Bypass capacitor between V_{SS} and V_{DD}	Ceramic capacitor or electrolytic capacitor
C_{PW2-3}	Capacitors between V_{SS} and V_{D1-2}	Ceramic capacitor
C_{CV}	Capacitor between C_{V1} and C_{V2}	Ceramic capacitor
C_{LCD1-4}	Capacitors between V_{SS} and V_{C1-4}	Ceramic capacitor
C_{LCD5-6}	Capacitors between C_{P1} and C_{P2}, C_{P3} and C_{P4}	Ceramic capacitor
BZ	Piezoelectric buzzer	PS1240P02 manufactured by TDK Corporation
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 reference capacitor	Ceramic capacitor
C_{VPP}	Capacitor between V_{SS} and V_{PP}	Ceramic capacitor

* For recommended component values, refer to “Recommended Operating Conditions” in the “Electrical Characteristics” chapter.

22 Package

SQFN9-64PIN (P-VQFN064-0909-0.50)

(Unit: mm)

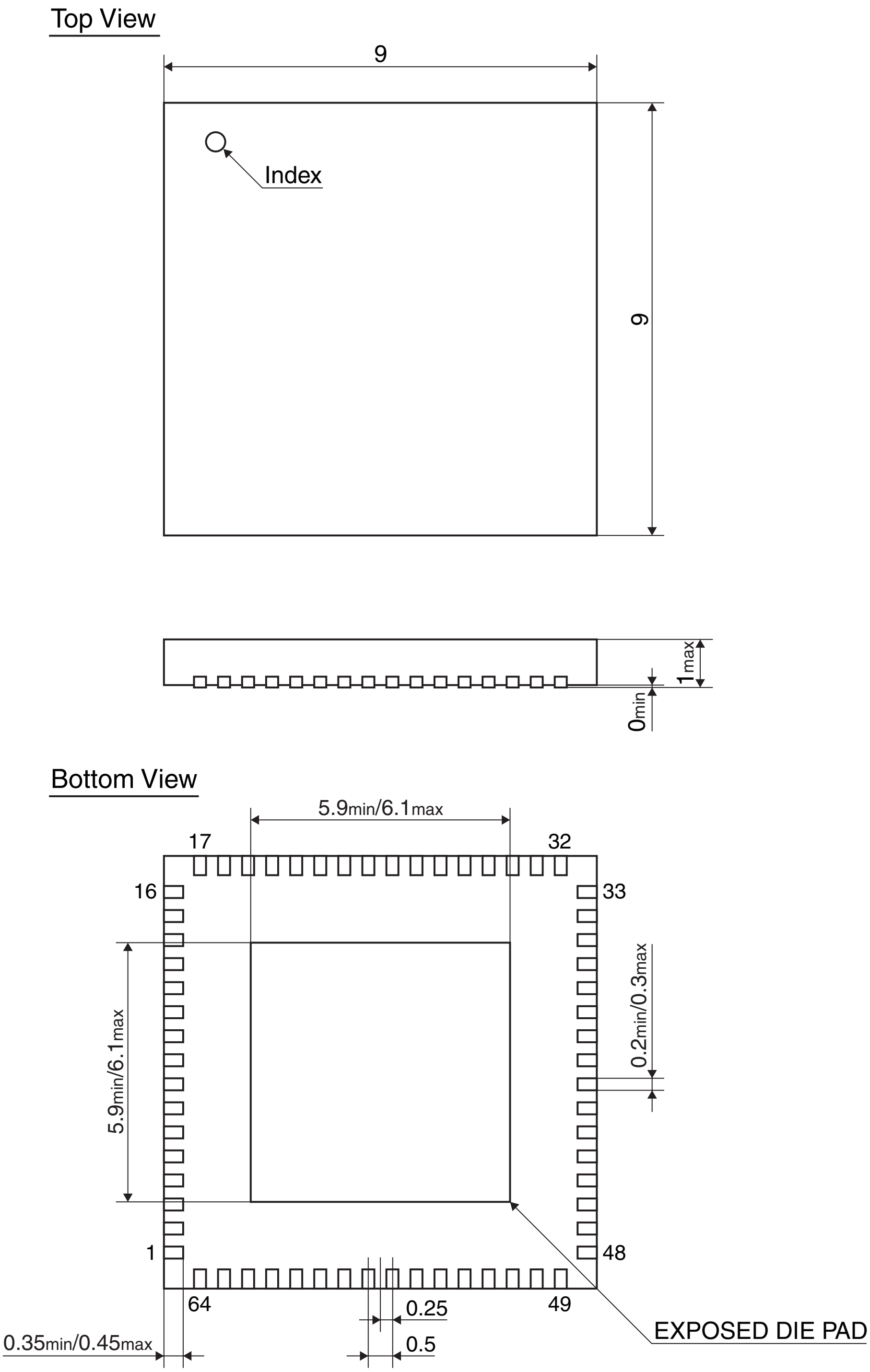


Figure 22.1 SQFN9-64PIN Package Dimensions

* The potential of the EXPOSED DIE PAD is the same as that of the substrate potential (V_{SS}) on the back of the IC.

QFP13-64PIN (P-LQFP064-1010-0.50)

(Unit: mm)

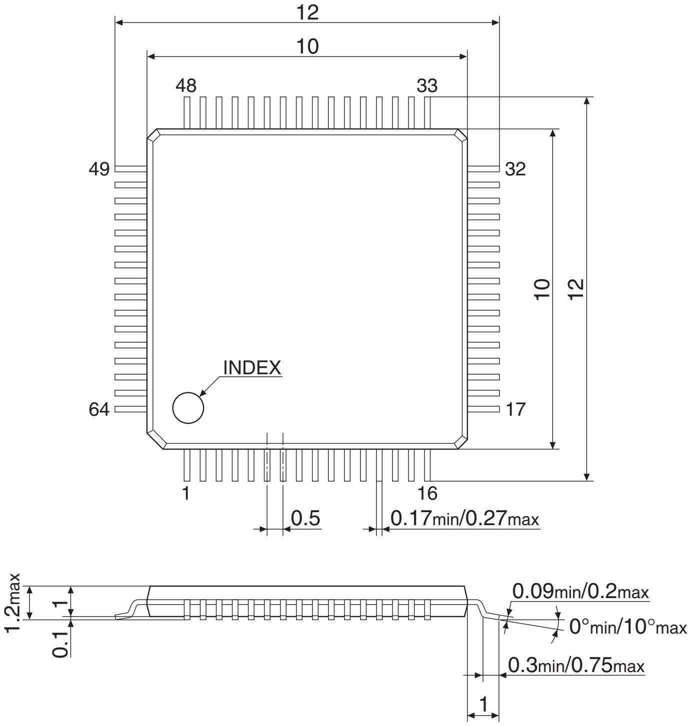


Figure 22.2 TQFP13-64PIN Package Dimensions

QFP14-80PIN (P-LQFP080-1212-0.50)

(Unit: mm)

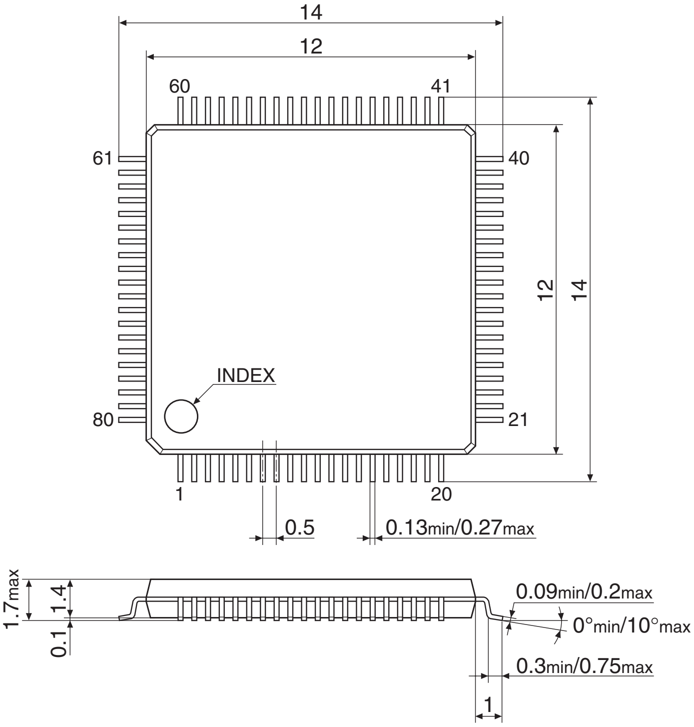


Figure 22.3 QFP14-80PIN Package Dimensions

QFP15-100PIN (P-LQFP100-1414-0.50)

(Unit: mm)

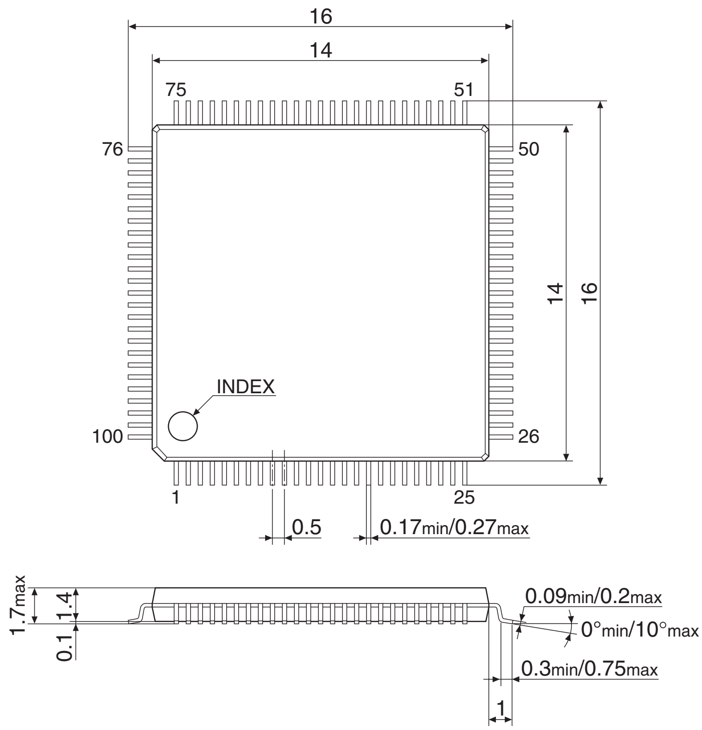


Figure 22.4 QFP15-100PIN Package Dimensions

Appendix A List of Peripheral Circuit Control Registers

Note: The following registers cannot be used in the 64-pin package model.
ITC: ITCLV8–9 registers (0x4090–0x4092), PPORT: P3*xxxx* registers (0x4230–0x423c)

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.
		7–3	–	0x06	–	R	–
		2–0	IRAMSZ[2:0]	0x3	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	

0x4020–0x4026 Power Generator (PWG2)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4020	PWGCTL (PWG2 Control Register)	15–8	–	0x00	–	R	–
		7–3	–	0x00	–	R	
		2–0	PWGMOD[2:0]	0x0	H0	R/WP	
0x4022	PWGTIM (PWG2 Timing Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1–0	DCCCLK[1:0]	0x0	H0	R/WP	
0x4024	PWGINTF (PWG2 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	MODCMPIF	0	H0	R/W	
0x4026	PWGINTE (PWG2 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	MODCMPIE	0	H0	R/W	

0x4040–0x4050 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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4042	CLGOSC (CLG Oscillation Control Register)	15–12	–	0x0	–	R	–
		11	EXOSCSLPC	1	H0	R/W	
		10	OSC3SLPC	1	H0	R/W	
		9	OSC1SLPC	1	H0	R/W	
		8	IOSCSLPC	1	H0	R/W	
		7–4	–	0x0	–	R	
		3	EXOSCEN	0	H0	R/W	
		2	OSC3EN	0	H0	R/W	
		1	OSC1EN	0	H0	R/W	
		0	IOSCEN	1	H0	R/W	
0x4044	CLGIOSC (CLG IOSC Control Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4	IOSCSTM	0	H0	R/WP	
		3–0	–	0x0	–	R	
0x4046	CLGOSC1 (CLG OSC1 Control Register)	15	–	0	–	R	–
		14	OSDRB	1	H0	R/WP	
		13	OSDEN	0	H0	R/WP	
		12	OSC1BUP	1	H0	R/WP	
		11	–	0	–	R	
		10–8	CGI1[2:0]	0x0	H0	R/WP	
		7–6	INV1B[1:0]	0x2	H0	R/WP	
		5–4	INV1N[1:0]	0x1	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	OSC1WT[1:0]	0x2	H0	R/WP	
0x4048	CLGOSC3 (CLG OSC3 Control Register)	15–12	–	0x0	–	R	–
		11–10	OSC3FQ[1:0]	0x3	H0	R/WP	
		9–8	OSC3MD[1:0]	0x0	H0	R/WP	
		7–6	–	0x0	–	R	
		5–4	OSC3INV[1:0]	0x1	H0	R/WP	
		3	–	0	–	R	
		2–0	OSC3WT[2:0]	0x0	H0	R/WP	
0x404c	CLGINTF (CLG Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7	–	0x0	–	R	
		6	(reserved)	0	H0	R	
		5	OSC1STPIF	0	H0	R/W	Cleared by writing 1.
		4	IOSCTEDIF	0	H0	R/W	
		3	–	0	–	R	–
		2	OSC3STAIF	0	H0	R/W	Cleared by writing 1.
		1	OSC1STAIF	0	H0	R/W	
		0	IOSCSTAIF	0	H0	R/W	
0x404e	CLGINTE (CLG Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6	(reserved)	0	H0	R	
		5	OSC1STPIE	0	H0	R/W	
		4	IOSCTEDIE	0	H0	R/W	
		3	–	0	–	R	
		2	OSC3STAIE	0	H0	R/W	
		1	OSC1STAIE	0	H0	R/W	
		0	IOSCSTAIE	0	H0	R/W	
0x4050	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	

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	Clock generator interrupt (ILVCLG)
		7–3	–	0x00	–	R	–
		2–0	ILV2[2:0]	0x0	H0	R/W	Power generator interrupt (ILVPWG2)
0x4084	ITCLV2 (ITC Interrupt Level Setup Register 2)	15–11	–	0x00	–	R	–
		10–8	ILV5[2:0]	0x0	H0	R/W	16-bit timer Ch.0 interrupt (ILVT16_0)
		7–3	–	0x00	–	R	–
		2–0	ILV4[2:0]	0x0	H0	R/W	Real-time clock interrupt (ILVRTCA_0)
0x4086	ITCLV3 (ITC Interrupt Level Setup Register 3)	15–11	–	0x00	–	R	–
		10–8	ILV7[2:0]	0x0	H0	R/W	16-bit timer Ch.1 interrupt (ILVT16_1)
		7–3	–	0x00	–	R	–
		2–0	ILV6[2:0]	0x0	H0	R/W	UART Ch.0 interrupt (ILVUART_0)
0x4088	ITCLV4 (ITC Interrupt Level Setup Register 4)	15–11	–	0x00	–	R	–
		10–8	ILV9[2:0]	0x0	H0	R/W	I²C interrupt (ILVI2C_0)
		7–3	–	0x00	–	R	–
		2–0	ILV8[2:0]	0x0	H0	R/W	Synchronous serial interface Ch.0 interrupt (ILVSPIA_0)
0x408a	ITCLV5 (ITC Interrupt Level Setup Register 5)	15–11	–	0x00	–	R	–
		10–8	ILV11[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.1 interrupt (ILVT16B_1)
		7–3	–	0x00	–	R	–
		2–0	ILV10[2:0]	0x0	H0	R/W	16-bit PWM timer Ch.0 interrupt (ILVT16B_0)
0x408c	ITCLV6 (ITC Interrupt Level Setup Register 6)	15–11	–	0x00	–	R	–
		10–8	ILV13[2:0]	0x0	H0	R/W	16-bit timer Ch.2 interrupt (ILVT16_2)
		7–3	–	0x00	–	R	–
		2–0	ILV12[2:0]	0x0	H0	R/W	UART Ch.1 interrupt (ILVUART_1)
0x408e	ITCLV7 (ITC Interrupt Level Setup Register 7)	15–11	–	0x00	–	R	–
		10–8	ILV15[2:0]	0x0	H0	R/W	LCD driver interrupt (ILVLCD8B)
		7–3	–	0x00	–	R	–
		2–0	ILV14[2:0]	0x0	H0	R/W	Sound generator interrupt (ILVSNDA_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	R/F converter Ch.3 interrupt (ILVRFC_3)
		7–3	–	0x00	–	R	–
		2–0	ILV18[2:0]	0x0	H0	R/W	R/F converter Ch.2 interrupt (ILVRFC_2)

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	–

0x40c0–0x40d2 Real-time Clock (RTCA)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x40c0	RTCCTL (RTC Control Register)	15	RTCTRMBSY	0	H0	R	–
		14–8	RTCTRM[6:0]	0x00	H0	W	Read as 0x00.
		7	–	0	–	R	–
		6	RTCBSY	0	H0	R	
		5	RTCHLD	0	H0	R/W	Cleared by setting the RTCCTL.RTCRST bit to 1.
		4	RTC24H	0	H0	R/W	–
		3	–	0	–	R	
		2	RTCADJ	0	H0	R/W	Cleared by setting the RTCCTL.RTCRST bit to 1.
		1	RTCRST	0	H0	R/W	–
		0	RTCRUN	0	H0	R/W	
0x40c2	RTCALM1 (RTC Second Alarm Register)	15	–	0	–	R	–
		14–12	RTCSHA[2:0]	0x0	H0	R/W	
		11–8	RTCSLA[3:0]	0x0	H0	R/W	
		7–0	–	0x00	–	R	
0x40c4	RTCALM2 (RTC Hour/Minute Alarm Register)	15	–	0	–	R	–
		14	RTCAPA	0	H0	R/W	
		13–12	RTCHHA[1:0]	0x0	H0	R/W	
		11–8	RTCHLA[3:0]	0x0	H0	R/W	
		7	–	0	–	R	
		6–4	RTCMIHA[2:0]	0x0	H0	R/W	
		3–0	RTCMILA[3:0]	0x0	H0	R/W	
0x40c6	RTCSWCTL (RTC Stopwatch Control Register)	15–12	BCD10[3:0]	0x0	H0	R	–
		11–8	BCD100[3:0]	0x0	H0	R	
		7–5	–	0x0	–	R	
		4	SWRST	0	H0	W	Read as 0.
		3–1	–	0x0	–	R	–
		0	SWRUN	0	H0	R/W	
0x40c8	RTCSEC (RTC Second/1Hz Register)	15	–	0	–	R	–
		14–12	RTCSH[2:0]	0x0	H0	R/W	
		11–8	RTCSL[3:0]	0x0	H0	R/W	
		7	RTC1HZ	0	H0	R	Cleared by setting the RTCCTL.RTCRST bit to 1.
		6	RTC2HZ	0	H0	R	
		5	RTC4HZ	0	H0	R	
		4	RTC8HZ	0	H0	R	
		3	RTC16HZ	0	H0	R	
		2	RTC32HZ	0	H0	R	
		1	RTC64HZ	0	H0	R	
		0	RTC128HZ	0	H0	R	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x40ca	RTCHUR (RTC Hour/Minute Register)	15	–	0	–	R	–
		14	RTCAP	0	H0	R/W	
		13–12	RTCHH[1:0]	0x1	H0	R/W	
		11–8	RTCHL[3:0]	0x2	H0	R/W	
		7	–	0	–	R	
		6–4	RTCMIH[2:0]	0x0	H0	R/W	
		3–0	RTCMIL[3:0]	0x0	H0	R/W	
0x40cc	RTCMON (RTC Month/Day Register)	15–13	–	0x0	–	R	–
		12	RTCMOH	0	H0	R/W	
		11–8	RTCMOL[3:0]	0x1	H0	R/W	
		7–6	–	0x0	–	R	
		5–4	RTCDH[1:0]	0x0	H0	R/W	
		3–0	RTCDL[3:0]	0x1	H0	R/W	
0x40ce	RTCYAR (RTC Year/Week Register)	15–11	–	0x00	–	R	–
		10–8	RTCWK[2:0]	0x0	H0	R/W	
		7–4	RTCYH[3:0]	0x0	H0	R/W	
		3–0	RTCYL[3:0]	0x0	H0	R/W	
0x40d0	RTCINTF (RTC Interrupt Flag Register)	15	RTCTRMIF	0	H0	R/W	Cleared by writing 1.
		14	SW1IF	0	H0	R/W	
		13	SW10IF	0	H0	R/W	
		12	SW100IF	0	H0	R/W	
		11–9	–	0x0	–	R	–
		8	ALARMIF	0	H0	R/W	Cleared by writing 1.
		7	1DAYIF	0	H0	R/W	
		6	1HURIF	0	H0	R/W	
		5	1MINIF	0	H0	R/W	
		4	1SECIF	0	H0	R/W	
		3	1_2SECIF	0	H0	R/W	
		2	1_4SECIF	0	H0	R/W	
		1	1_8SECIF	0	H0	R/W	
		0	1_32SECIF	0	H0	R/W	
0x40d2	RTCINTE (RTC Interrupt Enable Register)	15	RTCTRMIE	0	H0	R/W	–
		14	SW1IE	0	H0	R/W	
		13	SW10IE	0	H0	R/W	
		12	SW100IE	0	H0	R/W	
		11–9	–	0x0	–	R	
		8	ALARMIE	0	H0	R/W	
		7	1DAYIE	0	H0	R/W	
		6	1HURIE	0	H0	R/W	
		5	1MINIE	0	H0	R/W	
		4	1SECIE	0	H0	R/W	
		3	1_2SECIE	0	H0	R/W	
		2	1_4SECIE	0	H0	R/W	
		1	1_8SECIE	0	H0	R/W	
		0	1_32SECIE	0	H0	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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4102	SVDCTL (SVD Control Register)	15	–	0	–	R	–
		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]	0x1e	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	
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]	0x1	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	–	0	–	R	–
		14–8	P0OUT[6:0]	0x00	H0	R/W	
		7	–	0	–	R	
		6–0	P0IN[6:0]	0x00	H0	R	
0x4202	P0IOEN (P0 Port Enable Register)	15	–	0	–	R	–
		14–8	P0IEN[6:0]	0x00	H0	R/W	
		7	–	0	–	R	
		6–0	P0OEN[6:0]	0x00	H0	R/W	
0x4204	P0RCTL (P0 Port Pull-up/down Control Register)	15	–	0	–	R	–
		14–8	P0PDPU[6:0]	0x00	H0	R/W	
		7	–	0	–	R	
		6–0	P0REN[6:0]	0x00	H0	R/W	
0x4206	P0INTF (P0 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	P0IF[6:0]	0x00	H0	R/W	Cleared by writing 1.
0x4208	P0INTCTL (P0 Port Interrupt Control Register)	15	–	0	–	R	–
		14–8	P0EDGE[6:0]	0x00	H0	R/W	
		7	–	0	–	R	
		6–0	P0IE[6:0]	0x00	H0	R/W	
0x420a	P0CHATEN (P0 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	P0CHATEN[6:0]	0x00	H0	R/W	
0x420c	P0MODSEL (P0 Port Mode Select Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	P0SEL[6:0]	0x00	H0	R/W	
0x420e	P0FNCSEL (P0 Port Function Select Register)	15–14	–	0x0	–	R	–
		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]	0x00	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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
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]	0x00	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	
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	
0x4226	P2INTF (P2 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–0	P2IF[7:0]	0x00	H0	R/W	Cleared by writing 1.
0x4228	P2INTCTL (P2 Port Interrupt Control Register)	15–8	P2EDGE[7:0]	0x00	H0	R/W	–
		7–0	P2IE[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	P3OUT[7:5] and P3IN[7:5] are reserved bits in the 80-pin package.
		7–0	P3IN[7:0]	0x00	H0	R	
0x4232	P3IOEN (P3 Port Enable Register)	15–8	P3IEN[7:0]	0x00	H0	R/W	P3IEN[7:5] and P3OEN[7:5] are reserved bits in the 80-pin package.
		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	P3PDPU[7:5] and P3REN[7:5] are reserved bits in the 80-pin package.
		7–0	P3REN[7:0]	0x00	H0	R/W	
0x4236	P3INTF (P3 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–0	P3IF[7:0]	0x00	H0	R/W	Cleared by writing 1. P3IF[7:5] are reserved bits in the 80-pin package.
0x4238	P3INTCTL (P3 Port Interrupt Control Register)	15–8	P3EDGE[7:0]	0x00	H0	R/W	P3EDGE[7:5] and P3IE[7:5] are reserved bits in the 80-pin package.
		7–0	P3IE[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	P3SEL[7:5] are reserved bits in the 80-pin package.
0x42d0	PDDAT (Pd Port Data Register)	15–13	–	0x0	–	R	–
		12–8	PDOUT[4:0]	0x00	H0	R/W	
		7–5	–	0x0	–	R	
		4–3	PDIN[4:3]	x	H0	R	
		2	–	0	–	R	
		1–0	PDIN[1:0]	x	H0	R	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x42d2	PDIOEN (Pd Port Enable Register)	15–13	–	0x0	–	R	–
		12–11	PDIEN[4:3]	0x0	H0	R/W	
		10	(reserved)	0	H0	R/W	
		9–8	PDIEN[1:0]	0x0	H0	R/W	
		7–5	–	0x0	–	R	
		4–0	PDOEN[4:0]	0x00	H0	R/W	
0x42d4	PDRCTL (Pd Port Pull-up/down Control Register)	15–13	–	0x0	–	R	–
		12–11	PDPDPU[4:3]	0x0	H0	R/W	
		10	(reserved)	0	H0	R/W	
		9–8	PDPDPU[1:0]	0x0	H0	R/W	
		7–5	–	0x0	–	R	
		4–3	PDREN[4:3]	0x0	H0	R/W	
		2	(reserved)	0	H0	R/W	
		1–0	PDREN[1:0]	0x0	H0	R/W	
0x42dc	PDMODSEL (Pd Port Mode Select Register)	15–8	–	0x00	–	R	–
		7–5	–	0x0	–	R	
		4–0	PDSEL[4:0]	0x07	H0	R/W	
0x42de	PDFNCSEL (Pd Port Function Select Register)	15–10	–	0x00	–	R	–
		9–8	PD4MUX[1:0]	0x0	H0	R/W	
		7–6	PD3MUX[1:0]	0x0	H0	R/W	
		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	–
		7–4	–	0x0	–	R	
		3	P3INT	0	H0	R	Reserved bit in the 64-pin package
		2	P2INT	0	H0	R	–
		1	P1INT	0	H0	R	
		0	P0INT	0	H0	R	

0x4300–0x4316 Universal Port Multiplexer (UPMUX)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4300	P0UPMUX0 (P00–01 Universal Port Multiplexer Setting Register)	15–13	P01PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P01PERICH[1:0]	0x0	H0	R/W	
		10–8	P01PERISEL[2:0]	0x0	H0	R/W	
		7–5	P00PPFNC[2:0]	0x0	H0	R/W	
		4–3	P00PERICH[1:0]	0x0	H0	R/W	
		2–0	P00PERISEL[2:0]	0x0	H0	R/W	
0x4302	P0UPMUX1 (P02–03 Universal Port Multiplexer Setting Register)	15–13	P03PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P03PERICH[1:0]	0x0	H0	R/W	
		10–8	P03PERISEL[2:0]	0x0	H0	R/W	
		7–5	P02PPFNC[2:0]	0x0	H0	R/W	
		4–3	P02PERICH[1:0]	0x0	H0	R/W	
		2–0	P02PERISEL[2:0]	0x0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x4304	P0UPMUX2 (P04–05 Universal Port Multiplexer Setting Register)	15–13	P05PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P05PERICH[1:0]	0x0	H0	R/W	
		10–8	P05PERISEL[2:0]	0x0	H0	R/W	
		7–5	P04PPFNC[2:0]	0x0	H0	R/W	
		4–3	P04PERICH[1:0]	0x0	H0	R/W	
		2–0	P04PERISEL[2:0]	0x0	H0	R/W	
0x4306	P0UPMUX3 (P06 Universal Port Multiplexer Setting Register)	15–8	–	0x00	–	R	–
		7–5	P06PPFNC[2:0]	0x0	H0	R/W	
		4–3	P06PERICH[1:0]	0x0	H0	R/W	
		2–0	P06PERISEL[2:0]	0x0	H0	R/W	
0x4308	P1UPMUX0 (P10–11 Universal Port Multiplexer Setting Register)	15–13	P11PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P11PERICH[1:0]	0x0	H0	R/W	
		10–8	P11PERISEL[2:0]	0x0	H0	R/W	
		7–5	P10PPFNC[2:0]	0x0	H0	R/W	
		4–3	P10PERICH[1:0]	0x0	H0	R/W	
		2–0	P10PERISEL[2:0]	0x0	H0	R/W	
0x430a	P1UPMUX1 (P12–13 Universal Port Multiplexer Setting Register)	15–13	P13PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P13PERICH[1:0]	0x0	H0	R/W	
		10–8	P13PERISEL[2:0]	0x0	H0	R/W	
		7–5	P12PPFNC[2:0]	0x0	H0	R/W	
		4–3	P12PERICH[1:0]	0x0	H0	R/W	
		2–0	P12PERISEL[2:0]	0x0	H0	R/W	
0x430c	P1UPMUX2 (P14–15 Universal Port Multiplexer Setting Register)	15–13	P15PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P15PERICH[1:0]	0x0	H0	R/W	
		10–8	P15PERISEL[2:0]	0x0	H0	R/W	
		7–5	P14PPFNC[2:0]	0x0	H0	R/W	
		4–3	P14PERICH[1:0]	0x0	H0	R/W	
		2–0	P14PERISEL[2:0]	0x0	H0	R/W	
0x430e	P1UPMUX3 (P16–17 Universal Port Multiplexer Setting Register)	15–13	P17PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P17PERICH[1:0]	0x0	H0	R/W	
		10–8	P17PERISEL[2:0]	0x0	H0	R/W	
		7–5	P16PPFNC[2:0]	0x0	H0	R/W	
		4–3	P16PERICH[1:0]	0x0	H0	R/W	
		2–0	P16PERISEL[2:0]	0x0	H0	R/W	
0x4310	P2UPMUX0 (P20–21 Universal Port Multiplexer Setting Register)	15–13	P21PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P21PERICH[1:0]	0x0	H0	R/W	
		10–8	P21PERISEL[2:0]	0x0	H0	R/W	
		7–5	P20PPFNC[2:0]	0x0	H0	R/W	
		4–3	P20PERICH[1:0]	0x0	H0	R/W	
		2–0	P20PERISEL[2:0]	0x0	H0	R/W	
0x4312	P2UPMUX1 (P22–23 Universal Port Multiplexer Setting Register)	15–13	P23PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P23PERICH[1:0]	0x0	H0	R/W	
		10–8	P23PERISEL[2:0]	0x0	H0	R/W	
		7–5	P22PPFNC[2:0]	0x0	H0	R/W	
		4–3	P22PERICH[1:0]	0x0	H0	R/W	
		2–0	P22PERISEL[2:0]	0x0	H0	R/W	
0x4314	P2UPMUX2 (P24–25 Universal Port Multiplexer Setting Register)	15–13	P25PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P25PERICH[1:0]	0x0	H0	R/W	
		10–8	P25PERISEL[2:0]	0x0	H0	R/W	
		7–5	P24PPFNC[2:0]	0x0	H0	R/W	
		4–3	P24PERICH[1:0]	0x0	H0	R/W	
		2–0	P24PERISEL[2:0]	0x0	H0	R/W	
0x4316	P2UPMUX3 (P26–27 Universal Port Multiplexer Setting Register)	15–13	P27PPFNC[2:0]	0x0	H0	R/W	–
		12–11	P27PERICH[1:0]	0x0	H0	R/W	
		10–8	P27PERISEL[2:0]	0x0	H0	R/W	
		7–5	P26PPFNC[2:0]	0x0	H0	R/W	
		4–3	P26PERICH[1:0]	0x0	H0	R/W	
		2–0	P26PERISEL[2:0]	0x0	H0	R/W	

0x4380–0x438e UART (UART) Ch.0

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	
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 Synchronous Serial Interface (SPIA)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x43b0	SPI0MOD (SPIA Ch.0 Mode Register)	15–12	–	0x0	–	R	–
		11–8	CHLN[3:0]	0x7	H0	R/W	
		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 (SPIA 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 (SPIA Ch.0 Transmit Data Register)	15–0	TXD[15:0]	0x0000	H0	R/W	–
0x43b6	SPI0RXD (SPIA Ch.0 Receive Data Register)	15–0	RXD[15:0]	0x0000	H0	R	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x43b8	SPI0INTF (SPIA Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7	BSY	0	H0	R	
		6–4	–	0x0	–	R	
		3	OEIF	0	H0/S0	R/W	Cleared by writing 1.
		2	TENDIF	0	H0/S0	R/W	
		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 (SPIA Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	OEIE	0	H0	R/W	
		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	
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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
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	

0x5000–0x501a 16-bit PWM Timer (T16B) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5000	T16B0CLK (T16B 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	
		2–0	CLKSRC[2:0]	0x0	H0	R/W	
0x5002	T16B0CTL (T16B Ch.0 Counter Control Register)	15–9	–	0x00	–	R	–
		8	MAXBSY	0	H0	R	
		7–6	–	0x0	–	R	
		5–4	CNTMD[1:0]	0x0	H0	R/W	
		3	ONEST	0	H0	R/W	
		2	RUN	0	H0	R/W	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5004	T16B0MC (T16B Ch.0 Max Counter Data Register)	15–0	MC[15:0]	0xffff	H0	R/W	–
0x5006	T16B0TC (T16B Ch.0 Timer Counter Data Register)	15–0	TC[15:0]	0x0000	H0	R	–
0x5008	T16B0CS (T16B Ch.0 Counter Status Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	CAPI1	0	H0	R	
		2	CAPI0	0	H0	R	
		1	UP_DOWN	1	H0	R	
		0	BSY	0	H0	R	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x500a	T16B0INTF (T16B Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPOW1IF	0	H0	R/W	Cleared by writing 1.
		4	CMPCAP1IF	0	H0	R/W	
		3	CAPOW0IF	0	H0	R/W	
		2	CMPCAP0IF	0	H0	R/W	
		1	CNTMAXIF	0	H0	R/W	
		0	CNTZEROIF	0	H0	R/W	
0x500c	T16B0INTE (T16B Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPOW1IE	0	H0	R/W	
		4	CMPCAP1IE	0	H0	R/W	
		3	CAPOW0IE	0	H0	R/W	
		2	CMPCAP0IE	0	H0	R/W	
		1	CNTMAXIE	0	H0	R/W	
		0	CNTZEROIE	0	H0	R/W	
0x5010	T16B0CCCTL0 (T16B Ch.0 Compare/Capture 0 Control Register)	15	SCS	0	H0	R/W	–
		14–12	CBUFMD[2:0]	0x0	H0	R/W	
		11–10	CAPIS[1:0]	0x0	H0	R/W	
		9–8	CAPTRG[1:0]	0x0	H0	R/W	
		7	–	0	–	R	
		6	TOUTMT	0	H0	R/W	
		5	TOUTO	0	H0	R/W	
		4–2	TOUTMD[2:0]	0x0	H0	R/W	
		1	TOUTINV	0	H0	R/W	
		0	CCMD	0	H0	R/W	
0x5012	T16B0CCR0 (T16B Ch.0 Compare/Capture 0 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x5018	T16B0CCCTL1 (T16B Ch.0 Compare/Capture 1 Control Register)	15	SCS	0	H0	R/W	–
		14–12	CBUFMD[2:0]	0x0	H0	R/W	
		11–10	CAPIS[1:0]	0x0	H0	R/W	
		9–8	CAPTRG[1:0]	0x0	H0	R/W	
		7	–	0	–	R	
		6	TOUTMT	0	H0	R/W	
		5	TOUTO	0	H0	R/W	
		4–2	TOUTMD[2:0]	0x0	H0	R/W	
		1	TOUTINV	0	H0	R/W	
		0	CCMD	0	H0	R/W	
0x501a	T16B0CCR1 (T16B Ch.0 Compare/Capture 1 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–

0x5040–0x505a 16-bit PWM Timer (T16B) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5040	T16B1CLK (T16B 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	–	0	–	R	
		2–0	CLKSRC[2:0]	0x0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5042	T16B1CTL (T16B Ch.1 Counter Control Register)	15–9	–	0x00	–	R	–
		8	MAXBSY	0	H0	R	
		7–6	–	0x0	–	R	
		5–4	CNTMD[1:0]	0x0	H0	R/W	
		3	ONEST	0	H0	R/W	
		2	RUN	0	H0	R/W	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5044	T16B1MC (T16B Ch.1 Max Counter Data Register)	15–0	MC[15:0]	0xffff	H0	R/W	–
0x5046	T16B1TC (T16B Ch.1 Timer Counter Data Register)	15–0	TC[15:0]	0x0000	H0	R	–
0x5048	T16B1CS (T16B Ch.1 Counter Status Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	CAPI1	0	H0	R	
		2	CAPI0	0	H0	R	
		1	UP_DOWN	1	H0	R	
		0	BSY	0	H0	R	
0x504a	T16B1INTF (T16B Ch.1 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPOW1IF	0	H0	R/W	Cleared by writing 1.
		4	CMPCAP1IF	0	H0	R/W	
		3	CAPOW0IF	0	H0	R/W	
		2	CMPCAP0IF	0	H0	R/W	
		1	CNTMAXIF	0	H0	R/W	
		0	CNTZEROIF	0	H0	R/W	
0x504c	T16B1INTE (T16B Ch.1 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	CAPOW1IE	0	H0	R/W	
		4	CMPCAP1IE	0	H0	R/W	
		3	CAPOW0IE	0	H0	R/W	
		2	CMPCAP0IE	0	H0	R/W	
		1	CNTMAXIE	0	H0	R/W	
		0	CNTZEROIE	0	H0	R/W	
0x5050	T16B1CCCTL0 (T16B Ch.1 Compare/Capture 0 Control Register)	15	SCS	0	H0	R/W	–
		14–12	CBUFMD[2:0]	0x0	H0	R/W	
		11–10	CAPIS[1:0]	0x0	H0	R/W	
		9–8	CAPTRG[1:0]	0x0	H0	R/W	
		7	–	0	–	R	
		6	TOUTMT	0	H0	R/W	
		5	TOUTO	0	H0	R/W	
		4–2	TOUTMD[2:0]	0x0	H0	R/W	
		1	TOUTINV	0	H0	R/W	
		0	CCMD	0	H0	R/W	
0x5052	T16B1CCR0 (T16B Ch.1 Compare/Capture 0 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5058	T16B1CCCTL1 (T16B Ch.1 Compare/Capture 1 Control Register)	15	SCS	0	H0	R/W	–
		14–12	CBUFMD[2:0]	0x0	H0	R/W	
		11–10	CAPIS[1:0]	0x0	H0	R/W	
		9–8	CAPTRG[1:0]	0x0	H0	R/W	
		7	–	0	–	R	
		6	TOUTMT	0	H0	R/W	
		5	TOUTO	0	H0	R/W	
		4–2	TOUTMD[2:0]	0x0	H0	R/W	
		1	TOUTINV	0	H0	R/W	
		0	CCMD	0	H0	R/W	
0x505a	T16B1CCR1 (T16B Ch.1 Compare/Capture 1 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–

0x5200–0x520e UART (UART) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5200	UA1CLK (UART Ch.1 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	
0x5202	UA1MOD (UART Ch.1 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	
0x5204	UA1BR (UART Ch.1 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	
0x5206	UA1CTL (UART Ch.1 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x5208	UA1TXD (UART Ch.1 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x520a	UA1RXD (UART Ch.1 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
0x520c	UA1INTF (UART Ch.1 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 UA1RXD 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 UA1RXD register.
		1	RB1FIF	0	H0/S0	R	
		0	TBEIF	1	H0/S0	R	Cleared by writing to the UA1TXD register.
0x520e	UA1INTE (UART Ch.1 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	

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	

0x5300–0x530a Sound Generator (SNDA)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5300	SNDCLK (SNDA Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7	–	0	–	R	
		6–4	CLKDIV[2:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5302	SNDSEL (SNDA Select Register)	15–12	–	0x0	–	R	–
		11–8	STIM[3:0]	0x0	H0	R/W	
		7–3	–	0x00	–	R	
		2	SINV	0	H0	R/W	
		1–0	MOSEL[1:0]	0x0	H0	R/W	
0x5304	SNDCTL (SNDA Control Register)	15–9	–	0x00	–	R	–
		8	SSTP	0	H0	R/W	
		7–1	–	0x00	–	R	
		0	MODEN	0	H0	R/W	
0x5306	SNDDAT (SNDA Data Register)	15	MDTI	0	H0	R/W	–
		14	MDRS	0	H0	R/W	
		13–8	SLEN[5:0]	0x00	H0	R/W	
		7–0	SFRQ[7:0]	0xff	H0	R/W	
0x5308	SNDINTF (SNDA Interrupt Flag Register)	15–9	–	0x00	–	R	–
		8	SBSY	0	H0	R	
		7–2	–	0x00	–	R	
		1	EMIF	1	H0	R	Cleared by writing to the SNDDAT register.
		0	EDIF	0	H0	R/W	Cleared by writing 1 or writing to the SNDDAT register.
0x530a	SNDINTE (SNDA Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	EMIE	0	H0	R/W	
		0	EDIE	0	H0	R/W	

0x5400–0x5412 LCD Driver (LCD8B)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5400	LCD8CLK (LCD8B Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	1	H0	R/W	
		7	–	0	–	R	
		6–4	CLKDIV[2:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x5402	LCD8CTL (LCD8B Control Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	MODEN	0	H0	R/W	
0x5404	LCD8TIM1 (LCD8B Timing Control Register 1)	15–12	–	0x0	–	R	–
		11–8	FRMCNT[3:0]	0x3	H0	R/W	
		7–3	–	0x00	–	R	
		2–0	LDUTY[2:0]	0x7	H0	R/W	
0x5406	LCD8TIM2 (LCD8B Timing Control Register 2)	15–10	–	0x00	–	R	–
		9–8	BSTC[1:0]	0x1	H0	R/W	
		7–3	–	0x00	–	R	
		2–0	NLINE[2:0]	0x0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5408	LCD8PWR (LCD8B Power Control Register)	15–13	–	0x0	–	R	–
		12–8	LC[4:0]	0x00	H0	R/W	
		7–5	–	0x0	–	R	
		4	BSTEN	0	H0	R/W	
		3	BISEL	1	H0	R/W	
		2	HVLD	0	H0	R/W	
		1	VCSEL	0	H0	R/W	
		0	VCEN	0	H0	R/W	
0x540a	LCD8DSP (LCD8B Display Control Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6	SEGREV	1	H0	R/W	
		5	COMREV	1	H0	R/W	
		4	DSPREV	1	H0	R/W	
		3	–	0	–	R	
		2	DSPAR	0	H0	R/W	
		1–0	DSPC[1:0]	0x0	H0	R/W	
0x540c	LCD8COMC0 (LCD8B COM Pin Control Register 0)	15–8	–	0x00	–	R	–
		7	COM7DEN	1	H0	R/W	
		6	COM6DEN	1	H0	R/W	
		5	COM5DEN	1	H0	R/W	
		4	COM4DEN	1	H0	R/W	
		3	COM3DEN	1	H0	R/W	
		2	COM2DEN	1	H0	R/W	
		1	COM1DEN	1	H0	R/W	
		0	COM0DEN	1	H0	R/W	
0x5410	LCD8INTF (LCD8B Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	FRMIF	0	H0	R/W	Cleared by writing 1.
0x5412	LCD8INTE (LCD8B Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	FRMIE	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)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		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	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
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)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6	EVTEN	0	H0	R/W	
		5–4	SMODE[1:0]	0x0	H0	R/W	Setting to 0x1 is invalid.
		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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
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	

0x5480–0x5490 R/F Converter (RFC) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x5480	RFC2CLK (RFC Ch.2 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	
0x5482	RFC2CTL (RFC Ch.2 Control Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6	EVTEN	0	H0	R/W	
		5–4	SMODE[1:0]	0x0	H0	R/W	Setting to 0x1 is invalid.
		3–1	–	0x0	–	R	–
		0	MODEN	0	H0	R/W	
0x5484	RFC2TRG (RFC Ch.2 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	
0x5486	RFC2MCL (RFC Ch.2 Measurement Counter Low Register)	15–0	MC[15:0]	0x0000	H0	R/W	–
0x5488	RFC2MCH (RFC Ch.2 Measurement Counter High Register)	15–8	–	0x00	–	R	–
		7–0	MC[23:16]	0x00	H0	R/W	
0x548a	RFC2TCL (RFC Ch.2 Time Base Counter Low Register)	15–0	TC[15:0]	0x0000	H0	R/W	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x548c	RFC2TCH (RFC Ch.2 Time Base Counter High Register)	15–8	–	0x00	–	R	–
		7–0	TC[23:16]	0x00	H0	R/W	
0x548e	RFC2INTF (RFC Ch.2 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	
0x5490	RFC2INTE (RFC Ch.2 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	

0x54a0–0x54b0 R/F Converter (RFC) Ch.3

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x54a0	RFC3CLK (RFC Ch.3 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	
0x54a2	RFC3CTL (RFC Ch.3 Control Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6	EVTEN	0	H0	R/W	
		5–4	SMODE[1:0]	0x0	H0	R/W	Setting to 0x1 is invalid.
		3–1	–	0x0	–	R	–
		0	MODEN	0	H0	R/W	
0x54a4	RFC3TRG (RFC Ch.3 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	
0x54a6	RFC3MCL (RFC Ch.3 Measurement Counter Low Register)	15–0	MC[15:0]	0x0000	H0	R/W	–
0x54a8	RFC3MCH (RFC Ch.3 Measurement Counter High Register)	15–8	–	0x00	–	R	–
		7–0	MC[23:16]	0x00	H0	R/W	
0x54aa	RFC3TCL (RFC Ch.3 Time Base Counter Low Register)	15–0	TC[15:0]	0x0000	H0	R/W	–
0x54ac	RFC3TCH (RFC Ch.3 Time Base Counter High Register)	15–8	–	0x00	–	R	–
		7–0	TC[23:16]	0x00	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x54ae	RFC3INTF (RFC Ch.3 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	
0x54b0	RFC3INTE (RFC Ch.3 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	

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 0fc0	H0	R	

Appendix B Power Saving

Current consumption will vary dramatically, depending on CPU operating mode, operation clock frequency, peripheral circuits being operated, and power generator 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	PWG2	OSC1	IOSC/ OSC3/ EXOSC	RTCA	CPU	Current consumption listed in electrical characteristics
Standby	↑ Low	Super economy/ Economy	OFF	OFF	OFF	SLEEP	I_{SLP}
Clock counting			ON		ON	SLEEP or HALT	I_{HALT2}
Low-speed processing						OSC1 RUN	I_{RUN20}
Peripheral circuit operations	High ↓	Normal		ON		SLEEP or HALT	I_{HALT1}
High-speed processing						IOSC/OSC3/EXOSC RUN	I_{RUN10}

Note: The 64-pin package model does not support super economy mode.

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.

PWGCTL.PWGMOD[2:0] bits of the power generator

If the PWGCTL.PWGMOD[2:0] 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_{SLP} that is listed in “Electrical Characteristics.” Set the PWGCTL.PWGMOD[2:0] bits to 0x5 (super economy mode), 0x3 (economy mode), or 0x0 (automatic mode) before executing the slp instruction.

CLGOSC.IOSCSLPC/OSC1SLPC/OSC3SLPC/EXOSCSLPC bits of the clock generator

Setting the CLGOSC.IOSCSLPC, OSC1SLPC, OSC3SLPC, 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. Note that the real-time clock has no MODEN bit, therefore, current consumption does not vary if it is counting or idle.

OSC1 oscillator circuit configurations

The OSC1 oscillator circuit provides some configuration items to support various crystal resonators with ranges from cylinder type through surface-mount type. These configurations trade off current consumption for performance as shown below.

- The lower oscillation inverter gain setting (CLGOSC1.INV1B[1:0]/INV1N[1:0] bits) decreases current consumption.
- The lower OSC1 internal gate capacitance setting (CLGOSC1.CGI1[2:0] bits) decreases current consumption.
- Using lower OSC1 external gate and drain capacitances decreases current consumption.
- Using a crystal 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.

OSC3 (crystal/ceramic) oscillator circuit configurations

The OSC3 (crystal/ceramic) 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 (CLGOSC3.OSC3INV[1:0] bits) decreases current consumption.
- Using lower OSC3 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.

LCD driver configurations

- Setting the LCD voltage regulator to operate with the V_{C1} reference voltage (LCD8PWR.VCSEL bit = 0) increases current consumption. Select the V_{C2} reference voltage (LCD8PWR.VCSEL bit = 1) when the power supply voltage V_{DD} is 2.2 V or higher.
- The lower booster clock frequency setting (LCD8TIM2.BSTC[1:0] bits) for the LCD voltage booster decreases current consumption. Note, however, that the load characteristic becomes worse.
- Setting the LCD voltage regulator into heavy load protection mode (LCD8PWR.HVLD bit = 1) increases current consumption. Heavy load protection mode should be set only when the display becomes unstable.

Appendix C Mounting Precautions

This section describes various precautions for circuit board design and IC mounting.

OSC1/OSC3 oscillator circuit

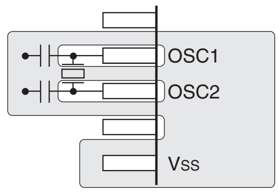
- Oscillation characteristics depend on factors such as components used (resonator, C_G, C_D) and circuit board patterns. In particular, with crystal resonators, select the appropriate capacitors (C_G, C_D) only after fully evaluating components actually mounted on the circuit board.
- Oscillator clock 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 V_{SS} to shield the 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.

Sample V_{SS} pattern (OSC1)



(4) After implementing these precautions, check the FOUT pin output clock waveform by running the actual application program within the product.
For the OSC1 waveform, 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.
For the OSC3 waveform, confirm that the frequency is as designed, is free of noise, and has minimal jitter.

Failure to observe precautions (1) to (3) adequately may lead to noise in OSC1CLK and jitter in OSC3CLK. Noise in the OSC1CLK will destabilize timers that use OSC1CLK as well as CPU Core operations. Jitter in the OSC3 output will reduce operating frequencies.

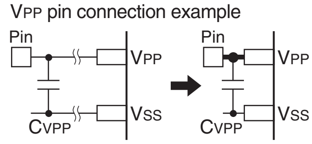
#RESET pin

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} ± 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.

V_{PP} pin connection example

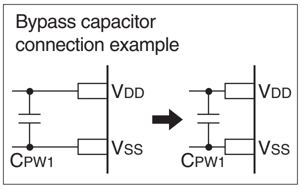


Power supply circuit

Sudden power supply fluctuations due to noise will cause malfunctions. Consider the following issues.

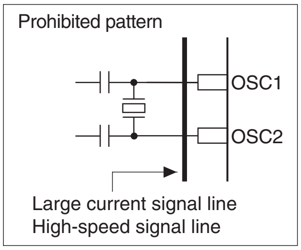
(1) Connections from the power supply to the V_{DD} and V_{SS} pins should be implemented via the shortest, thickest patterns possible.

(2) If a bypass capacitor is connected between V_{DD} and V_{SS}, connections between the V_{DD} and V_{SS} 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 SEG/COM lines and voltage boost/reduce 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 OSC1 oscillator circuit, OSC3 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).

(3) V_{C1-4}, C_{P1-4}, SEG*x*, and COM*x* pins
If the LCD driver is not used, these pins should be left open. The control registers should be fixed at the initial status (display off). The unused SEG*x* and COM*x* pins that are not required to connect should be left open even if the LCD driver is used.

(4) C_{V1-2} pins
If super economy mode is not used, the C_{V1} and C_{V2} pins should be left open. In this case, C_{PW3} can be omitted by connecting between the V_{DD} and V_{D2} pins directly. When these pins are not short-circuited, C_{PW3} is required even if super economy mode is not used.

Treatment of exposed die pad

The exposed die pad of the packages such as QFN has the same potential as that of the substrate on the back of the IC. When mounting these packages on a circuit board, please note the following:

(1) When soldering exposed die pad to mounting board
Connect the exposed die pad with a wiring pattern that has the same potential as the substrate potential on the back of the IC, or do not connect it electrically (leave it open electrically). Even if connected to the same potential on the back of the IC, the power supply pins must be connected to the power source (the exposed die pad cannot be used as a power supply pad).

(2) When not soldering exposed die pad to mounting board
Do not place any signal wiring pattern on the exposed die pad area of the mounting board.

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, an IC malfunction may occur. 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 pwg2_handler              ; 0x06       0x18    PWG2
.long clg_handler               ; 0x07       0x1c    CLG
.long rtca_handler              ; 0x08       0x20    RTCA
.long t16_0_handler             ; 0x09       0x24    T16 ch0
.long uart_0_handler            ; 0x0a       0x28    UART ch0
.long t16_1_handler             ; 0x0b       0x2c    T16 ch1
.long spia_0_handler            ; 0x0c       0x30    SPIA
.long i2c_handler               ; 0x0d       0x34    I2C
.long t16b_0_handler            ; 0x0e       0x38    T16B ch0
.long t16b_1_handler            ; 0x0f       0x3c    T16B ch1
.long uart_1_handler            ; 0x10       0x40    UART ch1
.long t16_2_handler             ; 0x11       0x44    T16 ch2
.long snda_handler              ; 0x12       0x48    SNDA
.long lcd8b_handler             ; 0x13       0x4c    LCD8B
.long rfc_0_handler             ; 0x14       0x50    RFC ch0
.long rfc_1_handler             ; 0x15       0x54    RFC ch1
.long rfc_2_handler             ; 0x16       0x58    RFC ch2
.long rfc_3_handler             ; 0x17       0x5c    RFC ch3
.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, 0x0fc0                                            ...(4)

        ; ----- Memory controller ----------------
        Xld.a   %r1, 0x41b0     ; FLASHC register address

        ; Flash read wait cycle
        Xld.a   %r0, 0x00       ; 0x00 = No 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
412645700	All	New establishment
412645701	17-2	LCD8B: External Connections (Old) No description (New) Note: When the panel is connected, the LCD8CTL.MODEN bit must be set to 1 to bias the panel even if display is turned off.
	20-6	Electrical characteristics: OSC3 CR oscillation frequency-resistance characteristic (Old) f_{OSC3R} [MHz] (New) f_{OSC3R} [kHz]
412645702	1-1	Features: Table 1.1.1 Features (Old) Watchdog timer (WDT) | Generates NMI or watchdog timer reset. (New) Watchdog timer (WDT) | Generates watchdog timer reset.
	1-9 to 1-11	Features: Table 1.3.3.1 Pin Description Modified the table (P0[6:4], P1[7:0], P3]7:0], and PD[2:0] → Without tolerant fail-safe structure)
	2-3	Power Supply, Reset, and Clocks: Automatic mode (Old) 4. When the slp instruction is executed in normal mode (only OSC1 operates during SLEEP) After a lapse of 1 ms from transition to SLEEP mode, the hardware switches from normal mode to economy mode and sets the PWGINTF.MODCMPIF bit to 1. (New) 4. When the slp instruction is executed in normal mode (only OSC1 operates during SLEEP) After a lapse of 1 ms from transition to SLEEP mode, the hardware switches from normal mode to economy mode and sets the PWGINTF.MODCMPIF bit to 1. Note: The IC does not enter economy mode if a clock source other than OSC1 is active when the slp instruction is executed. Therefore, stop clock sources other than OSC1 before executing the slp instruction.
	2-5	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.
	2-15	Power Supply, Reset, and Clocks: Canceling HALT or SLEEP mode (Old) • NMI from the watchdog timer (New) • NMI
	2-18	Power Supply, Reset, and Clocks: CLG System Clock Control Register - Bit 15 WUPMD (Old) No description (New) Notes: • When the CLGSCLK.WUPMD bit = 1, the clock source enable bits (CLGOSC.EXOSCEN, CLGOSC.OSC1EN, CLGOSC.OSC3EN, CLGOSC.IOSCEN) 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.
		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.
	2-21	Power Supply, 2 Reset, and Clocks: CLG OSC3 Control Register Modified the register table (OSC3WT[2:0]: Initial = 0x6 → 0x0)
	5-1	ITC: Figure 5.1.1 ITC Configuration Modified the figure (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.)

REVISION HISTORY

Code No.	Page	Contents
412645702	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: 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).
	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.
	8-1	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 8.1.1 WDT Configuration Modified the figure (NMIXRST, STATNMI, and the NMI output were deleted.)
		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).
	8-2	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.

Code No.	Page	Contents
412645702	8-2	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.
	8-3	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
	8-4	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.
	10-3	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})
		Reading detection results (Old) • Power supply voltage (EXSVD) ≥ Comparison voltage when SVDINTF.SVDDT bit = 0 • Power supply voltage (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 (EXSVD) ≥ SVD detection voltage V_{SVD} when SVDINTF.SVDDT bit = 0 • Power supply voltage (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 ...
	10-5	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}.
	10-6	SVD: SVD Control Register - Bits 12–8 SVDC[4:0] (Old) These bits select a comparison voltage for detecting low voltage from among 30 levels. Table 10.6.3 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 from among 30 levels. Table 10.6.3 Setting of SVD Detection Voltage V_{SVD} SVDCTL.SVDC[4:0] bits | SVD detection voltage V_{SVD} [V]
	10-7	SVD: SVD Status and Interrupt Flag Register - Bit 8 SVDDT (Old) 1 (R): Power supply voltage (EXSVD) < comparison voltage 0 (R): Power supply voltage (EXSVD) ≥ comparison voltage (New) 1 (R): Power supply voltage (EXSVD) < SVD detection voltage V_{SVD} 0 (R): Power supply voltage (EXSVD) ≥ SVD detection voltage V_{SVD}
	10-8	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.

REVISION HISTORY

Code No.	Page	Contents
412645702	11-1	T16: Figure 11.1.1 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.
	11-6	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.
	11-7	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.
	14-14	I2C: Figure 14.4.7.1 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)
	15-27	T16B: T16B Ch.*n* Interrupt Enable Register (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.
	15-30	T16B: T16B Ch.*n* Compare/Capture *m* Data Register Modified the register table (CC[15:0]: Initial = 0xffff → 0x0000)
	20-4	Electrical Characteristics: #RESET pin characteristics Modified the table (The VT+ Max. value and VT- Min. value were modified.)
		Electrical Characteristics: POR characteristics Modified the table (The V_{RST}OP Typ. value was modified.)
	20-5	Electrical Characteristics: OSC3 oscillator circuit characteristics Modified the table (The f_{OSC3I} conditions (CLGOSC3.OSC3FQ[1:0] bit values) were corrected.)
	20-6	Electrical Characteristics: OSC3 internal oscillation frequency-temperature characteristic, OSC3 internal oscillation frequency-power supply voltage characteristic Corrected the graphs (The CLGOSC3.OSC3FQ[1:0] bit values were corrected.)
		Electrical Characteristics: EXOSC external clock input characteristics Modified the table (The VT+ Max. value, VT- Min. value, and ΔV_T Min. value were modified.)
	20-7	Electrical Characteristics: Input/Output Port (PPORT) Characteristics Modified the table (The VT+ Max. value, VT- Min. value, and ΔV_T Min. value were modified.)
	20-9	Electrical Characteristics: Supply Voltage Detector (SVD) Characteristics Modified the figure (SVDCTL.SVDSC[1:0] = 0x1f → SVDCTL.SVDC[4:0] = 0x1e)
	20-16	Electrical Characteristics: R/F Converter (RFC) Characteristics Modified the table (The VT+ Max. value, VT- Min. value, and ΔV_T Min. value were modified.)
	21-2	Basic External Connection Diagram: Sample external components (Old) X'tal3 | Crystal resonator | CA-301 (1 MHz) manufactured by Seiko Epson Corporation (New) X'tal3 | Crystal resonator | CA-301 (4 MHz) manufactured by Seiko Epson Corporation
	AP-A-2	List of Peripheral Circuit Control Registers: CLG OSC3 Control Register Modified the register table (OSC3WT[2:0]: Initial = 0x6 → 0x0)
	AP-A-4	List of Peripheral Circuit Control Registers: WDT Control Register Modified the register table (NMIXRST, STATNMI → Reserved)
	AP-A-15～ AP-A-17	List of Peripheral Circuit Control Registers: T16B Ch.*n* Compare/Capture *m* Data Register Modified the register table (CC[15:0]: Initial = 0xffff → 0x0000)
	AP-C-1	Mounting Precautions: V_{PP} pin (Old) Connect C_{VPP} to this pin if the V_{PP} voltage is not stable. (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} ± 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)

Code No.	Page	Contents
412645702	AP-C-2	Mounting Precautions: Unused pins (Old) (4) CV1–2 pins If super economy mode is not used, these pins should be left open. (New) (4) CV1–2 pins If super economy mode is not used, the C_{V1} and C_{V2} pins should be left open. In this case, C_{PW3} can be omitted by connecting between the V_{DD} and V_{D2} pins directly. When these pins are not short-circuited, C_{PW3} is required even if super economy mode is not used.
412645703	1-2 to 3	1.1 Features Added the following annotations to Table 1.1.1. I²C (I2C) *1 *1 The input filter in I2C (SDA and SCL inputs) does not comply with the standard for removing noise spikes less than 50 ns. SLEEPモード *2 *2 The RAM retains data even in SLEEP mode. Modified Table 1.1.1. Shipping form: A JEITA name was added to the package name.
	2-10	2.3.4 Operations Oscillation start time and oscillation stabilization waiting time Added the following description: The oscillation stabilization waiting time for the OSC1 oscillator circuit should be set to 16,384 OSC1CLK clocks or more.
	2-14	2.4.2 Transition between Operating Modes SLEEP mode Added the following description: The RAM retains data even in SLEEP mode.
	3-3	3.3.3 List of debugger input/output pins Added notes. Notes: • Do not drive the DCLK pin with a high level from outside (e.g. pulling up with a resistor). Also, do not connect (short-circuit) between the DCLK pin and another GPIO port. In the both cases, the IC may not start up normally due to unstable pin input/output status at power on. • Do not drive the DSIO pin with a low level from outside, as it generates a debug interrupt that puts the CPU into DEBUG mode.
	4-3	4.3.3 Flash Programming Modified Figure 4.3.3.1. C_{VPP} was changed to that must always be connected. Corrected the following description: C_{VPP} should be connected if the V_{PP} voltage is not stable due to the effect of the distance between the V_{PP} and Flash V_{CC} OUT or other causes. → When supplying the V_{PP} power source, be sure to connect C_{VPP} for stabilizing the V_{PP} voltage.
	6-14	6.7.5 Pd Port Group Modified Table 6.7.5.1. PDIOEN register: PDOEN[4:3], [1:0] → PDOEN[4:0]
	8-4	8.4 Control Registers WDT Control Register Corrected the description of the WDTRUN[3:0] bits. Bits 3–0 WDTRUN[3:0] These bits control WDT to run and stop. 0xa (WP): Stop Values other than 0xa (WP): Run 0xa (R): Idle 0x0 (R): Running
	9-2	9.3.2 Theoretical Regulation Function Corrected Step 1. 1. Measure f_{OSC1} and calculate the frequency tolerance correction value "m [ppm] = -{(f_{OSC1} - 32,768 [Hz]) / 32,768 [Hz]} × 10^6." (Eq. 9.1) m: OSC1 frequency tolerance correction value [ppm]
	9-4	9.4.2 Real-Time Clock Counter Operations Corrective operation when a value out of the effective range is set Added a note. Note: Do not set the RTCMON.RTCMOL[3:0] bits to 0x0 if the RTCMON.RTCMOH bit = 0.
	9-6	9.6 Control Registers RTC Control Register Bits 14–8 RTCTRM[6:0] Added a note. Notes: ... • Writing 0x00 to the RTCCTL.RTCTRM[6:0] bits sets the RTCCTL.RTCTRMBSY bit to 1 as well. However, no correcting operation is performed.

REVISION HISTORY

Code No.	Page	Contents
412645703	9-11	9.6 Control Registers RTC Month/Day Register Bit 12 RTCMOH Bits 11–8 RTCMOL[3:0] Added a note. Notes: ... • Be sure to avoid setting the RTCMON.RTCMOH/RTCMOL[3:0] bits to 0x00.
	10-3	10.4.1 SVD Control Starting detection Corrected Step 4. 4. ... - Set the SVDINTE.SVDIE bit to 1.
	14-1	14.1 Overview Added the following description: • The input filter for the SDA and SCL inputs does not comply with the standard for removing noise spikes less than 50 ns.
	14-7 to 8	14.4.3 Data Reception in Master Mode Data receiving procedure Added Step 1. (The old step numbers were carried down in order.) 1. When receiving one-byte data, write 1 to the I2C*n*CTL.TXNACK bit. Modified Figure 14.4.3.2. A flow for Step 1 was added.
	14-12 to 13	14.4.6 Data Reception in Slave Mode Data receiving procedure Added Step 1. (The old step numbers were carried down in order.) 1. When receiving one-byte data, write 1 to the I2C*n*CTL.TXNACK bit. Modified Figure 14.4.6.2. A flow for Step 1 was added.
	15-5	15.4.2 Counter Block Operations MAX counter data register Added a note. Note: When rewriting the MAX value, the new MAX value should be written after the counter has been reset to the previously set MAX value.
	17-2	17.2.1 List of Output Pins Modified Table 17.2.1.1. SEGxx/COMxx pin I/O: O → A Added a note. Notes: ... • When an LCD panel is connected, set the LCD8CTL.MODEN bit to 1, as activating the LCD panel when it is set to 0 may cause the LCD panel characteristics to fluctuate.
	17-4	17.4.3 External Voltage Application Mode 2 Corrected the description. In this mode, the LCD drive voltage V_{C1} or V_{C2} is applied from outside the IC and other voltages are internally generated.
	17-6	17.5.2 Display On/Off Added a note. Note: When "Display off" is selected, the electric charges of V_{C4} (or V_{C3}) must be discharged in the following procedure. ... To turn the display on again, perform the above procedure in the reverse order.
	20-1	20.1 Absolute Maximum Ratings Modified the characteristics table. V_I: #RESET was added to the condition.
	20-1	20.2 Recommended Operating Conditions Added "(V_{SS} = 0 V) ∗1" and the following annotations: ∗1 The potential variation of the V_{SS} voltage should be suppressed to within ±0.3 V on the basis of the ground potential of the MCU mounting board while the Flash is being programmed, as it affects the Flash memory characteristics (programming count). ∗7 The component values should be determined after evaluating operations using an actual mounting board. Modified the characteristics table. $V_{C1-3/4}$ (1/3 bias): Condition = When an external voltage is applied $V_{C1} \le V_{C2} \le V_{C3}$ (= V_{C4}), $V_{C1} \le V_{DD}$ V_{C1-4} (1/4 bias): Condition = When an external voltage is applied $V_{C1} \le V_{C2} \le V_{C3} \le V_{C4}$, $V_{C1} \le V_{DD}$ C_{VPP}: ∗6 was deleted.

Code No.	Page	Contents
412645703	20-4	20.4 System Reset Controller (SRC) Characteristics Reset hold circuit characteristics Modified the characteristics table. t_{RSTR}: Min. = 0.5 ms, Max. = 0.9 ms
	20-7	20.6 Flash Memory Characteristics Added an annotation. *1 The potential variation of the V_{SS} voltage should be suppressed to within ±0.3 V on the basis of the ground potential of the MCU mounting board while the Flash is being programmed, as it affects the Flash memory characteristics (programming count).
	20-9	20.9 UART (UART) Characteristics Modified the characteristics table. U_{BRT2}: Max. = 115,200 bps (V_{DD} = 1.6 to 3.6 V), 57,600 bps (V_{DD} = 1.2 to 1.6 V)
	21-1	21 Basic External Connection Diagram Modified the figures. C_{VPP} was changed to that must always be connected.
	22-1 to 3	22 Package A JEITA name was added to the package name. Replaced Figure 22.1.
	AP-A-9	Appendix A List of Peripheral Circuit Control Registers PDIOEN (Pd Port Enable Register) Modified the register table. PDIOEN register: PDOEN[4:3], [1:0] → PDOEN[4:0]
	AP-C-3	Appendix C Mounting Precautions Added a description. Treatment of exposed die pad
412645704	Back of cover	Replaced the NOTICE.
	1-3, 1-6, 22-2	The TQFP14-80PIN package was changed to the QFP14-80PIN package. TQFP14-80PIN (P-TQFP080-1212-0.50, 12 × 12 mm, t = 1.2 mm, 0.5 mm pitch) → QFP14-80PIN (P-LQFP080-1212-0.50, 12 × 12 mm, t = 1.7 mm, 0.5 mm pitch)

EPSON

International Sales Operations

America

Epson America, Inc.

Headquarter:
3131 Katella Ave.,
Los Alamitos, CA 90720, USA
Phone: +1-800-463-7766

San Jose Office:
2860 Zanker Road Suite 204,
San Jose, CA 95134, USA
Phone: +1-800-463-7766

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.

4th Floor, Tower 1 of China Central Place, 81 Jianguo Road,
Chaoyang District, Beijing 100025, China
Phone: +86-10-8522-1199 Fax: +86-10-8522-1120

Shanghai Branch

Room 601-603, Building A One East, No. 325 East Longhua
Road, Shanghai 200023, China
Phone: +86-21-5330-4888 Fax: +86-21-5423-4677

Shenzhen Branch

Room 804-805, 8 Floor, Tower 2, Ali Center, No. 3331
Keyuan South RD (Shenzhen bay), Nanshan District,
Shenzhen 518054, China
Phone: +86-755-3299-0588 Fax: +86-755-3299-0560

Epson Taiwan Technology & Trading Ltd.

15F, No. 100, Songren Rd, Sinyi Dist, Taipei City 110, Taiwan
Phone: +886-2-8786-6688

Epson Singapore Pte., Ltd.

438B Alexandra Road,
Block B Alexandra TechnoPark, #04-01/04, Singapore 119968
Phone: +65-6586-5500 Fax: +65-6271-7066

Epson Korea Co., Ltd

10F Posco Tower Yeoksam, Teheranro 134 Gangnam-gu,
Seoul, 06235, Korea
Phone: +82-2-3420-6695

Seiko Epson Corp.
Sales & Marketing Division

MD Sales & Marketing Department

JR Shinjuku Miraina Tower, 4-1-6 Shinjuku, Shinjuku-ku,
Tokyo 160-8801, Japan

Document Code: 412645704
First Issue October 2013
Revised August 2023 in JAPAN Ⓛ