EPSON
EXCEED YOUR VISION

CMOS 32-BIT SINGLE CHIP MICROCONTROLLER

S1C31D41
Technical Manual

arm

SEIKO EPSON CORPORATION

Rev. 1.1

Evaluation board/kit and Development tool important notice

1. This evaluation board/kit or development tool is designed for use with engineering evaluation, demonstration, or development purposes only. Do not use it for other purposes. It is not intended to meet the design requirements of finished products.
2. This 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. Seiko Epson does not assume any responsibility or liability of any kind of damage and/or fire caused by its use. The user should cease to use it when any abnormal issue occurs even during proper and safe use.
3. Parts used for this evaluation board/kit or development tool may be changed without any notice.

NOTICE : PLEASE READ CAREFULLY BELOW 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.) and applications individually listed in this document ("General Purpose"). Epson products are NOT intended for any use beyond the General 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. Therefore, you are advised to use Epson products only for General Purpose uses. Should you desire to buy and use Epson products for a particular purpose other than a General Purpose use, 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.

   【Particular purpose】
   Space equipment (artificial satellites, rockets, etc.)
   Transportation vehicles and their control equipment (automobiles, aircraft, trains, ships, etc.)
   Medical equipment (other than applications individually listed in this document) / 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
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.

(Rev. e1.0, 2021.9)

Preface

This is a technical manual for designers and programmers who develop a product using the S1C31D41. This document describes the functions of the IC, embedded peripheral circuit operations, and their control methods.

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

(reserved): Reserved bit. Do not alter from the initial value.

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.

Low power mode

This manual describes the low power modes as HALT mode and SLEEP mode. These terms refer to sleep mode and deep sleep mode in the Cortex®-M0+ processor, respectively.

– Contents –

1 Overview

1.1 Features

The S1C31D41 is a 32-bit MCU that includes an Arm® Cortex®-M0+ processor and a specific hardware block called the HW Processor. The HW Processor features 2-channel Voice/Audio Playback such as BGM + voice, Voice Speed Conversion, and Self Memory Check without using any CPU resource. With the HW Processor, a low memory footprint and multi-language support are achievable because of its integrated high-compression/high-sound-quality algorithm for voice and audio. In addition to a speaker, though generally difficult, it allows small-sized equipment, which cannot mount a speaker, to playback voice and audio using a buzzer. The S1C31D41 is suitable for home electronics, white goods, and battery-based products, which require a voice and audio playback function.

Table 1.1.1 Features

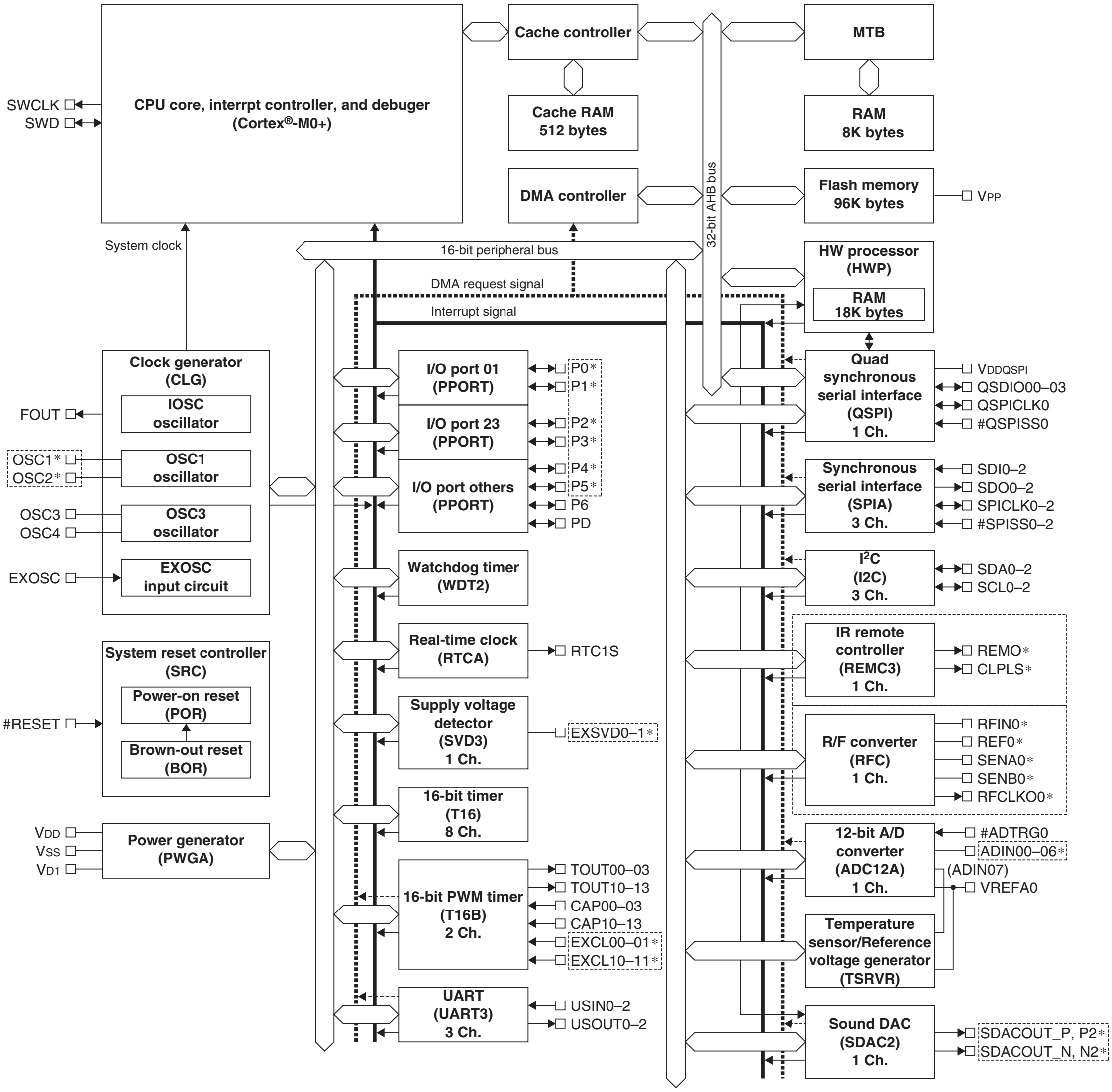
S1C31D41 lineup	32-pin package	48-pin package	64-pin package
CPU			
CPU	Arm® 32-bit RISC processor Cortex®-M0+		
Other	Serial-wire debug ports (SW-DP) and a micro trace buffer (MTB) included		
Embedded Flash memory			
Capacity	96K bytes (for both instructions and data)		
Erase/program count	1,000 times (min.) * When being programmed by the dedicated flash loader		
Other	On-board programming function		
	Flash programming voltage can be generated internally.		
Embedded RAMs			
General-purpose RAM	8K bytes		
Voice RAM	18K bytes (Usable as a general-purpose RAM when the HW processor is inactive.)		
Instruction cache	512 bytes		
HW processor (HWP)			
Sound play function			
Sound algorithm	EPSON high quality and high compression algorithm (EOV: EPSON Original Sound Format)		
Playback channels	2 channels with mixing supported (e.g. Ch.0: voice, Ch.1: BGM)		
Sampling frequency	15.625 kHz		
Bitrate	16/24 kbps		
Playback speed conversion	75% to 125%, 5% steps (when used alone) 85% to 115%, 5% steps (when used in combination with playback pitch conversion) * Available only in Ch.0		
Playback pitch conversion	75% to 125%, 5% steps (when used alone) 90% to 110%, 5% steps (when used in combination with playback pitch conversion) * Available only in Ch.0		
Sound output	Speaker output and buzzer output		
Other	Voice/audio playback using an electromagnetic or piezoelectric buzzer		
	Tone generator function		
Memory check function			
Embedded RAM check	Read/write check, March-C		
Embedded Flash check	Checksum, CRC		
External QSPI-Flash check	Checksum, CRC		
Sound DAC (SDAC2)			
Sampling frequency	15.625 kHz		
Serial interfaces			
UART (UART3)	3 channels		
	Baud-rate generator included, IrDA1.0 supported		
	Open drain output, signal polarity, and baud rate division ratio are configurable.		
	Infrared communication carrier modulation output function		
Synchronous serial interface (SPIA)	3 channels		
	2 to 16-bit variable data length		
	The 16-bit timer (T16) can be used for the baud-rate generator in master mode.		
Quad synchronous serial interface (QSPI)	1 channel		
	Supports single, dual, and quad transfer modes.		
	Low CPU overhead memory mapped access mode that can directly read data from the external flash memory with XIP (eXecute-In-Place) mode.		
I²C (I2C) *1	3 channels		
	Baud-rate generator included		
DMA controller (DMAC)			
Number of channels	4 channels		
Data transfer path	Memory to memory, memory to peripheral, and peripheral to memory		
Transfer mode	Basic, ping-pong, scatter-gather		
DMA trigger source	UART3, SPIA, QSPI, I2C, T16B, ADC12A, and software		

S1C31D41 lineup	32-pin package	48-pin package	64-pin package
Clock generator (CLG)			
System clock source	4 sources (IOSC/OSC1/OSC3/EXOSC)		
System clock frequency (operating frequency)	V_{D1} voltage mode = mode0: 16 MHz (max.) V_{D1} voltage mode = mode1: 1.8 MHz (max.)		
IOSC oscillator circuit (boot clock source)	V_{D1} voltage mode = mode0: 8/2/1 MHz (typ.) software selectable V_{D1} voltage mode = mode1: 1.8/0.9 MHz (typ.) software selectable		
	10 µs (typ.) starting time (time from cancelation of SLEEP state to vector table read by the CPU)		
OSC1 oscillator circuit	32.768 kHz (typ.) crystal oscillator		
	32kHz (typ.) embedded oscillator		
	Oscillation stop detection circuit included		
OSC3 oscillator circuit	0.2 to 16.3 MHz crystal/ceramic oscillator		
	16/8/4 MHz (typ.) embedded oscillator		
EXOSC clock input	0.016 to 16.3 MHz 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	25 bits (max.)	39 bits (max.)	55 bits (max.)
	Pins are shared with the peripheral I/O.		
Number of input interrupt ports	21 bits (max.)	35 bits (max.)	51 bits (max.)
Number of ports that support universal port multiplexer (UPMUX)	9 bits	20 bits	32 bits
	A peripheral circuit I/O function selected via software can be assigned to each port.		
Timers			
Watchdog timer (WDT2)	Generates NMI or watchdog timer reset.		
	Programmable NMI/reset generation cycle		
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)	8 channels		
	Generates the SPIA and QSPI master clocks, and the ADC12A operating clock/trigger signal.		
16-bit PWM timer (T16B)	2 channels		
	Event counter/capture function		
	PWM waveform generation function		
	Number of PWM output or capture input ports: 4 ports/channel		
Supply voltage detector (SVD3)			
Number of channels	1 channel		
Detection voltage	V_{DD} or an external voltage (2 external detection ports are available.)		
Detection level	V_{DD}: 28 levels (1.8 to 5.0 V)/external voltage: 32 levels (1.2 to 5.0 V)		
Other	Intermittent operation mode		
	Generates an interrupt or reset according to the detection level evaluation.		
12-bit A/D converter (ADC12A)			
Conversion method	Successive approximation type		
Resolution	12 bits		
Number of conversion channels	1 channel		
Number of analog signal inputs	6 ports/channel (max.)		8 ports/channel (max.)
	1 port is dedicated for the built-in temperature sensor.		
Temperature sensor/reference voltage generator (TSRVR)			
Temperature sensor circuit	Sensor output can be measured using ADC12A.		
Reference voltage generator	Reference voltage for ADC12A is selectable from 2.0 V, 2.5 V, V_{DD}, and external input.		
R/F converter (RFC)			
Conversion method	–	CR oscillation type with 24-bit counters	
Number of conversion channels	–	1 channel (Up to two sensors can be connected.)	
Supported sensors	–	DC-bias resistive sensors	
IR remote controller (REMC3)			
Number of transmitter channels	–	1 channel	
Other	–	EL lamp drive waveform can be generated (by the hardware) for an application example.	
	–	Output inversion function	
Reset			
#RESET pin	Reset when the reset pin is set to low.		
Power-on reset	Reset at power on.		
Brown-out reset	Reset when the power supply voltage drops (when $V_{DD} \leq 1.45$ V (typ.) is detected).		
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).		

S1C31D41 lineup	32-pin package	48-pin package	64-pin package
Interrupt			
Non-maskable interrupt	6 systems (Reset, NMI, HardFault, SVCall, PendSV, SysTic)		
Programmable interrupt	External interrupt: 3 systems		
	Internal interrupt: 27 systems		
Power supply voltage			
V_{DD} operating voltage	1.8 to 5.5 V ∗ If V_{DD} > 3.6 V, the V_{D1} voltage mode must be set to mode0.		
V_{DD} operating voltage for Flash programming	2.2 to 5.5 V		
QSPI-Flash interface power voltage	3.0 to 3.6 V (voltage different from V_{DD} can be supplied.)		
Operating temperature			
Operating temperature range	-40 to 85 °C		
Current consumption (Typ. value)			
SLEEP mode ∗2	0.34 µA IOSC = OFF, OSC1 = OFF, OSC3 = OFF		
	0.9 µA IOSC = OFF, OSC1 = 32.768 kHz (crystal oscillator), OSC3 = OFF, RTCA = ON		
HALT mode ∗3	1.5 µA IOSC = OFF, OSC1 = 32.768 kHz (crystal oscillator), OSC3 = OFF		
RUN mode	215 µA/MHz V_{D1} voltage mode = mode0, CPU = IOSC		
	130 µA/MHz V_{D1} voltage mode = mode1, CPU = IOSC		
Shipping form			
Package ∗4	TQFP12-32PIN (P-TQFP0328-0707-0.80, 7 × 7 mm, t = 1.2 mm, 0.8 mm pitch)	TQFP12-48PIN (P-TQFP048-0707-0.50, 7 × 7 mm, t = 1.2 mm, 0.5 mm pitch)	QFP13-64PIN (P-LQFP064-1010-0.50, 10 × 10 mm, t = 1.7 mm, 0.5 mm pitch)

∗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 SLEEP mode refers to deep sleep mode in the Cortex®-M0+ processor. The RAM retains data even in SLEEP mode.
∗3 HALT mode refers to sleep mode in the Cortex®-M0+ processor.
∗4 Shown in parentheses are JEITA package names.

1.2 Block Diagram



* The pin configuration depends on the package. For detailed information, refer to Section 1.3, "Pins."

Figure 1.2.1 S1C31D41 Block Diagram

1.3 Pins

1.3.1 Pin Configuration Diagram

TQFP12-32PIN

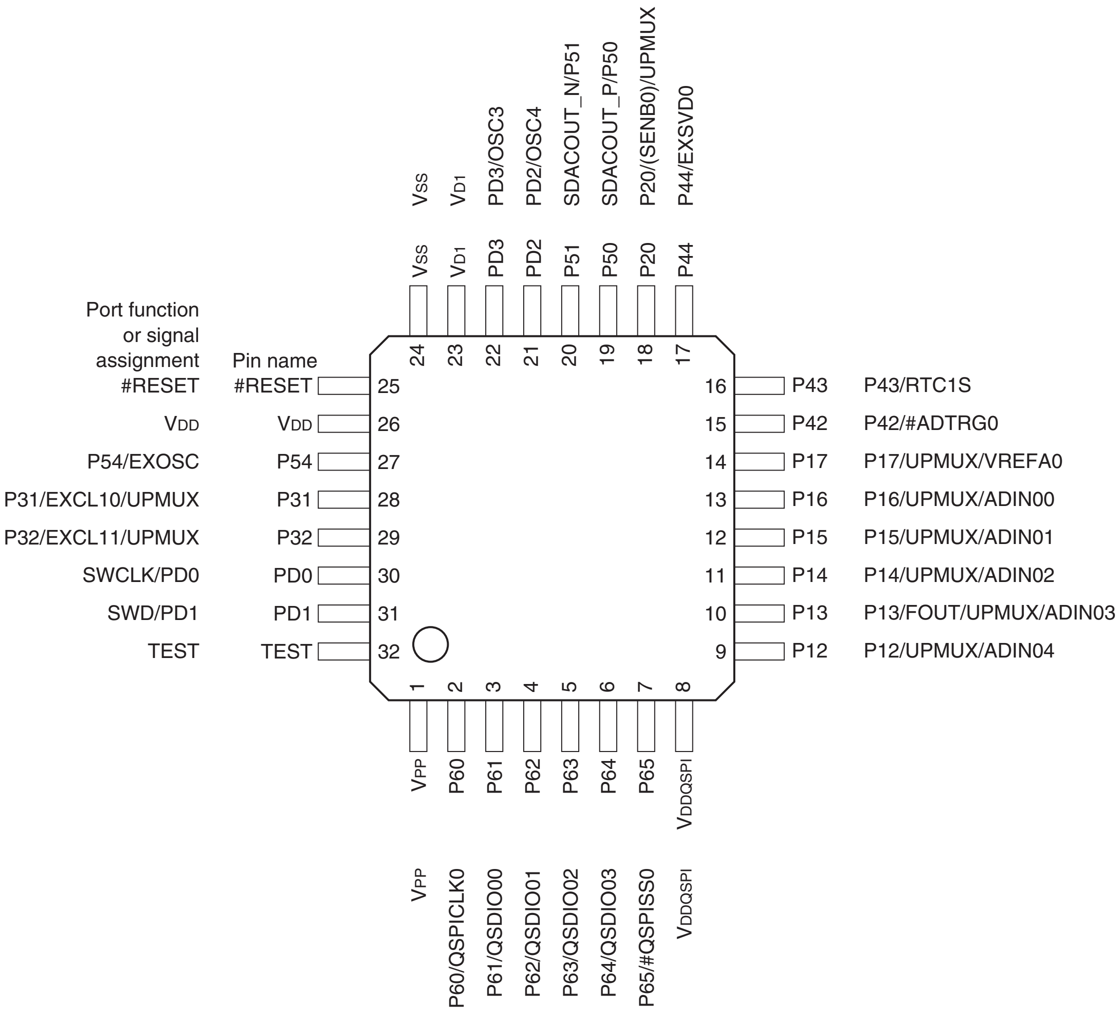


Figure 1.3.1.1 S1C31D41 Pin Configuration Diagram (TQFP12-32PIN)

TQFP12-48PIN

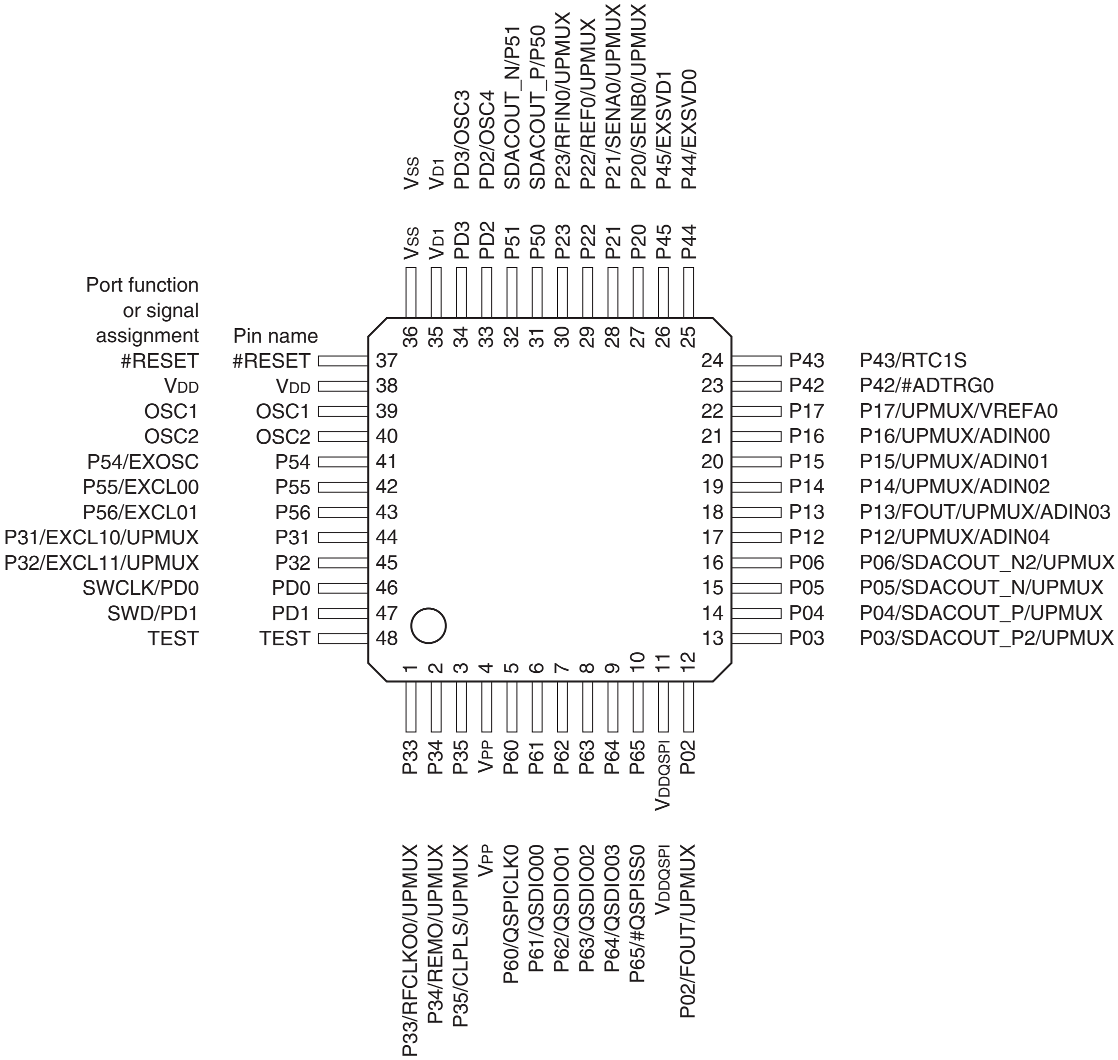


Figure 1.3.1.2 S1C31D41 Pin Configuration Diagram (TQFP12-48PIN)

QFP13-64PIN

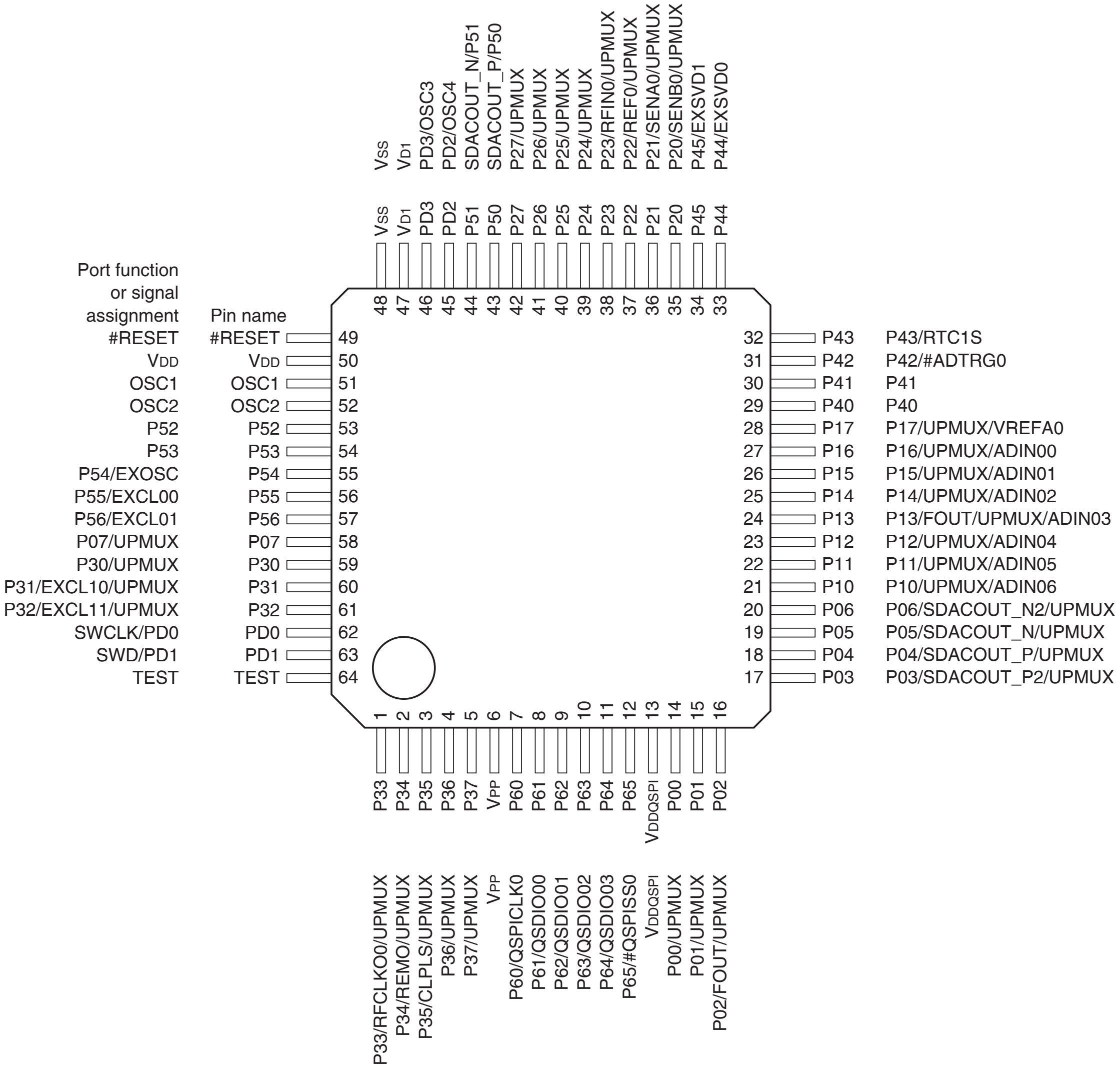


Figure 1.3.1.3 S1C31D41 Pin Configuration Diagram (QFP13-64PIN)

1.3.2 Pin Descriptions

Symbol meanings

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

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

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

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

Table 1.3.2.1 Pin Description

Pin name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function	Package 32-pin	Package 48-pin	Package 64-pin
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	–	–	V_{D1} regulator output	✓	✓	✓
V_{DDQSPI}	V_{DDQSPI}	P	–	–	QSPI interface/P6 port group power supply	✓	✓	✓
OSC1	OSC1	A	–	–	OSC1 oscillator circuit input	–	✓	✓
OSC2	OSC2	A	–	–	OSC1 oscillator circuit output	–	✓	✓
TEST	TEST	I	I (Pull-down)	–	Test mode enable input	✓	✓	✓
#RESET	#RESET	I	I (Pull-up)	–	Reset input	✓	✓	✓
P00	P00	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P01	P01	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P02	P02	I/O	Hi-Z	✓	I/O port	–	✓	✓
	FOUT	O			Clock external output			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P03	P03	I/O	Hi-Z	✓	I/O port	–	✓	✓
	SDACOUT_P2	O			Buzzer sound DAC positive output 2			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P04	P04	I/O	Hi-Z	✓	I/O port	–	✓	✓
	SDACOUT_P	O			Buzzer sound DAC positive output 1			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P05	P05	I/O	Hi-Z	✓	I/O port	–	✓	✓
	SDACOUT_N	O			Buzzer sound DAC nagetive output 1			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P06	P06	I/O	Hi-Z	✓	I/O port	–	✓	✓
	SDACOUT_N2	O			Buzzer sound DAC nagetive output 2			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P07	P07	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P10	P10	I/O	Hi-Z	–	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN06	A			12-bit A/D converter Ch.0 analog signal input 6			
P11	P11	I/O	Hi-Z	–	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN05	A			12-bit A/D converter Ch.0 analog signal input 5			
P12	P12	I/O	Hi-Z	–	I/O port	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN04	A			12-bit A/D converter Ch.0 analog signal input 4			

Pin name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function	Package 32-pin	Package 48-pin	Package 64-pin
P13	P13	I/O	Hi-Z	–	I/O port	✓	✓	✓
	FOUT	O			Clock external output			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN03	A			12-bit A/D converter Ch.0 analog signal input 3			
P14	P14	I/O	Hi-Z	–	I/O port	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN02	A			12-bit A/D converter Ch.0 analog signal input 2			
P15	P15	I/O	Hi-Z	–	I/O port	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN01	A			12-bit A/D converter Ch.0 analog signal input 1			
P16	P16	I/O	Hi-Z	–	I/O port	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	ADIN00	A			12-bit A/D converter Ch.0 analog signal input 0			
P17	P16	I/O	Hi-Z	–	I/O port	✓	✓	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
	VREFA0	A			12-bit A/D converter Ch.0 reference voltage input			
P20	P20	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)			
P21	P21	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)			
P22	P22	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)			
P23	P23	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)			
P24	P24	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P25	P25	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P26	P26	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P27	P27	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P30	P30	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P31	P31	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)			
P32	P32	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)			
P33	P33	I/O	Hi-Z	✓	I/O port	–	✓	✓
	RFCLKO0	O			R/F converter Ch.0 clock monitor output			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P34	P34	I/O	Hi-Z	✓	I/O port	–	✓	✓
	REMO	O			IR remote controller transmit data output			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P35	P35	I/O	Hi-Z	✓	I/O port	–	✓	✓
	CLPLS	O			IR remote controller clear pulse output			
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P36	P36	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P37	P37	I/O	Hi-Z	✓	I/O port	–	–	✓
	UPMUX	I/O			User-selected I/O (universal port multiplexer)			
P40	P40	I/O	Hi-Z	–	I/O port	–	–	✓
P41	P41	I/O	Hi-Z	✓	I/O port	–	–	✓
P42	P42	I/O	Hi-Z	–	I/O port	✓	✓	✓
	#ADTRG0	I			12-bit A/D converter Ch.0 trigger input			
P43	P43	I/O	Hi-Z	–	I/O port	✓	✓	✓
	RTC1S	O			Real-time clock 1-second cycle pulse output			
P44	P44	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	EXSVD0	A			Supply voltage detector external voltage detection input 0			

Pin name	Assigned signal	I/O	Initial state	Tolerant fail-safe structure	Function	Package 32-pin	48-pin	64-pin
P45	P45	I/O	Hi-Z	✓	I/O port	–	✓	✓
	EXSVD1	A			Supply voltage detector external voltage detection input 1			
P50	SDACOUT_P	O	O (L)	✓	Sound DAC positive output	✓	✓	✓
	P50	I/O			I/O port			
P51	SDACOUT_N	O	O (L)	✓	Sound DAC nagetive output	✓	✓	✓
	P51	I/O			I/O port			
P52	P52	I/O	Hi-Z	✓	I/O port	–	–	✓
P53	P53	I/O	Hi-Z	✓	I/O port	–	–	✓
P54	P54	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	EXOSC	I			Clock generator external clock input			
P55	P55	I/O	Hi-Z	✓	I/O port	–	✓	✓
	EXCL00	I			16-bit PWM timer Ch.0 event counter input 0			
P56	P56	I/O	Hi-Z	✓	I/O port	–	✓	✓
	EXCL01	I			16-bit PWM timer Ch.0 event counter input 1			
P60	P60	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	QSPICLK0	I/O			Quad synchronous serial interface Ch.0 clock input/output			
P61	P61	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	QSDIO00	I/O			Quad synchronous serial interface Ch.0 data input/output			
P62	P62	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	QSDIO01	I/O			Quad synchronous serial interface Ch.0 data input/output			
P63	P63	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	QSDIO02	I/O			Quad synchronous serial interface Ch.0 data input/output			
P64	P64	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	QSDIO03	I/O			Quad synchronous serial interface Ch.0 data input/output			
P65	P65	I/O	Hi-Z	✓	I/O port	✓	✓	✓
	#QSPISS0	I/O			Quad synchronous serial interface Ch.0 slave-select input/output			
PD0	SWCLK	I	I (Pull-up)	✓	Serial-wire debugger clock input	✓	✓	✓
	PD0	I/O			I/O port			
PD1	SWD	I/O	I (Pull-up)	✓	Serial-wire debugger data input/output	✓	✓	✓
	PD1	I/O			I/O port			
PD2	PD2	I/O	Hi-Z	–	I/O port	✓	✓	✓
	OSC4	A			OSC3 oscillator circuit output			
PD3	PD3	I/O	Hi-Z	–	I/O port	✓	✓	✓
	OSC3	A			OSC3 oscillator circuit 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 1.3.2.2 Peripheral Circuit Input/output Function Selectable by UPMUX

Peripheral circuit	Signal to be assigned	I/O	Channel number *n*	Function
I^2C	SCL*n*	I/O	*n* = 0–2	I2C Ch.*n* clock input/output
(I2C)	SDA*n*	I/O		I2C Ch.*n* data input/output
UART	USIN*n*	I	*n* = 0–2	UART3 Ch.*n* data input
(UART3)	USOUT*n*	O		UART3 Ch.*n* data output
Synchronous serial	SDI*n*	I	*n* = 0–2	SPIA Ch.*n* data input
interface	SDO*n*	O		SPIA Ch.*n* data output
(SPIA)	SPICLK*n*	I/O		SPIA Ch.*n* clock input/output
	#SPISS*n*	I		SPIA Ch.*n* slave-select input
16-bit PWM timer	TOUT*n*0/CAP*n*0	I/O	*n* = 0, 1	T16B Ch.*n* PWM output/capture input 0
(T16B)	TOUT*n*1/CAP*n*1	I/O		T16B Ch.*n* PWM output/capture input 1
	TOUT*n*2/CAP*n*2	I/O		T16B Ch.*n* PWM output/capture input 2
	TOUT*n*3/CAP*n*3	I/O		T16B Ch.*n* PWM output/capture input 3

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 (PWGA)

2.1.1 Overview

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

- Embedded V_{D1} regulator
  - The V_{D1} regulator generates the V_{D1} voltage to drive internal circuits, this makes it possible to keep current consumption constant independent of the V_{DD} voltage level.
  - The V_{D1} regulator supports two operation modes, normal mode and economy mode, and setting the V_{D1} regulator into economy mode at light loads helps achieve low-power operations.
  - The V_{D1} regulator supports two voltage modes, mode0 and mode1, and setting the V_{D1} regulator into mode1 during low-speed operation helps achieve low-power operations.

Figure 2.1.1.1 shows the PWGA configuration.

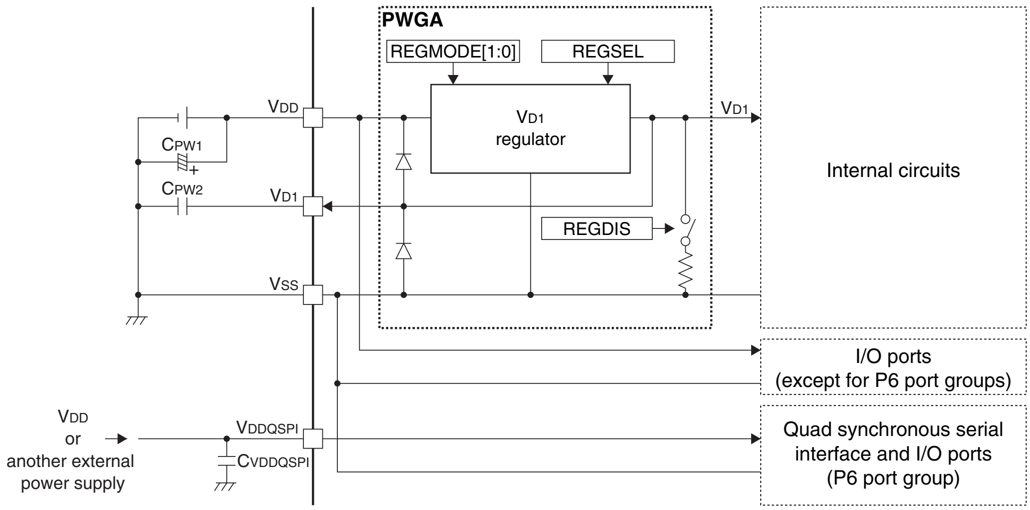


Figure 2.1.1.1 PWGA Configuration

2.1.2 Pins

Table 2.1.2.1 lists the PWGA pins.

Table 2.1.2.1 List of PWGA Pins

Pin name	I/O	Initial status	Function
V_{DD}	P	–	Power supply (+)
V_{SS}	P	–	GND
V_{D1}	A	–	V_{D1} regulator output pin
V_{DDQSPI}	P	–	QSPI interface (QSPI-Flash) and I/O power supply (for P6 port group)

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

V_{DDQSPI}

V_{DDQSPI} is the power supply dedicated for the quad synchronous serial interface (QSPI-Flash). It is also used as the power supply for the I/O ports P60 to P65.

2.1.3 V_{D1} Regulator Operation Mode

The V_{D1} regulator supports two operation modes, normal mode and economy mode. Setting the V_{D1} regulator into economy mode at light loads helps achieve low-power operations. Table 2.1.3.1 lists examples of light load conditions in which economy mode can be set.

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

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

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

2.1.4 V_{D1} Regulator Voltage Mode

The V_{D1} regulator supports two voltage modes, mode0 and mode1.
When the IC runs with a low-speed clock, setting the V_{D1} regulator into mode1 reduces power consumption.

When the voltage mode is switched, the system clock source automatically stops operating and it resumes operating after the voltage has stabilized. Table 2.1.4.1 shows the stop period of the system clock.

Table 2.1.4.1 System Clock Stop Period After Switching Voltage Mode

System clock	Stop period
IOSC	4,096 cycles
OSC1	Number of cycles set using the CLGOSC1.OSC1WT[1:0] bits

Procedure to switch from mode0 to mode1

1. Set the MODEN bits of the peripheral circuits to 0. (Stop using peripheral circuits)
2. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
3. Switch the system clock to a low-speed clock (OSC1, IOSC 1.8 MHz or 0.9 MHz).
4. Stop OSC3 and EXOSC.
5. Configure the following PWGACTL register bits.
   - Set the PWGACTL.REGSEL bit to 0. (Switch to mode1)
   - Set the PWGACTL.REGDIS bit to 1. (Discharge)
   - Set the PWGACTL.REGMODE[1:0] bits to 0x2. (Set to normal mode)
6. Configure the following PWGACTL register bits after the system clock supply has resumed.
   - Set the PWGACTL.REGDIS bit to 0. (Stop discharging)
   - Set the PWGACTL.REGMODE[1:0] bits to 0x0. (Set to automatic mode)
7. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

Procedure to switch from mode1 to mode0

1. Set the MODEN bits of the peripheral circuits to 0. (Stop using peripheral circuits)
2. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
3. Configure the following PWGACTL register bits.
   - Set the PWGACTL.REGSEL bit to 1. (Switch to mode0)
   - Set the PWGACTL.REGMODE[1:0] bits to 0x2. (Set to normal mode)
4. Set the PWGACTL.REGMODE[1:0] bits to 0x0 after the system clock supply has resumed. (Set to automatic mode)
5. Switch the system clock to a high-speed clock.
6. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

Notes: • After the voltage mode has been switched, correct the RTC, as the RTC operating clock is also stopped for the period set using the CLGOSC1.OSC1WT[1:0] bits.

• Always use the IC in mode0 when V_{DD} is 3.6 V or higher.

• When two voltage modes are used, set the V_{D1} regulator into mode1 before putting the IC into SLEEP or HALT mode.

2.2 System Reset Controller (SRC)

2.2.1 Overview

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

- Embedded reset hold circuit maintains reset state to boot the system safely while the internal power supply is unstable after power on or the oscillation frequency is unstable after the clock source is initiated.
- Supports reset requests from multiple reset sources.
  - #RESET pin
  - POR and BOR
  - Reset request from the CPU
  - Watchdog timer reset
  - Supply voltage detector reset
  - Peripheral circuit software reset (supports some peripheral circuits only)
- The CPU registers and peripheral circuit control bits will be reset with an appropriate initialization condition according to changes in status.

Figure 2.2.1.1 shows the SRC configuration.

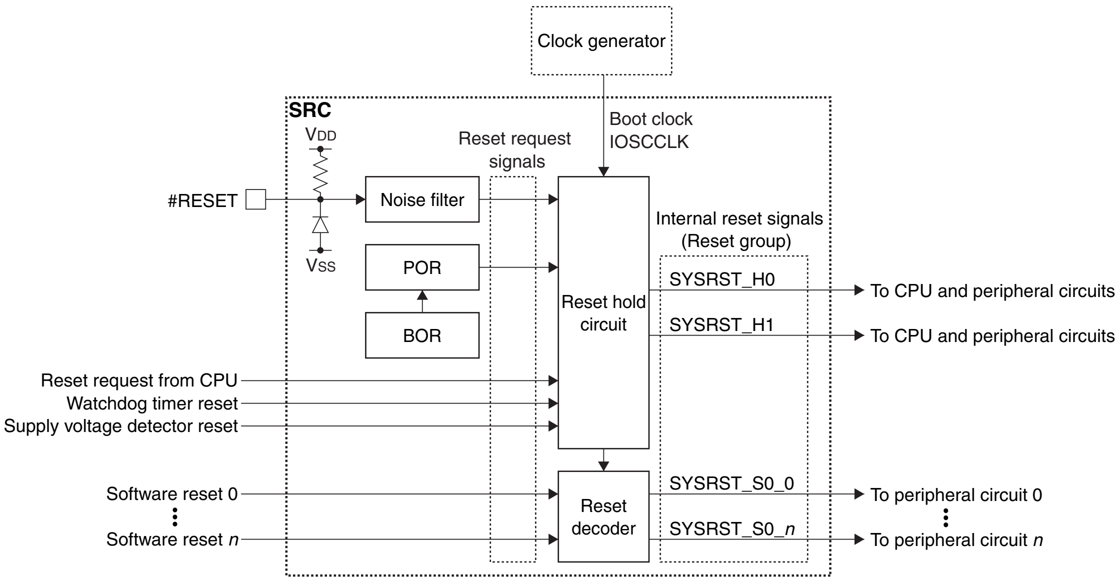


Figure 2.2.1.1 SRC Configuration

2.2.2 Input Pin

Table 2.2.2.1 shows the SRC pin.

Table 2.2.2.1 SRC Pin

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

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

2.2.3 Reset Sources

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

#RESET pin

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

POR and BOR

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

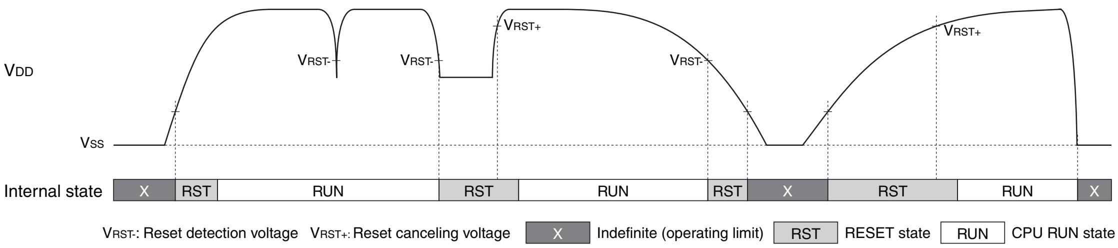


Figure 2.2.3.1 Example of Internal Reset by POR and BOR

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

Reset request from the CPU

The CPU issues a reset request by writing 1 to the AIRCR.SYSRESETREQ bit in the Cortex®-M0+ Application Interrupt and Reset Control Register. For more information, refer to the "ARM®v6-M Architecture Reference Manual."

Watchdog timer reset

Setting the watchdog timer into reset mode will issue a reset request when the counter overflows. This helps return the runaway CPU to a normal operating state. For more information, refer to the "Watchdog timer" chapter.

Supply voltage detector reset

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

Peripheral circuit software reset

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

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

2.2.4 Initialization Conditions (Reset Groups)

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

Table 2.2.4.1 List of Reset Groups

Reset group	Reset source	Reset cancelation timing
H0	#RESET pin POR and BOR Reset request from the CPU 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 and BOR Reset request from the CPU	
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
  - Low-power OSC1 oscillator circuit in which the oscillator type can be specified from high-precision 32.768 kHz crystal oscillator (an external resonator is required) and internal oscillator
  - 16 MHz (max.) high-speed OSC3 oscillator circuit in which the oscillator type can be specified from crystal/ceramic oscillator (an external resonator is required) and internal oscillator
  - EXOSC clock input circuit that allows input of square wave and sine wave clock signals up to 16 MHz
- 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.
- 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.

Table 2.3.1.1 CLG Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
IOSC oscillator circuit	Available	Available	Available
OSC1 crystal oscillator circuit	Unavailable	Available	Available
OSC1 internal oscillator circuit	Available	Available	Available
OSC2 crystal/ceramic oscillator circuit	Available	Available	Available
OSC3 internal oscillator circuit	Available	Available	Available
EXOSC clock input	Available	Available	Available

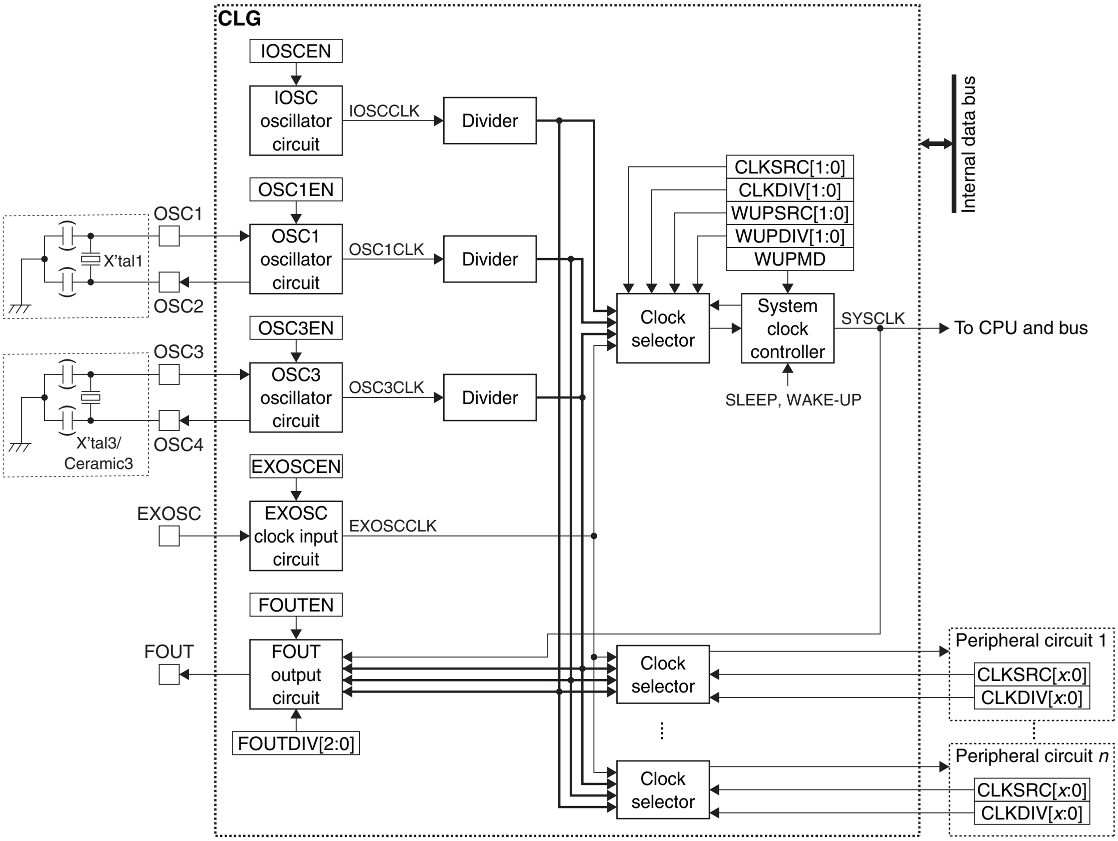


Figure 2.3.1.1 CLG Configuration

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.

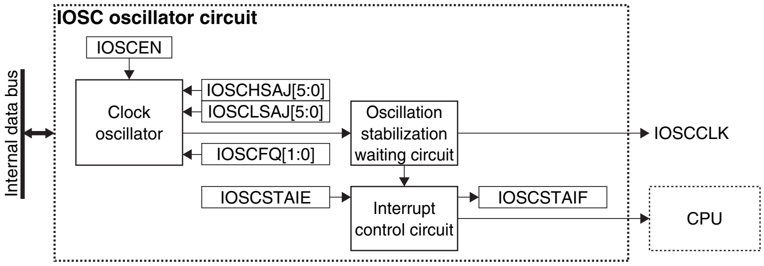


Figure 2.3.3.1 IOSC Oscillator Circuit Configuration

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

OSC1 oscillator circuit

The OSC1 oscillator circuit is a low-power oscillator circuit that allows software to select the oscillator type from two different types shown below. Figure 2.3.3.2 shows the configuration of the OSC1 oscillator circuit.

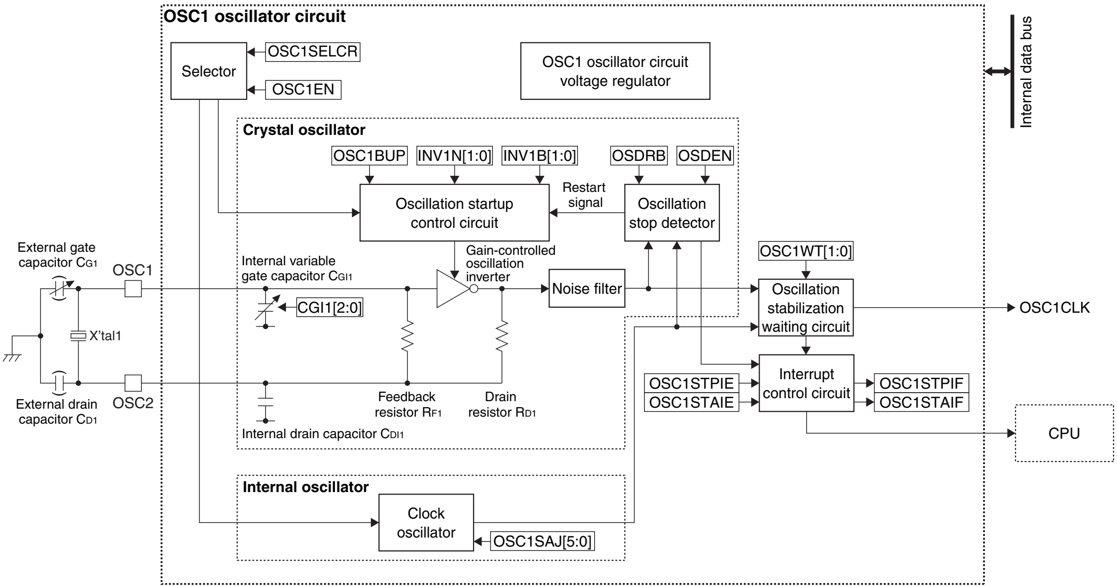


Figure 2.3.3.2 OSC1 Oscillator Circuit Configuration

Crystal oscillator

This oscillator circuit includes a gain-controlled oscillation inverter and a variable gate capacitor allowing use of various crystal resonators (32.768 kHz typ.) 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.

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.

Internal oscillator

This 32 kHz oscillator circuit operates without any external parts.
When the internal oscillator circuit is used, set the OSC1 pin level to Vss and leave the OSC3 pin open.

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.

OSC3 oscillator circuit

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

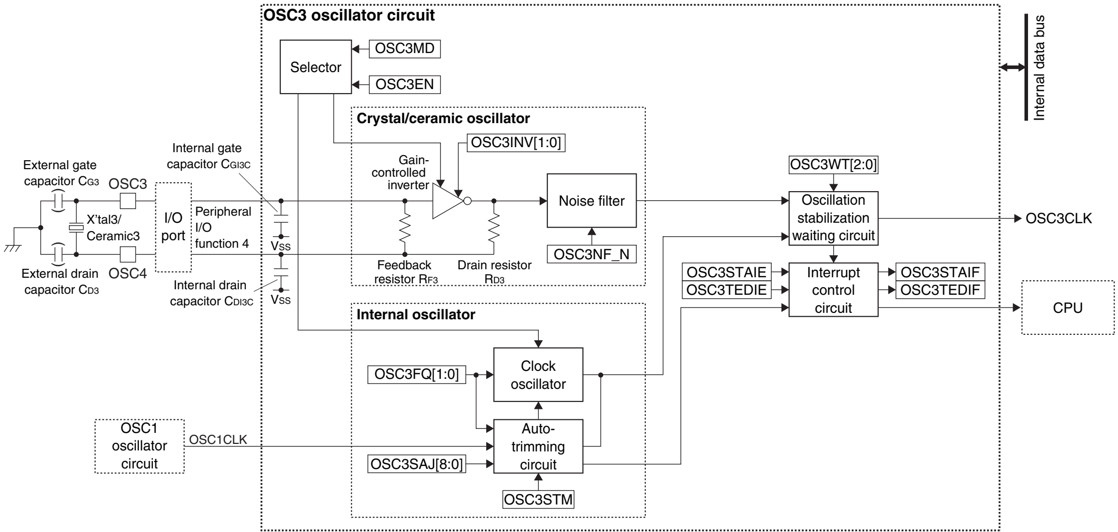


Figure 2.3.3.3 OSC3 Oscillator Circuit Configuration

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.

Internal oscillator

This oscillator circuit features a fast startup and no external parts are required for oscillating. The OSC3CLK frequency can be selected using the CLGOSC3.OSC3FQ[1:0] bits. This 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, refer to "OSC3 oscillation auto-trimming function" in this chapter.

For the recommended parts and the oscillation characteristics, refer to the "Basic External Connection Diagram" chapter and "OSC3 oscillator circuit characteristics" in 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.

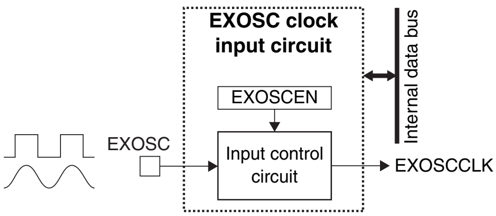


Figure 2.3.3.4 EXOSC Clock Input Circuit

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.

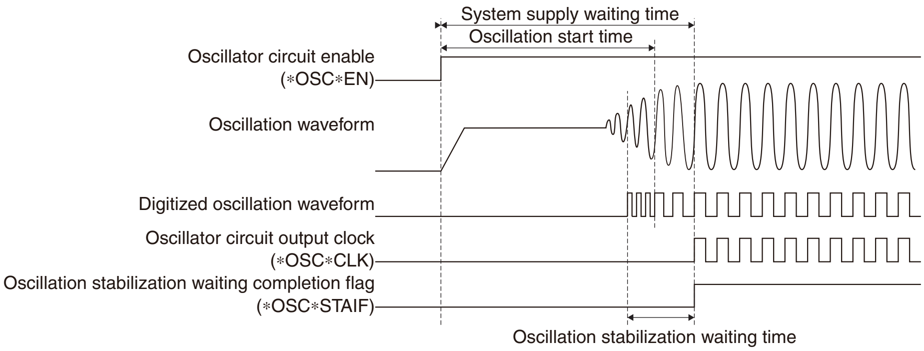


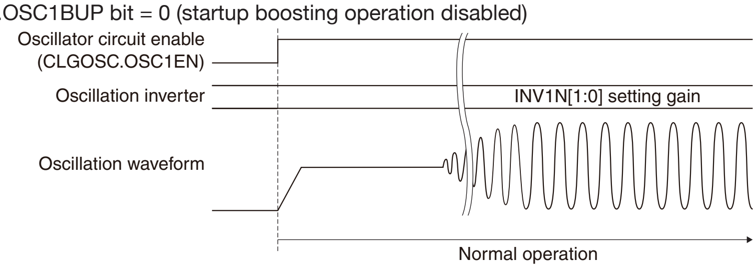
Figure 2.3.4.1 Oscillation Start Time and Oscillation Stabilization Waiting Time

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 when crystal oscillator is selected, or 4,096 OSC1CLK clocks or more when internal oscillator is selected. The oscillation stabilization waiting time for the OSC3 oscillator circuit should be set to 4,096 OSC3CLK clocks or more.
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.

When the oscillation startup control circuit in the OSC1 oscillator circuit is enabled by setting the CLGOSC1.OSC1BUP bit to 1, it uses 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) to reduce oscillation start time. 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.

(1) CLGOSC1.OSC1BUP bit = 0 (startup boosting operation disabled)



(2) CLGOSC1.OSC1BUP bit = 1 (startup boosting operation enabled)

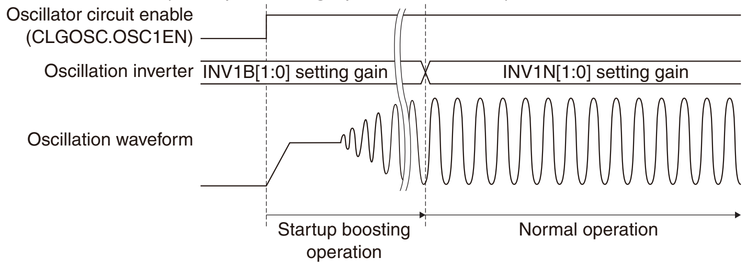


Figure 2.3.4.2 Operation Example when the Oscillation Startup Control Circuit is Used

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 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the CLGIOSC.IOSCFQ[1:0] bits. (Select frequency)
5. Set the CLGTRIM1.IOSCLSAJ[5:0] bits (f_{IOSC} = 2/1 MHz) or CLGTRIM1.IOSCHSAJ[5:0] bits (f_{IOSC} = 8 MHz) as necessary. (Finely adjust oscillation frequency)
6. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)
7. Write 1 to the CLGOSC.IOSCEN bit. (Start oscillation)
8. IOSCCLK can be used if the CLGINTF.IOSCSTAIF bit = 1 after an interrupt occurs.

The setting values of the CLGTRIM1.IOSCLSAJ[5:0] and CLGTRIM1.IOSCHSAJ[5:0] bits should be determined after performing evaluation using the populated circuit board.

Note: Make sure the CLGOSC.IOSCEN bit is set to 0 (while the IOSC oscillation is halted) when setting the CLGTRIM1.IOSCLSAJ[5:0] or CLGTRIM1.IOSCHSAJ[5:0] bits.

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 SYSPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGOSC1 register bits:
   - CLGOSC1.OSC1SELCR bit (Select oscillator type)
   - CLGOSC1.OSC1WT[1:0] bits (Set oscillation stabilization waiting time)

   In addition to the above, configure the following bits when using the crystal oscillator:
   - CLGOSC1.INV1N[1:0] bits (Set oscillation inverter gain)
   - CLGOSC1.CGI1[2:0] bits (Set internal gate capacitor)
   - CLGOSC1.INV1B[1:0] bits (Set oscillation inverter gain for startup boosting period)
   - CLGOSC1.OSC1BUP bit (Enable/disable oscillation startup control circuit)
5. When using the internal oscillator, set the CLGTRIM2.OSC1SAJ[5:0] bits as necessary. (Finely adjust oscillation frequency)
6. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)
7. Write 1 to the CLGOSC.OSC1EN bit. (Start oscillation)
8. 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], CLGOSC1.INV1B[1:0], and CLGTRIM2.OSC1SAJ[5:0] bits should be determined after performing evaluation using the populated circuit board.

Note: Make sure the CLGOSC.OSC1EN bit is set to 0 (while the OSC1 oscillation is halted) when setting the CLGTRIM2.OSC1SAJ[5:0] bits.

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 SYSPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGOSC3 register bits:
   - CLGOSC3.OSC3MD bit (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. When using the internal oscillator, set the CLGTRIM3.OSC3SAJ[8:0] bits as necessary. (Finely adjust oscillation frequency)
6. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)
7. When using the crystal/ceramic oscillator, assign the OSC3 oscillator input/output functions to the ports. (Refer to the “I/O Ports” chapter.)
8. Write 1 to the CLGOSC.OSC3EN bit. (Start oscillation)
9. OSC3CLK can be used if the CLGINTF.OSC3STAIF bit = 1 after an interrupt occurs.

The setting values of the CLGOSC3.OSC3INV[1:0], CLGOSC3.OSC3WT[2:0], and CLGTRIM3.OSC3SAJ[8:0] 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 setting the CLGTRIM3.OSC3SAJ[8:0] bits.

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 SYSPROT.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

Whether the clock sources being operated are stopped or not when the CPU enters SLEEP mode (deep sleep mode) 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.

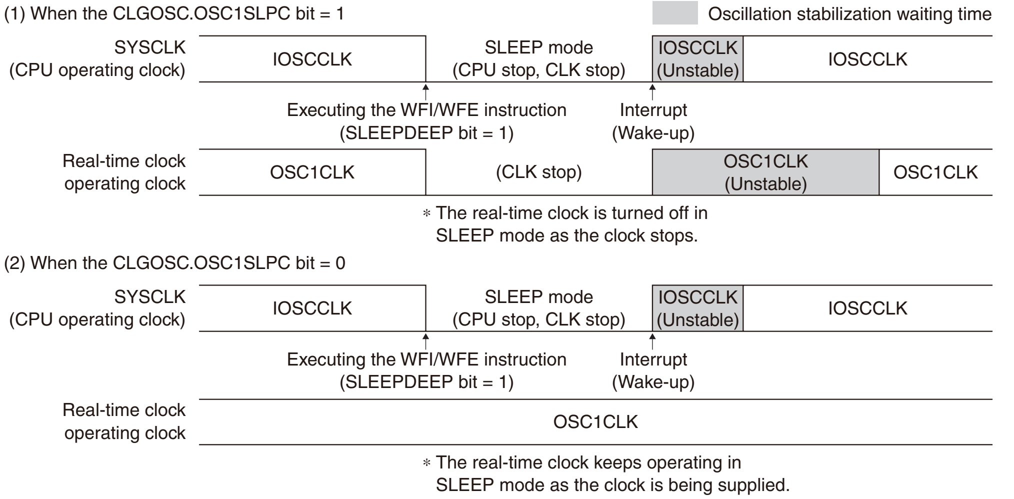


Figure 2.3.4.3 Clock Control Example in SLEEP Mode

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.

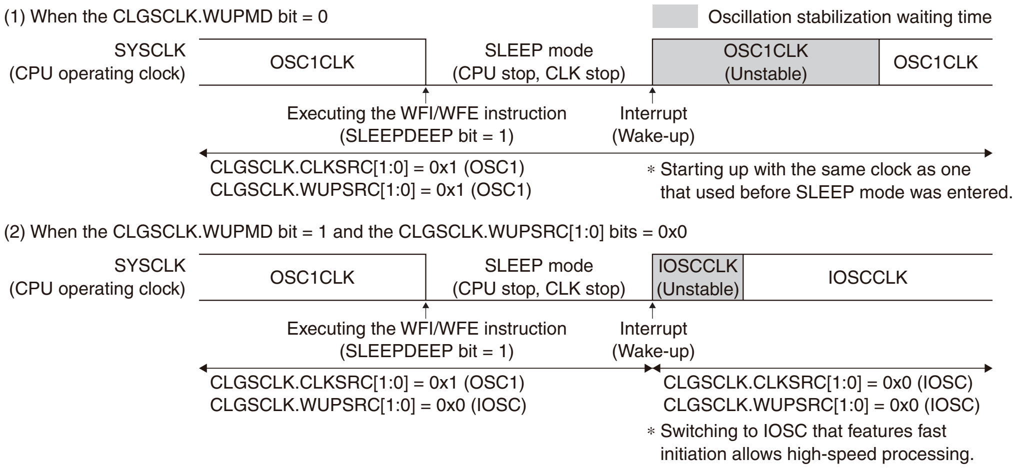


Figure 2.3.4.4 Clock Control Example at SLEEP Cancelation

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)

OSC3 oscillation auto-trimming function

The auto-trimming function adjusts the OSC3CLK clock frequency by trimming the clock with reference to the high precision OSC1CLK clock generated by the OSC1 oscillator circuit (crystal oscillator). However, this function is effective only when 16 MHz (CLGOSC3.OSC3FQ[1:0] bits = 0x3) has been selected to the OSC3 oscillation frequency.

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 OSC3 oscillation, check if the stabilized clock is supplied (CLGINTF.OSC3STAIF bit = 1).
3. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
4. Configure the following CLGINTF register bits:
   - Write 1 to the CLGINTF.OSC3TEDIF bit. (Clear interrupt flag)
   - Write 1 to the CLGINTF.OSC3TERIF bit. (Clear interrupt flag)
5. Configure the following CLGINTF register bits:
   - Set the CLGINTE.OSC3TEDIE bit to 1. (Enable interrupt)
   - Set the CLGINTE.OSC3TERIE bit to 1. (Enable interrupt)
6. Write 1 to the CLGOSC3.OSC3STM bit. (Enable OSC3 oscillation auto-trimming)
7. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)
8. The trimmed OSC3CLK can be used if the CLGINTF.OSC3TEDIF bit = 1 after an interrupt occurs. If the CLGINTF.OSC3TERIF bit = 1, an error has occurred during the auto-trimming operation (the clock has not been adjusted).

After the trimming operation has completed, the CLGOSC3.OSC3STM bit automatically reverts to 0. Although the trimming time depends on the temperature, an average of several 10 ms is required.

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 SYSPROT.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 SYSPROT.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.

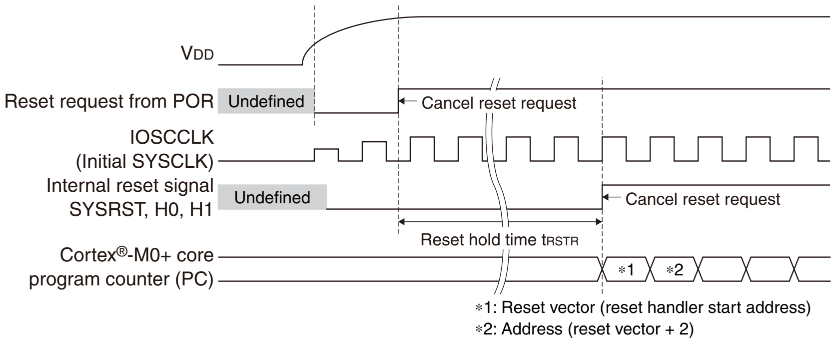


Figure 2.4.1.1 Initial Boot Sequence

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 Cortex®-M0+ core executes the WFI or WFE instruction with the SLEEPDEEP bit of the Cortex®-M0+ System Control Register set to 0, it suspends program execution and stops operating. This state is referred to HALT mode in this IC. 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 Cortex®-M0+ core executes the WFI or WFE instruction with the SLEEPDEEP bit of the Cortex®-M0+ System Control Register set to 1, it suspends program execution and stops operating. This state is referred to SLEEP mode in this IC. 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.

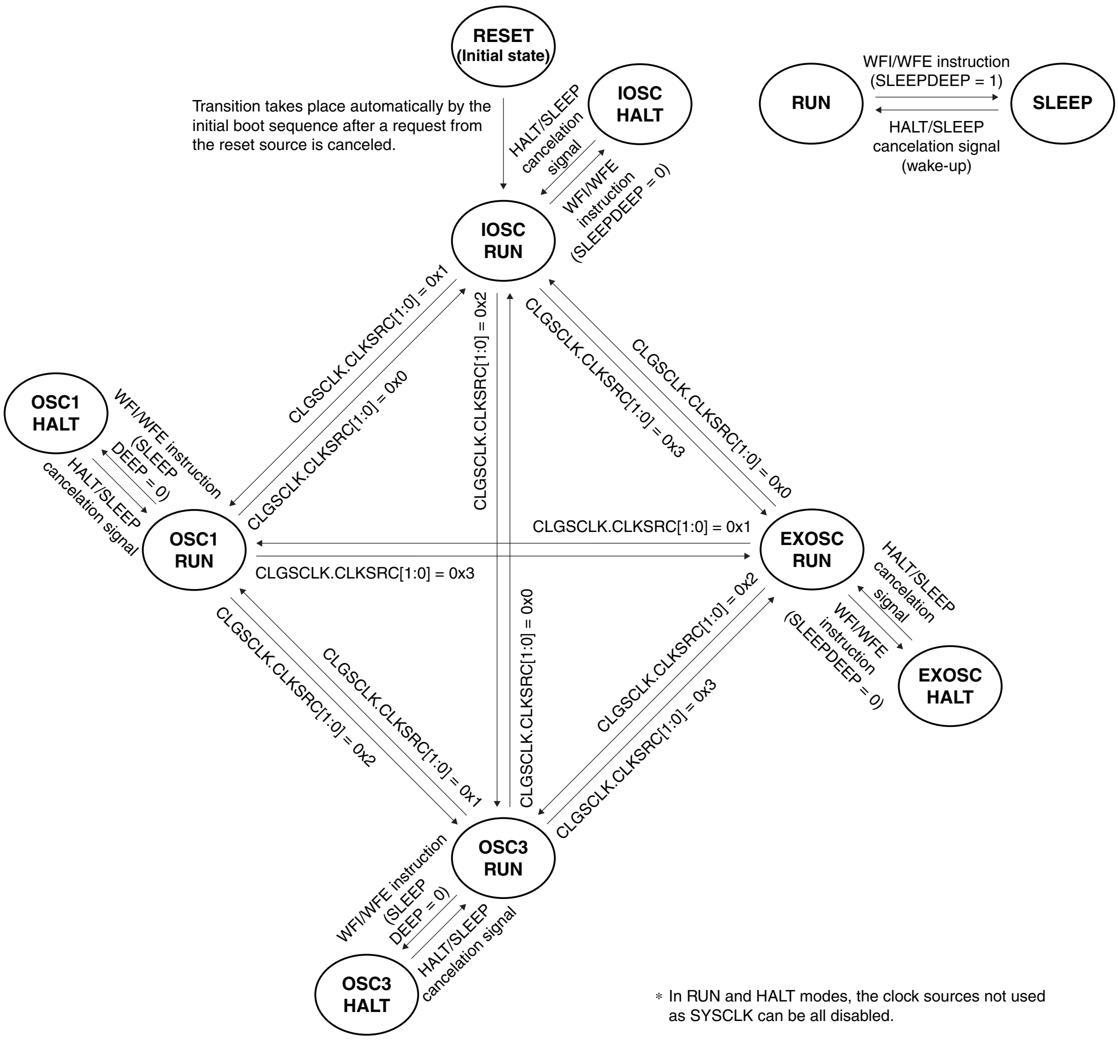


Figure 2.4.2.1 Operating Mode-to-Mode State Transition Diagram

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.

- Interrupt request from a peripheral circuit
- NMI from the watchdog timer
- Reset request

2.5 Interrupts

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

Table 2.5.1 CLG Interrupt Functions

Interrupt	Interrupt flag	Set condition	Clear condition
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
OSC3 oscillation auto-trimming completion	CLGINTF.OSC3TEDIF	When the OSC3 oscillation auto-trimming operation has completed	Writing 1
OSC3 oscillation auto-trimming error	CLGINTF.OSC3TERIF	When the OSC3 oscillation auto-trimming operation has terminated due to an error	Writing 1

CLG provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU 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” chapter.

2.6 Control Registers

PWGA Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PWGACTL	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5	REGDIS	0	H0	R/WP	
	4	REGSEL	1	H0	R/WP	
	3–2	–	0x0	–	R	
	1–0	REGMODE[1:0]	0x0	H0	R/WP	

Bits 15–6 Reserved

Bit 5 REGDIS

This bit enables the V_{D1} regulator discharge function.

1 (R/WP): Enable

0 (R/WP): Disable

Bit 4 REGSEL

This bit controls the V_{D1} regulator voltage mode.

1 (R/WP): mode0

0 (R/WP): mode1

Bits 3–2 Reserved

Bits 1–0 REGMODE[1:0]

These bits control the V_{D1} regulator operating mode.

Table 2.6.1 Internal Regulator Operating Mode

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

CLG System Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGSCLK	15	WUPMD	0	H0	R/WP	–
	14	–	0	–	R	
	13–12	WUPDIV[1:0]	0x0	H0	R/WP	
	11–10	–	0x0	–	R	
	9–8	WUPSRC[1:0]	0x0	H0	R/WP	
	7–6	–	0x0	–	R	
	5–4	CLKDIV[1:0]	0x2	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.

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. 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.

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

CLGSCLK.WUPDIV[1:0] bits	CLGSCLK.WUPSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	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 2.6.3 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	–
	7–2	–	0x00	–	R	
	1–0	IOSCFQ[1:0]	0x2	H0	R/WP	

Bits 15–2 Reserved

Bits 1–0 IOSCFQ[1:0]

These bits select the IOSCCLK frequency.

Table 2.6.4 IOSCCLK Frequency Selection

CLGIOSC. IOSCFQ[1:0] bits	IOSCCLK frequency	
	VD1 voltage mode = mode0	VD1 voltage mode = mode1
0x3	Reserved	Setting prohibited
0x2	8 MHz	
0x1	2.0 MHz	1.8 MHz
0x0	1.0 MHz	0.9 MHz

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	OSC1SELCR	0	H0	R/WP	
	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 OSC1SELCR

This bit selects an oscillator type of the OSC1 oscillator circuit.

1 (R/WP): Internal oscillator

0 (R/WP): Crystal oscillator

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]	0x1	H0	R/WP	
	9	OSC3MD	0	H0	R/WP	
	8	–	0	–	R	
	7–6	–	0x0	–	R	
	5–4	OSC3INV[1:0]	0x3	H0	R/WP	
	3	OSC3STM	0	H0	R/WP	
	2–0	OSC3WT[2:0]	0x6	H0	R/WP	

Bits 15–12 Reserved

Bit 11–10 OSC3FQ[1:0]

These bits set the oscillation frequency of the OSC3 internal oscillator circuit.

Table 2.6.9 OSC3CLK Frequency Selection

CLGOSC3.OSC3FQ[1:0] bits	OSC3CLK frequency
0x3	16 MHz
0x2	Reserved
0x1	8 MHz
0x0	4 MHz

Bit 9 OSC3MD

This bit selects an oscillator type of the OSC3 oscillator circuit.
1 (R/WP): Crystal/ceramic oscillator
0 (R/WP): Internal oscillator

Bits 8–6 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 2.6.10 OSC3 Oscillation Inverter Gain Setting

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

Bit 3 OSC3STM

This bit controls the OSC3CLK 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:
- 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.
- Be sure to avoid altering the CLGOSC3.OSC3FQ[1:0] bits while the auto-trimming is being executed.

Bits 2–0 OSC3WT[2:0]

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

Table 2.6.11 OSC3 Oscillation Stabilization Waiting Time Setting

CLGOSC3.OSC3WT[2:0] bits	Oscillation stabilization waiting time
0x7	65,536 clocks
0x6	16,384 clocks
0x5	8,192 clocks
0x4	4,096 clocks
0x3–0x0	Setting prohibited

CLG Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGINTF	15–9	–	0x00	–	R	–
	8	OSC3TERIF	0	H0	R/W	Cleared by writing 1.
	7	–	0	–	R	–
	6	(reserved)	0	H0	R	
	5	OSC1STPIF	0	H0	R/W	Cleared by writing 1.
	4	OSC3TEDIF	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 15–9, 7, 6, 3 Reserved

Bit 8 OSC3TERIF
Bit 5 OSC1STPIF
Bit 4 OSC3TEDIF
Bit 2 OSC3STAIF
Bit 1 OSC1STAIF
Bit 0 IOSCSTAIF

These bits indicate the CLG 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.OSC3TERIF bit: OSC3 oscillation auto-trimming error interrupt
CLGINTF.OSC1STPIF bit: OSC1 oscillation stop interrupt
CLGINTF.OSC3TEDIF bit: OSC3 oscillation auto-trimming completion interrupt
CLGINTF.OSC3STAIF bit: OSC3 oscillation stabilization waiting completion interrupt
CLGINTF.OSC1STAIF bit: OSC1 oscillation stabilization waiting completion interrupt
CLGINTF.IOSCSTAIF bit: IOSC oscillation stabilization waiting completion interrupt

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	15–9	–	0x00	–	R	–
	8	OSC3TERIE	0	H0	R/W	
	7	–	0	–	R	
	6	(reserved)	0	H0	R/W	
	5	OSC1STPIE	0	H0	R/W	
	4	OSC3TEDIE	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 15–9, 7, 6, 3 Reserved

Bit 8 OSC3TERIE
Bit 5 OSC1STPIE
Bit 4 OSC3TEDIE
Bit 2 OSC3STAIE
Bit 1 OSC1STAIE
Bit 0 IOSCSTAIE

These bits enable the CLG interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

Each bit corresponds to the interrupt as follows:
CLGINTE.OSC3TERIE bit: OSC3 oscillation auto-trimming error interrupt
CLGINTE.OSC1STPIE bit: OSC1 oscillation stop interrupt
CLGINTE.OSC3TEDIE bit: OSC3 oscillation auto-trimming completion interrupt
CLGINTE.OSC3STAIE bit: OSC3 oscillation stabilization waiting completion interrupt
CLGINTE.OSC1STAIE bit: OSC1 oscillation stabilization waiting completion interrupt
CLGINTE.IOSCSTAIE bit: IOSC oscillation stabilization waiting completion interrupt

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 2.6.12 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.

CLG Oscillation Frequency Trimming Register 1

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGTRIM1	15–14	–	0x0	–	R	–
	13–8	IOSCLSAJ[5:0]	*	H0	R/WP	* Determined by factory adjustment.
	7–6	–	0x0	–	R	–
	5–0	IOSCHSAJ[5:0]	*	H0	R/WP	* Determined by factory adjustment.

Bits 15–14 Reserved

Bits 13–8 IOSCLSAJ[5:0]

These bits set the frequency trimming value for the IOSC internal oscillator circuit.
This setting affects the low-speed oscillation frequencies (1 MHz and 2 MHz).

Table 2.6.13 Low-Speed Oscillation Frequency Trimming Setting of IOSC Internal Oscillator Circuit

CLGTRIM1.IOSCLSAJ[5:0] bits	IOSC oscillation frequency (2/1 MHz)
0x3f	High
:	:
0x00	Low

Bits 7–6 Reserved

Bits 5–0 IOSCHSAJ[5:0]

These bits set the frequency trimming value for the IOSC internal oscillator circuit.
This setting affects the high-speed oscillation frequency (8 MHz).

Table 2.6.14 High-Speed Oscillation Frequency Trimming Setting of IOSC Internal Oscillator Circuit

CLGTRIM1.IOSCHSAJ[5:0] bits	IOSC oscillation frequency (8 MHz)
0x3f	High
:	:
0x00	Low

Note: The initial values of the CLGTRIM1.IOSCLSAJ[5:0] and CLGTRIM1.IOSCHSAJ[5:0] bits were adjusted so that the IOSC oscillator circuit characteristics described in the “Electrical Characteristics” chapter can be guaranteed. Be aware that the frequency characteristics may not be satisfied when these settings are altered. When altering these settings, always make sure that the IOSC oscillator circuit is inactive.

CLG Oscillation Frequency Trimming Register 2

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGTRIM2	15–8	–	0x00	–	R	–
	7–6	–	0x0	–	R	
	5–0	OSC1SAJ[5:0]	*	H0	R/WP	* Determined by factory adjustment.

Bits 15–6 Reserved

Bits 5–0 OSC1SAJ[5:0]

These bits set the frequency trimming value for the OSC1 internal oscillator circuit.
This setting does not affect the OSC1 crystal oscillation frequency.

Table 2.6.15 Oscillation Frequency Trimming Setting of OSC1 Internal Oscillator Circuit

CLGTRIM2.OSC1SAJ[5:0] bits	OSC1 internal oscillator frequency
0x3f	High
:	:
0x00	Low

Note: The initial value of the CLGTRIM2.OSC1SAJ[5:0] bits was adjusted so that the OSC1 oscillator circuit characteristics described in the “Electrical Characteristics” chapter can be guaranteed. Be aware that the frequency characteristic may not be satisfied when this setting is altered. When altering this setting, always make sure that the OSC1 oscillator circuit is inactive.

CLG Oscillation Frequency Trimming Register 3

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CLGTRIM3	15–9	–	0x00	–	R	–
	8–0	OSC3SAJ[8:0]	*	H0	R/WP	* Determined by factory adjustment.

Bits 15–9 Reserved

Bits 8–0 OSC3SAJ[8:0]

These bits set the frequency trimming value for the OSC3 internal oscillator circuit.
This setting does not affect the OSC3 crystal/ceramic oscillation frequency.

Table 2.6.16 Oscillation Frequency Trimming Setting of OSC3 Internal Oscillator Circuit

CLGTRIM3.OSC3SAJ[8:0] bits	OSC3 internal oscillator frequency
0x1ff	High
:	:
0x00	Low

Note: The initial value of the CLGTRIM3.OSC3SAJ[8:0] bits was adjusted so that the OSC3 oscillator circuit characteristics described in the "Electrical Characteristics" chapter can be guaranteed. Be aware that the frequency characteristic may not be satisfied when this setting is altered. When altering this setting, always make sure that the OSC3 oscillator circuit is inactive.

3 CPU and Debugger

3.1 Overview

This IC incorporates a Cortex®-M0+ CPU manufactured by Arm Ltd.

3.2 CPU

The following shows the system configuration of the Cortex®-M0+ CPU embedded in this IC:

- Cortex®-M0+ core
- 32-bit single-cycle multiplier
- Nested vectored interrupt controller (NVIC)
- System timer (Systick)
- Serial-wire debug port (SW-DP)
- Micro trace buffer (MTB)
- Number of hardware break points: 4
- Number of watch points: 2

3.3 Debugger

This IC includes a serial-wire debug port (SW-DP).

3.3.1 List of Debugger Input/Output Pins

Table 3.3.3.1 lists the debug pins.

Table 3.3.1.1 List of Debug Pins

Pin name	I/O	Initial state	Function
SWCLK	O	O	On-chip debugger clock input pin Input a clock from a debugging tool.
SWD	I/O	I	On-chip debugger data input/output pin Used to input/output debugging data.

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

3.3.2 External Connection

Figure 3.3.2.1 shows a connection example between this IC and a debugging tool when performing debugging.

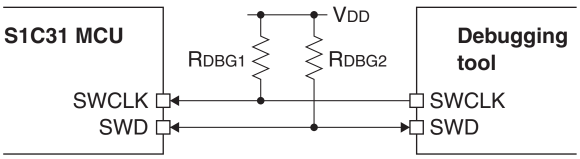


Figure 3.3.2.1 External Connection

For the recommended pull-up resistor value, refer to "Recommended Operating Conditions, Debug pin pull-up resistors R_{DBG1-2}" in the "Electrical Characteristics" chapter. R_{DBG1} and R_{DBG2} are not required when using the debug pins as general-purpose I/O port pins.

3.4 Reference Documents

Arm Ltd. provides various documents for developing a system with a Cortex®-M0+ CPU included. For detailed information on the Cortex®-M0+ CPU that are not described in this manual, refer to the following documents:

1. ARM®v6-M Architecture Reference Manual
2. Cortex®-M0+Technical Reference Manual
3. Cortex®-M0+ Devices Generic User Guide

These documents can be downloaded from the document site of Arm Ltd.

https://developer.arm.com/documentation

4 Memory and Bus

4.1 Overview

This IC supports up to 4G bytes of accessible memory space for both instructions and data.
The features are listed below.

- Embedded Flash memory that supports on-board programming
- Write-protect function to protect system control registers

Figure 4.1.1 shows the memory map.

S1C31D41

Address (end)	Address (start)	Area
0xffff ffff	0xf022 2000	Reserved
0xf022 1fff	0xf022 0000	MTB SRAM area (8K bytes) (Device size: 32 bits)
0xf021 ffff	0xf020 1000	Reserved
0xf020 0fff	0xf020 0000	MTB SFR area (4K bytes) (Device size: 32 bits)
0xf01f ffff	0xf000 1000	Reserved
0xf000 0fff	0xf000 0000	System ROM table area (4K bytes) (Device size: 32 bits)
0xefff ffff	0xe000 0000	PPB and reserved area for Cortex®-M0+ (Device size: 32 bits)
0xdfff ffff	0x0020 4000	Reserved
0x0020 3fff	0x0020 1000	Peripheral circuit area (12K bytes) (Device size: 32 bits)
0x0020 0fff	0x0020 0000	Peripheral circuit area (4K bytes) (Device size: 16 bits)
0x001f ffff	0x0015 6800	Reserved
0x0015 67ff	0x0015 2000	Voice RAM area (18K bytes) (Device size: 32 bits)
0x0015 1fff	0x0015 0000	RAM area (8K bytes) (Device size: 32 bits)
0x0014 ffff	0x0014 0000	Reserved
0x0013 ffff	0x0004 0000	Memory mapped access area for external Flash memory (1M bytes) (Device size: 32 bits)
0x0003 ffff	0x0001 8000	Reserved
0x0001 7fff	0x0000 0000	Flash area (96K bytes) (Device size: 32 bits)

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., LDR Rt, [Rn] → 32-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- or 16-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

4.3 Flash Memory

The Flash memory is used to store application programs and data. Address 0x0 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" 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

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 using a flash loader. The V_{PP} voltage is supplied from the internal voltage booster.
Be sure to connect C_{VPP} between the V_{SS} and V_{PP} pins for generating the voltage using the internal power supply.

Notes:
- When programming the Flash memory, 2.2 V or more V_{DD} voltage is required.
- Be sure to avoid using the V_{PP} pin output for driving external circuits.

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.

4.5 Peripheral Circuit Control Registers

The control registers for the peripheral circuits are located in the peripheral circuit area beginning with address 0x0020 0000. Table 4.5.1 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 4.5.1 Peripheral Circuit Control Register Map

Peripheral circuit	Address	Register name	
System register (SYS)	0x0020 0000	SYSPROT	System Protect Register
Power generator (PWGA)	0x0020 0020	PWGACTL	PWGA Control Register
Clock generator(CLG)	0x0020 0040	CLGSCLK	CLG System Clock Control Register
	0x0020 0042	CLGOSC	CLG Oscillation Control Register
	0x0020 0044	CLGIOSC	CLG IOSC Control Register
	0x0020 0046	CLGOSC1	CLG OSC1 Control Register
	0x0020 0048	CLGOSC3	CLG OSC3 Control Register
	0x0020 004c	CLGINTF	CLG Interrupt Flag Register
	0x0020 004e	CLGINTE	CLG Interrupt Enable Register
	0x0020 0050	CLGFOUT	CLG FOUT Control Register
	0x4000 0052	CLGTRIM1	CLG Oscillation Frequency Trimming Register 1
	0x4000 0054	CLGTRIM2	CLG Oscillation Frequency Trimming Register 2
	0x4000 005a	CLGTRIM3	CLG Oscillation Frequency Trimming Register 3
Cache controller (CACHE)	0x0020 0080	CACHECTL	CACHE Control Register
Watchdog timer (WDT2)	0x0020 00a0	WDT2CLK	WDT2 Clock Control Register
	0x0020 00a2	WDT2CTL	WDT2 Control Register
	0x0020 00a4	WDT2CMP	WDT2 Counter Compare Match Register
Real-time clock (RTCA)	0x0020 00c0	RTCACTLL	RTCA Control Register (Low Byte)
	0x0020 00c1	RTCACTLH	RTCA Control Register (High Byte)
	0x0020 00c2	RTCAALM1	RTCA Second Alarm Register
	0x0020 00c4	RTCAALM2	RTCA Hour/Minute Alarm Register
	0x0020 00c6	RTCASWCTL	RTCA Stopwatch Control Register
	0x0020 00c8	RTCASEC	RTCA Second/1Hz Register
	0x0020 00ca	RTCAHUR	RTCA Hour/Minute Register
	0x0020 00cc	RTCAMON	RTCA Month/Day Register
	0x0020 00ce	RTCAYAR	RTCA Year/Week Register
	0x0020 00d0	RTCAINTF	RTCA Interrupt Flag Register
	0x0020 00d2	RTCAINTE	RTCA Interrupt Enable Register
Supply voltage detector (SVD3)	0x0020 0100	SVD3CLK	SVD3 Clock Control Register
	0x0020 0102	SVD3CTL	SVD3 Control Register
	0x0020 0104	SVD3INTF	SVD3 Status and Interrupt Flag Register
	0x0020 0106	SVD3INTE	SVD3 Interrupt Enable Register
16-bit timer (T16) Ch.0	0x0020 0160	T16_0CLK	T16 Ch.0 Clock Control Register
	0x0020 0162	T16_0MOD	T16 Ch.0 Mode Register
	0x0020 0164	T16_0CTL	T16 Ch.0 Control Register
	0x0020 0166	T16_0TR	T16 Ch.0 Reload Data Register
	0x0020 0168	T16_0TC	T16 Ch.0 Counter Data Register
	0x0020 016a	T16_0INTF	T16 Ch.0 Interrupt Flag Register
	0x0020 016c	T16_0INTE	T16 Ch.0 Interrupt Enable Register
Flash controller (FLASHC)	0x0020 01b0	FLASHCWAIT	FLASHC Flash Read Cycle Register
I/O ports (PPORT)	0x0020 0200	PPORTP0DAT	P0 Port Data Register
	0x0020 0202	PPORTP0IOEN	P0 Port Enable Register
	0x0020 0204	PPORTP0RCTL	P0 Port Pull-up/down Control Register
	0x0020 0206	PPORTP0INTF	P0 Port Interrupt Flag Register
	0x0020 0208	PPORTP0INTCTL	P0 Port Interrupt Control Register
	0x0020 020a	PPORTP0CHATEN	P0 Port Chattering Filter Enable Register
	0x0020 020c	PPORTP0MODSEL	P0 Port Mode Select Register
	0x0020 020e	PPORTP0FNCSEL	P0 Port Function Select Register
	0x0020 0210	PPORTP1DAT	P1 Port Data Register
	0x0020 0212	PPORTP1IOEN	P1 Port Enable Register
	0x0020 0214	PPORTP1RCTL	P1 Port Pull-up/down Control Register

Peripheral circuit	Address	Register name	
I/O ports (PPORT)	0x0020 0216	PPORTP1INTF	P1 Port Interrupt Flag Register
	0x0020 0218	PPORTP1INTCTL	P1 Port Interrupt Control Register
	0x0020 021a	PPORTP1CHATEN	P1 Port Chattering Filter Enable Register
	0x0020 021c	PPORTP1MODSEL	P1 Port Mode Select Register
	0x0020 021e	PPORTP1FNCSEL	P1 Port Function Select Register
	0x0020 0220	PPORTP2DAT	P2 Port Data Register
	0x0020 0222	PPORTP2IOEN	P2 Port Enable Register
	0x0020 0224	PPORTP2RCTL	P2 Port Pull-up/down Control Register
	0x0020 0226	PPORTP2INTF	P2 Port Interrupt Flag Register
	0x0020 0228	PPORTP2INTCTL	P2 Port Interrupt Control Register
	0x0020 022a	PPORTP2CHATEN	P2 Port Chattering Filter Enable Register
	0x0020 022c	PPORTP2MODSEL	P2 Port Mode Select Register
	0x0020 022e	PPORTP2FNCSEL	P2 Port Function Select Register
	0x0020 0230	PPORTP3DAT	P3 Port Data Register
	0x0020 0232	PPORTP3IOEN	P3 Port Enable Register
	0x0020 0234	PPORTP3RCTL	P3 Port Pull-up/down Control Register
	0x0020 0236	PPORTP3INTF	P3 Port Interrupt Flag Register
	0x0020 0238	PPORTP3INTCTL	P3 Port Interrupt Control Register
	0x0020 023a	PPORTP3CHATEN	P3 Port Chattering Filter Enable Register
	0x0020 023c	PPORTP3MODSEL	P3 Port Mode Select Register
	0x0020 023e	PPORTP3FNCSEL	P3 Port Function Select Register
	0x0020 0240	PPORTP4DAT	P4 Port Data Register
	0x0020 0242	PPORTP4IOEN	P4 Port Enable Register
	0x0020 0244	PPORTP4RCTL	P4 Port Pull-up/down Control Register
	0x0020 0246	PPORTP4INTF	P4 Port Interrupt Flag Register
	0x0020 0248	PPORTP4INTCTL	P4 Port Interrupt Control Register
	0x0020 024a	PPORTP4CHATEN	P4 Port Chattering Filter Enable Register
	0x0020 024c	PPORTP4MODSEL	P4 Port Mode Select Register
	0x0020 024e	PPORTP4FNCSEL	P4 Port Function Select Register
	0x0020 0250	PPORTP5DAT	P5 Port Data Register
	0x0020 0252	PPORTP5IOEN	P5 Port Enable Register
	0x0020 0254	PPORTP5RCTL	P5 Port Pull-up/down Control Register
	0x0020 0256	PPORTP5INTF	P5 Port Interrupt Flag Register
	0x0020 0258	PPORTP5INTCTL	P5 Port Interrupt Control Register
	0x0020 025a	PPORTP5CHATEN	P5 Port Chattering Filter Enable Register
	0x0020 025c	PPORTP5MODSEL	P5 Port Mode Select Register
	0x0020 025e	PPORTP5FNCSEL	P5 Port Function Select Register
	0x0020 0260	PPORTP6DAT	P6 Port Data Register
	0x0020 0262	PPORTP6IOEN	P6 Port Enable Register
	0x0020 0264	PPORTP6RCTL	P6 Port Pull-up/down Control Register
	0x0020 0266	PPORTP6INTF	P6 Port Interrupt Flag Register
	0x0020 0268	PPORTP6INTCTL	P6 Port Interrupt Control Register
	0x0020 026a	PPORTP6CHATEN	P6 Port Chattering Filter Enable Register
	0x0020 026c	PPORTP6MODSEL	P6 Port Mode Select Register
	0x0020 026e	PPORTP6FNCSEL	P6 Port Function Select Register
	0x0020 02d0	PPORTPDDAT	Pd Port Data Register
	0x0020 02d2	PPORTPDIOEN	Pd Port Enable Register
	0x0020 02d4	PPORTPDRCTL	Pd Port Pull-up/down Control Register
	0x0020 02dc	PPORTPDMODSEL	Pd Port Mode Select Register
	0x0020 02de	PPORTPDFNCSEL	Pd Port Function Select Register
	0x0020 02e0	PPORTCLK	P Port Clock Control Register
	0x0020 02e2	PPORTINTFGRP	P Port Interrupt Flag Group Register
Universal port multiplexer (UPMUX)	0x0020 0300	UPMUXP0MUX0	P00–01 Universal Port Multiplexer Setting Register
	0x0020 0302	UPMUXP0MUX1	P02–03 Universal Port Multiplexer Setting Register
	0x0020 0304	UPMUXP0MUX2	P04–05 Universal Port Multiplexer Setting Register
	0x0020 0306	UPMUXP0MUX3	P06–07 Universal Port Multiplexer Setting Register
	0x0020 0308	UPMUXP1MUX0	P10–11 Universal Port Multiplexer Setting Register
	0x0020 030a	UPMUXP1MUX1	P12–13 Universal Port Multiplexer Setting Register
	0x0020 030c	UPMUXP1MUX2	P14–15 Universal Port Multiplexer Setting Register
	0x0020 030e	UPMUXP1MUX3	P16–17 Universal Port Multiplexer Setting Register
	0x0020 0310	UPMUXP2MUX0	P20–21 Universal Port Multiplexer Setting Register
	0x0020 0312	UPMUXP2MUX1	P22–23 Universal Port Multiplexer Setting Register
	0x0020 0314	UPMUXP2MUX2	P24–25 Universal Port Multiplexer Setting Register
	0x0020 0316	UPMUXP2MUX3	P26–27 Universal Port Multiplexer Setting Register

Peripheral circuit	Address	Register name	
Universal port multiplexer (UPMUX)	0x0020 0318	UPMUXP3MUX0	P30–31 Universal Port Multiplexer Setting Register
	0x0020 031a	UPMUXP3MUX1	P32–33 Universal Port Multiplexer Setting Register
	0x0020 031c	UPMUXP3MUX2	P34–35 Universal Port Multiplexer Setting Register
	0x0020 031e	UPMUXP3MUX3	P36–37 Universal Port Multiplexer Setting Register
UART (UART3) Ch.0	0x0020 0380	UART3_0CLK	UART3 Ch.0 Clock Control Register
	0x0020 0382	UART3_0MOD	UART3 Ch.0 Mode Register
	0x0020 0384	UART3_0BR	UART3 Ch.0 Baud-Rate Register
	0x0020 0386	UART3_0CTL	UART3 Ch.0 Control Register
	0x0020 0388	UART3_0TXD	UART3 Ch.0 Transmit Data Register
	0x0020 038a	UART3_0RXD	UART3 Ch.0 Receive Data Register
	0x0020 038c	UART3_0INTF	UART3 Ch.0 Status and Interrupt Flag Register
	0x0020 038e	UART3_0INTE	UART3 Ch.0 Interrupt Enable Register
	0x0020 0390	UART3_0 TBEDMAEN	UART3 Ch.0 Transmit Buffer Empty DMA Request Enable Register
	0x0020 0392	UART3_0 RB1FDMAEN	UART3 Ch.0 Receive Buffer One Byte Full DMA Request Enable Register
	0x0020 0394	UART3_0CAWF	UART3 Ch.0 Carrier Waveform Register
16-bit timer (T16) Ch.1	0x0020 03a0	T16_1CLK	T16 Ch.1 Clock Control Register
	0x0020 03a2	T16_1MOD	T16 Ch.1 Mode Register
	0x0020 03a4	T16_1CTL	T16 Ch.1 Control Register
	0x0020 03a6	T16_1TR	T16 Ch.1 Reload Data Register
	0x0020 03a8	T16_1TC	T16 Ch.1 Counter Data Register
	0x0020 03aa	T16_1INTF	T16 Ch.1 Interrupt Flag Register
	0x0020 03ac	T16_1INTE	T16 Ch.1 Interrupt Enable Register
Synchronous serial interface (SPIA) Ch.0	0x0020 03b0	SPIA_0MOD	SPIA Ch.0 Mode Register
	0x0020 03b2	SPIA_0CTL	SPIA Ch.0 Control Register
	0x0020 03b4	SPIA_0TXD	SPIA Ch.0 Transmit Data Register
	0x0020 03b6	SPIA_0RXD	SPIA Ch.0 Receive Data Register
	0x0020 03b8	SPIA_0INTF	SPIA Ch.0 Interrupt Flag Register
	0x0020 03ba	SPIA_0INTE	SPIA Ch.0 Interrupt Enable Register
	0x0020 03bc	SPIA_0TBEDMAEN	SPIA Ch.0 Transmit Buffer Empty DMA Request Enable Register
	0x0020 03be	SPIA_0RBFDMAEN	SPIA Ch.0 Receive Buffer Full DMA Request Enable Register
I^{2}C (I2C) Ch.0	0x0020 03c0	I2C_0CLK	I2C Ch.0 Clock Control Register
	0x0020 03c2	I2C_0MOD	I2C Ch.0 Mode Register
	0x0020 03c4	I2C_0BR	I2C Ch.0 Baud-Rate Register
	0x0020 03c8	I2C_0OADR	I2C Ch.0 Own Address Register
	0x0020 03ca	I2C_0CTL	I2C Ch.0 Control Register
	0x0020 03cc	I2C_0TXD	I2C Ch.0 Transmit Data Register
	0x0020 03ce	I2C_0RXD	I2C Ch.0 Receive Data Register
	0x0020 03d0	I2C_0INTF	I2C Ch.0 Status and Interrupt Flag Register
	0x0020 03d2	I2C_0INTE	I2C Ch.0 Interrupt Enable Register
	0x0020 03d4	I2C_0TBEDMAEN	I2C Ch.0 Transmit Buffer Empty DMA Request Enable Register
	0x0020 03d6	I2C_0RBFDMAEN	I2C Ch.0 Receive Buffer Full DMA Request Enable Register
16-bit PWM timer (T16B) Ch.0	0x0020 0400	T16B_0CLK	T16B Ch.0 Clock Control Register
	0x0020 0402	T16B_0CTL	T16B Ch.0 Counter Control Register
	0x0020 0404	T16B_0MC	T16B Ch.0 Max Counter Data Register
	0x0020 0406	T16B_0TC	T16B Ch.0 Timer Counter Data Register
	0x0020 0408	T16B_0CS	T16B Ch.0 Counter Status Register
	0x0020 040a	T16B_0INTF	T16B Ch.0 Interrupt Flag Register
	0x0020 040c	T16B_0INTE	T16B Ch.0 Interrupt Enable Register
	0x0020 040e	T16B_0MZDMAEN	T16B Ch.0 Counter Max/Zero DMA Request Enable Register
	0x0020 0410	T16B_0CCCTL0	T16B Ch.0 Compare/Capture 0 Control Register
	0x0020 0412	T16B_0CCR0	T16B Ch.0 Compare/Capture 0 Data Register
	0x0020 0414	T16B_0CC0DMAEN	T16B Ch.0 Compare/Capture 0 DMA Request Enable Register
	0x0020 0418	T16B_0CCCTL1	T16B Ch.0 Compare/Capture 1 Control Register
	0x0020 041a	T16B_0CCR1	T16B Ch.0 Compare/Capture 1 Data Register
	0x0020 041c	T16B_0CC1DMAEN	T16B Ch.0 Compare/Capture 1 DMA Request Enable Register
	0x0020 0420	T16B_0CCCTL2	T16B Ch.0 Compare/Capture 2 Control Register
	0x0020 0422	T16B_0CCR2	T16B Ch.0 Compare/Capture 2 Data Register
	0x0020 0424	T16B_0CC2DMAEN	T16B Ch.0 Compare/Capture 2 DMA Request Enable Register
	0x0020 0428	T16B_0CCCTL3	T16B Ch.0 Compare/Capture 3 Control Register
	0x0020 042a	T16B_0CCR3	T16B Ch.0 Compare/Capture 3 Data Register
	0x0020 042c	T16B_0CC3DMAEN	T16B Ch.0 Compare/Capture 3 DMA Request Enable Register
16-bit PWM timer (T16B) Ch.1	0x0020 0440	T16B_1CLK	T16B Ch.1 Clock Control Register
	0x0020 0442	T16B_1CTL	T16B Ch.1 Counter Control Register

Peripheral circuit	Address	Register name	
16-bit PWM timer (T16B) Ch.1	0x0020 0444	T16B_1MC	T16B Ch.1 Max Counter Data Register
	0x0020 0446	T16B_1TC	T16B Ch.1 Timer Counter Data Register
	0x0020 0448	T16B_1CS	T16B Ch.1 Counter Status Register
	0x0020 044a	T16B_1INTF	T16B Ch.1 Interrupt Flag Register
	0x0020 044c	T16B_1INTE	T16B Ch.1 Interrupt Enable Register
	0x0020 044e	T16B_1MZDMAEN	T16B Ch.1 Counter Max/Zero DMA Request Enable Register
	0x0020 0450	T16B_1CCCTL0	T16B Ch.1 Compare/Capture 0 Control Register
	0x0020 0452	T16B_1CCR0	T16B Ch.1 Compare/Capture 0 Data Register
	0x0020 0454	T16B_1CC0DMAEN	T16B Ch.1 Compare/Capture 0 DMA Request Enable Register
	0x0020 0458	T16B_1CCCTL1	T16B Ch.1 Compare/Capture 1 Control Register
	0x0020 045a	T16B_1CCR1	T16B Ch.1 Compare/Capture 1 Data Register
	0x0020 045c	T16B_1CC1DMAEN	T16B Ch.1 Compare/Capture 1 DMA Request Enable Register
	0x0020 0460	T16B_1CCCTL2	T16B Ch.1 Compare/Capture 2 Control Register
	0x0020 0462	T16B_1CCR2	T16B Ch.1 Compare/Capture 2 Data Register
	0x0020 0464	T16B_1CC2DMAEN	T16B Ch.1 Compare/Capture 2 DMA Request Enable Register
	0x0020 0468	T16B_1CCCTL3	T16B Ch.1 Compare/Capture 3 Control Register
	0x0020 046a	T16B_1CCR3	T16B Ch.1 Compare/Capture 3 Data Register
	0x0020 046c	T16B_1CC3DMAEN	T16B Ch.1 Compare/Capture 3 DMA Request Enable Register
16-bit timer (T16) Ch.3	0x0020 0480	T16_3CLK	T16 Ch.3 Clock Control Register
	0x0020 0482	T16_3MOD	T16 Ch.3 Mode Register
	0x0020 0484	T16_3CTL	T16 Ch.3 Control Register
	0x0020 0486	T16_3TR	T16 Ch.3 Reload Data Register
	0x0020 0488	T16_3TC	T16 Ch.3 Counter Data Register
	0x0020 048a	T16_3INTF	T16 Ch.3 Interrupt Flag Register
	0x0020 048c	T16_3INTE	T16 Ch.3 Interrupt Enable Register
16-bit timer (T16) Ch.4	0x0020 04a0	T16_4CLK	T16 Ch.4 Clock Control Register
	0x0020 04a2	T16_4MOD	T16 Ch.4 Mode Register
	0x0020 04a4	T16_4CTL	T16 Ch.4 Control Register
	0x0020 04a6	T16_4TR	T16 Ch.4 Reload Data Register
	0x0020 04a8	T16_4TC	T16 Ch.4 Counter Data Register
	0x0020 04aa	T16_4INTF	T16 Ch.4 Interrupt Flag Register
	0x0020 04ac	T16_4INTE	T16 Ch.4 Interrupt Enable Register
16-bit timer (T16) Ch.5	0x0020 04c0	T16_5CLK	T16 Ch.5 Clock Control Register
	0x0020 04c2	T16_5MOD	T16 Ch.5 Mode Register
	0x0020 04c4	T16_5CTL	T16 Ch.5 Control Register
	0x0020 04c6	T16_5TR	T16 Ch.5 Reload Data Register
	0x0020 04c8	T16_5TC	T16 Ch.5 Counter Data Register
	0x0020 04ca	T16_5INTF	T16 Ch.5 Interrupt Flag Register
	0x0020 04cc	T16_5INTE	T16 Ch.5 Interrupt Enable Register
Synchronous serial interface (SPIA) Ch.2	0x0020 04d0	SPIA_2MOD	SPIA Ch.2 Mode Register
	0x0020 04d2	SPIA_2CTL	SPIA Ch.2 Control Register
	0x0020 04d4	SPIA_2TXD	SPIA Ch.2 Transmit Data Register
	0x0020 04d6	SPIA_2RXD	SPIA Ch.2 Receive Data Register
	0x0020 04d8	SPIA_2INTF	SPIA Ch.2 Interrupt Flag Register
	0x0020 04da	SPIA_2INTE	SPIA Ch.2 Interrupt Enable Register
	0x0020 04dc	SPIA_2TBEDMAEN	SPIA Ch.2 Transmit Buffer Empty DMA Request Enable Register
	0x0020 04de	SPIA_2RBFDMAEN	SPIA Ch.2 Receive Buffer Full DMA Request Enable Register
UART (UART3) Ch.1	0x0020 0600	UART3_1CLK	UART3 Ch.1 Clock Control Register
	0x0020 0602	UART3_1MOD	UART3 Ch.1 Mode Register
	0x0020 0604	UART3_1BR	UART3 Ch.1 Baud-Rate Register
	0x0020 0606	UART3_1CTL	UART3 Ch.1 Control Register
	0x0020 0608	UART3_1TXD	UART3 Ch.1 Transmit Data Register
	0x0020 060a	UART3_1RXD	UART3 Ch.1 Receive Data Register
	0x0020 060c	UART3_1INTF	UART3 Ch.1 Status and Interrupt Flag Register
	0x0020 060e	UART3_1INTE	UART3 Ch.1 Interrupt Enable Register
	0x0020 0610	UART3_1 TBEDMAEN	UART3 Ch.1 Transmit Buffer Empty DMA Request Enable Register
	0x0020 0612	UART3_1 RB1FDMAEN	UART3 Ch.1 Receive Buffer One Byte Full DMA Request Enable Register
	0x0020 0614	UART3_1CAWF	UART3 Ch.1 Carrier Waveform Register
UART (UART3) Ch.2	0x0020 0620	UART3_2CLK	UART3 Ch.2 Clock Control Register
	0x0020 0622	UART3_2MOD	UART3 Ch.2 Mode Register
	0x0020 0624	UART3_2BR	UART3 Ch.2 Baud-Rate Register
	0x0020 0626	UART3_2CTL	UART3 Ch.2 Control Register
	0x0020 0628	UART3_2TXD	UART3 Ch.2 Transmit Data Register

Peripheral circuit	Address	Register name	
UART (UART3) Ch.2	0x0020 062a	UART3_2RXD	UART3 Ch.2 Receive Data Register
	0x0020 062c	UART3_2INTF	UART3 Ch.2 Status and Interrupt Flag Register
	0x0020 062e	UART3_2INTE	UART3 Ch.2 Interrupt Enable Register
	0x0020 0630	UART3_2 TBEDMAEN	UART3 Ch.2 Transmit Buffer Empty DMA Request Enable Register
	0x0020 0632	UART3_2 RB1FDMAEN	UART3 Ch.2 Receive Buffer One Byte Full DMA Request Enable Register
	0x0020 0634	UART3_2CAWF	UART3 Ch.2 Carrier Waveform Register
16-bit timer (T16) Ch.6	0x0020 0660	T16_6CLK	T16 Ch.6 Clock Control Register
	0x0020 0662	T16_6MOD	T16 Ch.6 Mode Register
	0x0020 0664	T16_6CTL	T16 Ch.6 Control Register
	0x0020 0666	T16_6TR	T16 Ch.6 Reload Data Register
	0x0020 0668	T16_6TC	T16 Ch.6 Counter Data Register
	0x0020 066a	T16_6INTF	T16 Ch.6 Interrupt Flag Register
	0x0020 066c	T16_6INTE	T16 Ch.6 Interrupt Enable Register
Synchronous serial interface (SPIA) Ch.1	0x0020 0670	SPIA_1MOD	SPIA Ch.1 Mode Register
	0x0020 0672	SPIA_1CTL	SPIA Ch.1 Control Register
	0x0020 0674	SPIA_1TXD	SPIA Ch.1 Transmit Data Register
	0x0020 0676	SPIA_1RXD	SPIA Ch.1 Receive Data Register
	0x0020 0678	SPIA_1INTF	SPIA Ch.1 Interrupt Flag Register
	0x0020 067a	SPIA_1INTE	SPIA Ch.1 Interrupt Enable Register
	0x0020 067c	SPIA_1TBEDMAEN	SPIA Ch.1 Transmit Buffer Empty DMA Request Enable Register
	0x0020 067e	SPIA_1RBFDMAEN	SPIA Ch.1 Receive Buffer Full DMA Request Enable Register
16-bit timer (T16) Ch.2	0x0020 0680	T16_2CLK	T16 Ch.2 Clock Control Register
	0x0020 0682	T16_2MOD	T16 Ch.2 Mode Register
	0x0020 0684	T16_2CTL	T16 Ch.2 Control Register
	0x0020 0686	T16_2TR	T16 Ch.2 Reload Data Register
	0x0020 0688	T16_2TC	T16 Ch.2 Counter Data Register
	0x0020 068a	T16_2INTF	T16 Ch.2 Interrupt Flag Register
	0x0020 068c	T16_2INTE	T16 Ch.2 Interrupt Enable Register
Quad synchronous serial interface (QSPI) Ch.0	0x0020 0690	QSPI_0MOD	QSPI Ch.0 Mode Register
	0x0020 0692	QSPI_0CTL	QSPI Ch.0 Control Register
	0x0020 0694	QSPI_0TXD	QSPI Ch.0 Transmit Data Register
	0x0020 0696	QSPI_0RXD	QSPI Ch.0 Receive Data Register
	0x0020 0698	QSPI_0INTF	QSPI Ch.0 Interrupt Flag Register
	0x0020 069a	QSPI_0INTE	QSPI Ch.0 Interrupt Enable Register
	0x0020 069c	QSPI_0TBEDMAEN	QSPI Ch.0 Transmit Buffer Empty DMA Request Enable Register
	0x0020 069e	QSPI_0RBFDMAEN	QSPI Ch.0 Receive Buffer Full DMA Request Enable Register
	0x0020 06a0	QSPI_0FRLDMAEN	QSPI Ch.0 FIFO Data Ready DMA Request Enable Register
	0x0020 06a2	QSPI_0MMACFG1	QSPI Ch.0 Memory Mapped Access Configuration Register 1
	0x0020 06a4	QSPI_0RMADRH	QSPI Ch.0 Remapping Start Address High Register
	0x0020 06a6	QSPI_0MMACFG2	QSPI Ch.0 Memory Mapped Access Configuration Register 2
	0x0020 06a8	QSPI_0nMB	QSPI Ch.0 Mode Byte Register
I²C (I2C) Ch.1	0x0020 06c0	I2C_1CLK	I2C Ch.1 Clock Control Register
	0x0020 06c2	I2C_1MOD	I2C Ch.1 Mode Register
	0x0020 06c4	I2C_1BR	I2C Ch.1 Baud-Rate Register
	0x0020 06c8	I2C_1OADR	I2C Ch.1 Own Address Register
	0x0020 06ca	I2C_1CTL	I2C Ch.1 Control Register
	0x0020 06cc	I2C_1TXD	I2C Ch.1 Transmit Data Register
	0x0020 06ce	I2C_1RXD	I2C Ch.1 Receive Data Register
	0x0020 06d0	I2C_1INTF	I2C Ch.1 Status and Interrupt Flag Register
	0x0020 06d2	I2C_1INTE	I2C Ch.1 Interrupt Enable Register
	0x0020 06d4	I2C_1TBEDMAEN	I2C Ch.1 Transmit Buffer Empty DMA Request Enable Register
	0x0020 06d6	I2C_1RBFDMAEN	I2C Ch.1 Receive Buffer Full DMA Request Enable Register
I²C (I2C) Ch.2	0x0020 06e0	I2C_2CLK	I2C Ch.2 Clock Control Register
	0x0020 06e2	I2C_2MOD	I2C Ch.2 Mode Register
	0x0020 06e4	I2C_2BR	I2C Ch.2 Baud-Rate Register
	0x0020 06e8	I2C_2OADR	I2C Ch.2 Own Address Register
	0x0020 06ea	I2C_2CTL	I2C Ch.2 Control Register
	0x0020 06ec	I2C_2TXD	I2C Ch.2 Transmit Data Register
	0x0020 06ee	I2C_2RXD	I2C Ch.2 Receive Data Register
	0x0020 06f0	I2C_2INTF	I2C Ch.2 Status and Interrupt Flag Register
	0x0020 06f2	I2C_2INTE	I2C Ch.2 Interrupt Enable Register
	0x0020 06f4	I2C_2TBEDMAEN	I2C Ch.2 Transmit Buffer Empty DMA Request Enable Register
	0x0020 06f6	I2C_2RBFDMAEN	I2C Ch.2 Receive Buffer Full DMA Request Enable Register

Peripheral circuit	Address	Register name	
IR remote controller (REMC3)	0x0020 0720	REMC3CLK	REMC3 Clock Control Register
	0x0020 0722	REMC3DBCTL	REMC3 Data Bit Counter Control Register
	0x0020 0724	REMC3DBCNT	REMC3 Data Bit Counter Register
	0x0020 0726	REMC3APLEN	REMC3 Data Bit Active Pulse Length Register
	0x0020 0728	REMC3DBLEN	REMC3 Data Bit Length Register
	0x0020 072a	REMC3INTF	REMC3 Status and Interrupt Flag Register
	0x0020 072c	REMC3INTE	REMC3 Interrupt Enable Register
	0x0020 0730	REMC3CARR	REMC3 Carrier Waveform Register
	0x0020 0732	REMC3CCTL	REMC3 Carrier Modulation Control Register
16-bit timer (T16) Ch.7	0x0020 0780	T16_7CLK	T16 Ch.7 Clock Control Register
	0x0020 0782	T16_7MOD	T16 Ch.7 Mode Register
	0x0020 0784	T16_7CTL	T16 Ch.7 Control Register
	0x0020 0786	T16_7TR	T16 Ch.7 Reload Data Register
	0x0020 0788	T16_7TC	T16 Ch.7 Counter Data Register
	0x0020 078a	T16_7INTF	T16 Ch.7 Interrupt Flag Register
	0x0020 078c	T16_7INTE	T16 Ch.7 Interrupt Enable Register
12-bit A/D converter (ADC12A) Ch.0	0x0020 07a2	ADC12A_0CTL	ADC12A Ch.0 Control Register
	0x0020 07a4	ADC12A_0TRG	ADC12A Ch.0 Trigger/Analog Input Select Register
	0x0020 07a6	ADC12A_0CFG	ADC12A Ch.0 Configuration Register
	0x0020 07a8	ADC12A_0INTF	ADC12A Ch.0 Interrupt Flag Register
	0x0020 07aa	ADC12A_0INTE	ADC12A Ch.0 Interrupt Enable Register
	0x0020 07ac	ADC12A_0DMAEN0	ADC12A Ch.0 DMA Request Enable Register 0
	0x0020 07ae	ADC12A_0DMAEN1	ADC12A Ch.0 DMA Request Enable Register 1
	0x0020 07b0	ADC12A_0DMAEN2	ADC12A Ch.0 DMA Request Enable Register 2
	0x0020 07b2	ADC12A_0DMAEN3	ADC12A Ch.0 DMA Request Enable Register 3
	0x0020 07b4	ADC12A_0DMAEN4	ADC12A Ch.0 DMA Request Enable Register 4
	0x0020 07b6	ADC12A_0DMAEN5	ADC12A Ch.0 DMA Request Enable Register 5
	0x0020 07b8	ADC12A_0DMAEN6	ADC12A Ch.0 DMA Request Enable Register 6
	0x0020 07ba	ADC12A_0DMAEN7	ADC12A Ch.0 DMA Request Enable Register 7
	0x0020 07bc	ADC12A_0ADD	ADC12A Ch.0 Result Register
Temperature sensor/reference voltage generator (TSRVR)	0x0020 07c0	TSRVR_0TCTL	TSRVR Ch.0 Temperature Sensor Control Register
	0x0020 07c2	TSRVR_0VCTL	TSRVR Ch.0 Reference Voltage Generator Control Register
R/F converter (RFC) Ch.0	0x0020 0840	RFC_0CLK	RFC Ch.0 Clock Control Register
	0x0020 0842	RFC_0CTL	RFC Ch.0 Control Register
	0x0020 0844	RFC_0TRG	RFC Ch.0 Oscillation Trigger Register
	0x0020 0846	RFC_0MCL	RFC Ch.0 Measurement Counter Low Register
	0x0020 0848	RFC_0MCH	RFC Ch.0 Measurement Counter High Register
	0x0020 084a	RFC_0TCL	RFC Ch.0 Time Base Counter Low Register
	0x0020 084c	RFC_0TCH	RFC Ch.0 Time Base Counter High Register
	0x0020 084e	RFC_0INTF	RFC Ch.0 Interrupt Flag Register
	0x0020 0850	RFC_0INTE	RFC Ch.0 Interrupt Enable Register
Sound DAC (SDAC2)	0x0020 0860	SDAC2CLK	SDAC2 Clock Control Register
	0x0020 0862	SDAC2CTL	SDAC2 Control Register
	0x0020 0864	SDAC2MOD	SDAC2 Mode Register
	0x0020 0866	SDAC2_0DAT	SDAC2 Ch.0 Data Register
	0x0020 0868	SDAC2INTF	SDAC2 Interrupt Flag Register
	0x0020 086a	SDAC2INTE	SDAC2 Interrupt Enable Register
	0x0020 0870	SDAC2RESAMP	SDAC2 Resampler Rate Register
	0x0020 0878	SDAC2TONE	SDAC2 Tone Divider Register
	0x0020 087e	SDAC2_1DAT	SDAC2 Ch.1 Data Register
HW processor (HWP)	0x0020 08a2	HWPCTL	HWP Control Register
	0x0020 08a4	HWPINTF	HWP Interrupt Flag Register
	0x0020 08a6	HWPINTE	HWP Interrupt Enable Register
	0x0020 08a8	HWPCMDTRG	HWP Command Trigger Register
DMA controller (DMAC)	0x0020 1000	DMACSTAT	DMAC Status Register
	0x0020 1004	DMACCFG	DMAC Configuration Register
	0x0020 1008	DMACCPTR	DMAC Control Data Base Pointer Register
	0x0020 100c	DMACACPTR	DMAC Alternate Control Data Base Pointer Register
	0x0020 1014	DMACSWREQ	DMAC Software Request Register
	0x0020 1020	DMACRMSET	DMAC Request Mask Set Register
	0x0020 1024	DMACRMCLR	DMAC Request Mask Clear Register
	0x0020 1028	DMACENSET	DMAC Enable Set Register
	0x0020 102c	DMACENCLR	DMAC Enable Clear Register
	0x0020 1030	DMACPASET	DMAC Primary-Alternate Set Register
	0x0020 1034	DMACPACLR	DMAC Primary-Alternate Clear Register
	0x0020 1038	DMACPRSET	DMAC Priority Set Register

Peripheral circuit	Address	Register name	
DMA controller (DMAC)	0x0020 103c	DMACPRCLR	DMAC Priority Clear Register
	0x0020 104c	DMACERRIF	DMAC Error Interrupt Flag Register
	0x0020 2000	DMACENDIF	DMAC Transfer Completion Interrupt Flag Register
	0x0020 2008	DMACENDIESET	DMAC Transfer Completion Interrupt Enable Set Register
	0x0020 200c	DMACENDIECLR	DMAC Transfer Completion Interrupt Enable Clear Register
	0x0020 2010	DMACERRIESET	DMAC Error Interrupt Enable Set Register
	0x0020 2014	DMACERRIECLR	DMAC Error Interrupt Enable Clear Register

4.5.1 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 SYSPROT.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 SYSPROT.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.

4.6 Instruction Cache

This IC includes an instruction cache. Enabling the cache function translates into reduced current consumption, as the Flash memory access frequency is decreased.
This function is enabled by setting the CACHECTL.CACHEEN bit to 1. Setting this bit to 0 clears the instruction codes stored in the cache.
Before placing the IC into SLEEP or HALT mode, disable the cache function by setting the CACHECTL.CACHEEN bit to 0.

4.7 Memory Mapped Access Area For External Flash Memory

This area is used to read data from the external Flash memory via the quad synchronous serial interface. For more information, refer to the “Quad Synchronous Serial Interface” chapter.

4.8 Control Registers

System Protect Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SYSPROT	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).

CACHE Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
CACHECTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	–	1	–	R	
	0	CACHEEN	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 CACHEEN

This bit enables the instruction cache function.

1 (R/W): Enable instruction cache

0 (R/W): Disable instruction cache

FLASHC Flash Read Cycle Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
FLASHCWAIT	15–9	–	0x00	–	R	–
	8	(reserved)	0	H0	R/WP	
	7–2	–	0x00	–	R	
	1–0	RDWAIT[1:0]	0x1	H0	R/WP	

Bits 15–2 Reserved

Bits 1–0 RDWAIT[1:0]

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

Table 4.8.1 Setting Number of Bus Access Cycles for Flash Read

FLASHCWAIT. RDWAIT[1:0] bits	Number of bus access cycles	System clock frequency	
		PWGACTL. REGSEL bit = 0	PWGACTL. REGSEL bit = 1
0x3	4	2.1 MHz (max.)	16.3 MHz (max.)
0x2	3		
0x1	2		
0x0	1	1.02 MHz (max.)	8.4 MHz (max.)

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

• When the FLASHCWAIT.RDWAIT[1:0] bit setting is altered from 0x2 to 0x1, add two NOP instructions immediately after that.

Program example:
```
FLASHC->WAIT_b.RDWAIT = 1;
asm("NOP");
asm("NOP");
CLG->OSC_b.IOSCEN = 0;
```

5 Interrupt

5.1 Overview

This IC includes a nested vectored interrupt controller (NVIC). For detailed information on the NVIC, refer to the documents introduced in Section 3.4, such as “ARM®v6-M Architecture Reference Manual.”
Figure 5.1.1 shows the configuration of the interrupt system.

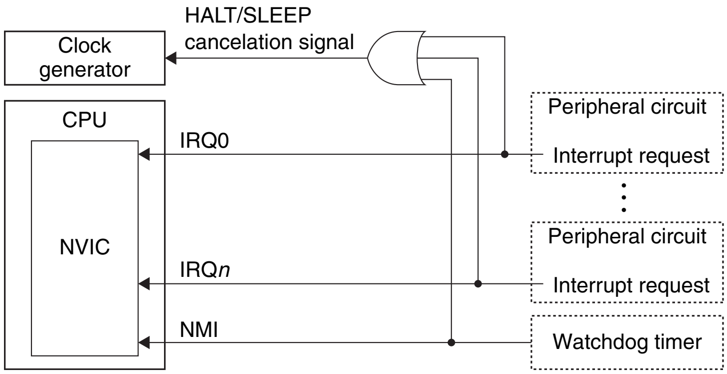


Figure 5.1.1 Configuration of Interrupt System

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

VTOR initial value = 0x0

Interrupt number	IRQ number	Vector address	Hardware interrupt name	Cause of hardware interrupt	Priority
–	–	VTOR + 0x00	(Stack pointer initial value)	–	–
1	–	VTOR + 0x04	Reset	• Low input to the #RESET pin • Power-on reset • Key reset • Watchdog timer overflow *1 • Supply voltage detector reset	-3
2	-14	VTOR + 0x08	NMI	Watchdog timer overflow *1	-2
3	-13	VTOR + 0x0c	HardFault	• Bus error • Undefined instruction • Unaligned address etc.	-1
4–10	–	–	Reserved	–	–
11	-5	VTOR + 0x2c	SVCall	SVC instruction	Configurable
12–13	–	-	Reserved	–	–
14	-2	VTOR + 0x38	PendSV	–	
15	-1	VTOR + 0x3c	SysTick	SysTick timer underflow	
16	0	VTOR + 0x40	DMA controller interrupt	• DMA transfer completion • DMA transfer error	
17	1	VTOR + 0x44	Supply voltage detector interrupt	Power supply voltage drop detection	
18	2	VTOR + 0x48	Port interrupt (P0/P1)	P0–P1 port input	
19	3	VTOR + 0x4c	Port interrupt (P2/P3)	P2–P3 port input	Configurable
20	4	VTOR + 0x50	Port interrupt (Others)	P4–P6 port input	
21	5	VTOR + 0x54	Clock generator interrupt	• IOSC oscillation stabilization waiting completion • OSC1 oscillation stabilization waiting completion • OSC3 oscillation stabilization waiting completion • OSC1 oscillation stop • OSC3 oscillation auto-trimming completion • OSC3 oscillation auto-trimming error	

Interrupt number	IRQ number	Vector address	Hardware interrupt name	Cause of hardware interrupt	Priority
22	6	VTOR + 0x58	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	Configurable
23	7	VTOR + 0x5c	HW processor	• Error occurrence • State transition	
24	8	VTOR + 0x60	Sound DAC	• Error occurrence • Data request	
25	9	VTOR + 0x64	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	
26	10	VTOR + 0x68	16-bit timer Ch.1 interrupt	Underflow	
27	11	VTOR + 0x6c	Synchronous serial interface Ch.0 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty • Overrun error	
28	12	VTOR + 0x70	I^2C Ch.0 interrupt	• End of data transfer • General call address reception • NACK reception • STOP condition • START condition • Error detection • Receive buffer full • Transmit buffer empty	
29	13	VTOR + 0x74	16-bit PWM timer Ch.0 interrupt	• Capture overwrite • Compare/capture • Counter MAX • Counter zero	
30	14	VTOR + 0x78	16-bit PWM timer Ch.1 interrupt	• Capture overwrite • Compare/capture • Counter MAX • Counter zero	
31	15	VTOR + 0x7c	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	
32	16	VTOR + 0x80	16-bit timer Ch.2 interrupt	Underflow	
33	17	VTOR + 0x84	Quad synchronous serial interface Ch.0 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty • Overrun error	
34	18	VTOR + 0x88	I^2C Ch.1 interrupt	• End of data transfer • General call address reception • NACK reception • STOP condition • START condition • Error detection • Receive buffer full • Transmit buffer empty	
35	19	VTOR + 0x8c	UART Ch.2 interrupt	• End of transmission • Framing error • Parity error • Overrun error • Receive buffer two bytes full • Receive buffer one byte full • Transmit buffer empty	
			16-bit timer Ch.0 interrupt	Underflow	
36	20	VTOR + 0x90	16-bit timer Ch.3 interrupt	Underflow	

Interrupt number	IRQ number	Vector address	Hardware interrupt name	Cause of hardware interrupt	Priority
37	21	VTOR + 0x90	Synchronous serial interface Ch.1 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty • Overrun error	Configurable
38	22	VTOR + 0x98	16-bit timer Ch.4 interrupt	Underflow	
39	23	VTOR + 0x9c	16-bit timer Ch.5 interrupt	Underflow	
40	24	VTOR + 0xa0	16-bit timer Ch.6 interrupt	Underflow	
41	25	VTOR + 0xa4	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	
42	26	VTOR + 0xa8	12-bit A/D converter interrupt	• Analog input signal *m* A/D conversion completion • Analog input signal *m* A/D conversion result overwrite error	
43	27	VTOR + 0xac	Synchronous serial interface Ch.2 interrupt	• End of transmission • Receive buffer full • Transmit buffer empty • Overrun error	
			16-bit timer Ch.7 interrupt	Underflow	
44	28	VTOR + 0xb0	I²C Ch.2 interrupt	• End of data transfer • General call address reception • NACK reception • STOP condition • START condition • Error detection • Receive buffer full • Transmit buffer empty	
45	29	VTOR + 0xb4	IR remote controller interrupt	• Compare AP • Compare DB	
46–47	–	–	Reserved	–	

∗1 Either reset or NMI can be selected as the watchdog timer interrupt via software.

5.2.1 Vector Table Offset Address (VTOR)

The Cortex®-M0+ Vector Table Offset Register (VTOR) is provided to set the offset (start) address of the vector table in which interrupt vectors are programmed. "VTOR" described in Table 5.2.1 means the value set to this register. After an initial reset, VTOR is set to address 0x0. Therefore, even when the vector table location is changed, it is necessary that at least the reset vector be written to this address. For more information on VTOR, refer to the documents introduced in Section 3.4, such as "Cortex®-M0+ Devices Generic User Guide."

5.2.2 Priority of Interrupts

The priorities of SVCall, PendSV, and SysTick are configurable to the desired levels using the Cortex®-M0+ System Handler Priority Registers (SHPR2 and SHPR3). The priorities of the interrupt number 16 or later are configurable to the desired levels using the Cortex®-M0+ Interrupt Priority Registers (NVIC_IPR0–7). The priority value can be set within a range of 0 to 192 (a lower value has a higher priority). The priorities of reset, NMI, and HardFault are fixed at the predefined values. For more information, refer to the documents introduced in Section 3.4, such as "Cortex®-M0+ Devices Generic User Guide."

5.3 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 CPU 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 CPU even if the interrupt flag is set to 1. An interrupt request is also sent to the CPU 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 NMI

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.

6 DMA Controller (DMAC)

6.1 Overview

The main features of the DMAC are outlined below.

- Supports byte, halfword, and word transfers.
- Each DMAC channel can be configured to different transfer conditions independently.
- Supports memory-to-memory, memory-to-peripheral circuit, and peripheral circuit-to-memory transfers.
- Supports hardware DMA requests from peripheral circuits and software DMA requests.
- Priority level for each channel is selectable from two levels.
- DMA transfers are allowed even if the CPU is placed into HALT mode.

Figure 6.1.1 shows the configuration of the DMAC.

Table 6.1.1 DMAC Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	4 channels (Ch.0 to Ch.3)		
Transfer source memories	Internal Flash memory, external Flash memory, and RAM		
Transfer destination memories	RAM		
Transfer source peripheral circuits	UART3, SPIA, QSPI, I2C, T16B, and ADC12A		
Transfer destination peripheral circuits	UART3, SPIA, QSPI, I2C, T16B		

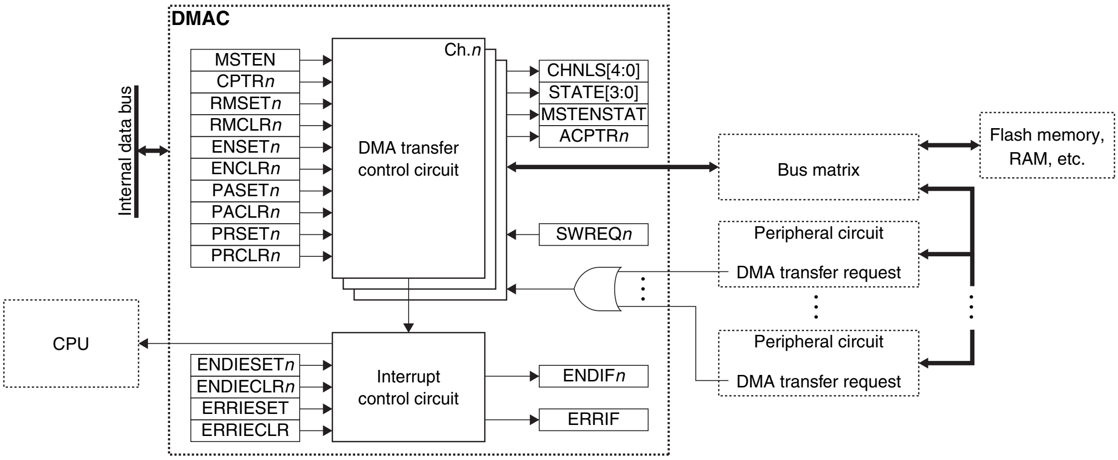


Figure 6.1.1 DMAC Configuration

6.2 Operations

6.2.1 Initialization

The DMAC should be initialized with the procedure shown below.

1. Set the data structure base address to the DMACCPTR register.
2. Configure the data structure for the channels to be used.
   - Set the control data.
   - Set the transfer source end pointer.
   - Set the transfer destination end pointer.
3. Set the DMACCFG.MSTEN bit to 1. (Enable DMAC)
4. Configure the DMACRMSET and DMACRMCLR registers.
   (Configure masks for DMA transfer requests from peripheral circuits)
5. Configure the DMACENSET and DMACENCLR registers. (Enable channels used)
6. Configure the DMACPASET and DMACPACLR registers. (Select data structure used)
7. Configure the DMACPRSET and DMACPRCLR registers. (Set priorities)
8 Set the following registers when using the interrupt:
   - Write 1 to the interrupt flags in the DMACENDIF and DMACERRIF registers. (Clear interrupt flags)
   - Configures the DMACENDIESET/DMACENDIECLR and DMACERRIESET/DMACERRIECLR registers. (Enable/disable interrupts)
9. Set the DMA request enable bits of the peripheral circuits that use DMA transfer to 1.
10. To issue a software DMA request to Ch.*n*, write 1 to the DMACSWREQ.SWREQ*n* bit.

6.3 Priority

If DMA requests are issued to two or more channels, the DMA transfers are performed in order from the highest-priority channel. The channel of which the priority level is set to 1 by the DMACPRSET.PRSET*n* bit has the highest priority. If two or more channels have been set to the same priority level, the smaller channel number takes precedence.

6.4 Data Structure

To perform DMA transfers, a data structure that contains basic transfer control information must be provided. The data structure consists of two blocks, primary data structure and alternate data structure, and one of them is used according to the DMA transfer mode.
The data structure can be located at an arbitrary address in the RAM area by setting the base address to the DMACCPTR.CPTR[31:0] bits.
The data structure for each channel consists of a transfer source end pointer, a transfer destination end pointer, and control data. An area of 16 bytes × 2 is allocated in the RAM for each channel.
The whole size of the data structure and the alternate data structure base address depend on the number of channels implemented.

Table 6.4.1 Data Structure Size According to Number of Channels Implemented

Number of channels implemented	Data structure size	Primary data structure base address	Alternate data structure base address
1	32 bytes	DMACCPTR.CPTR[31:0] (CPTR[4:0] = 0x00)	DMACCPTR.CPTR[31:0] + 0x010
2	64 bytes	DMACCPTR.CPTR[31:0] (CPTR[5:0] = 0x00)	DMACCPTR.CPTR[31:0] + 0x020
3 to 4	128 bytes	DMACCPTR.CPTR[31:0] (CPTR[6:0] = 0x00)	DMACCPTR.CPTR[31:0] + 0x040
5 to 8	256 bytes	DMACCPTR.CPTR[31:0] (CPTR[7:0] = 0x00)	DMACCPTR.CPTR[31:0] + 0x080
9 to 16	512 bytes	DMACCPTR.CPTR[31:0] (CPTR[8:0] = 0x000)	DMACCPTR.CPTR[31:0] + 0x100
17 to 32	1,024 bytes	DMACCPTR.CPTR[31:0] (CPTR[9:0] = 0x000)	DMACCPTR.CPTR[31:0] + 0x200

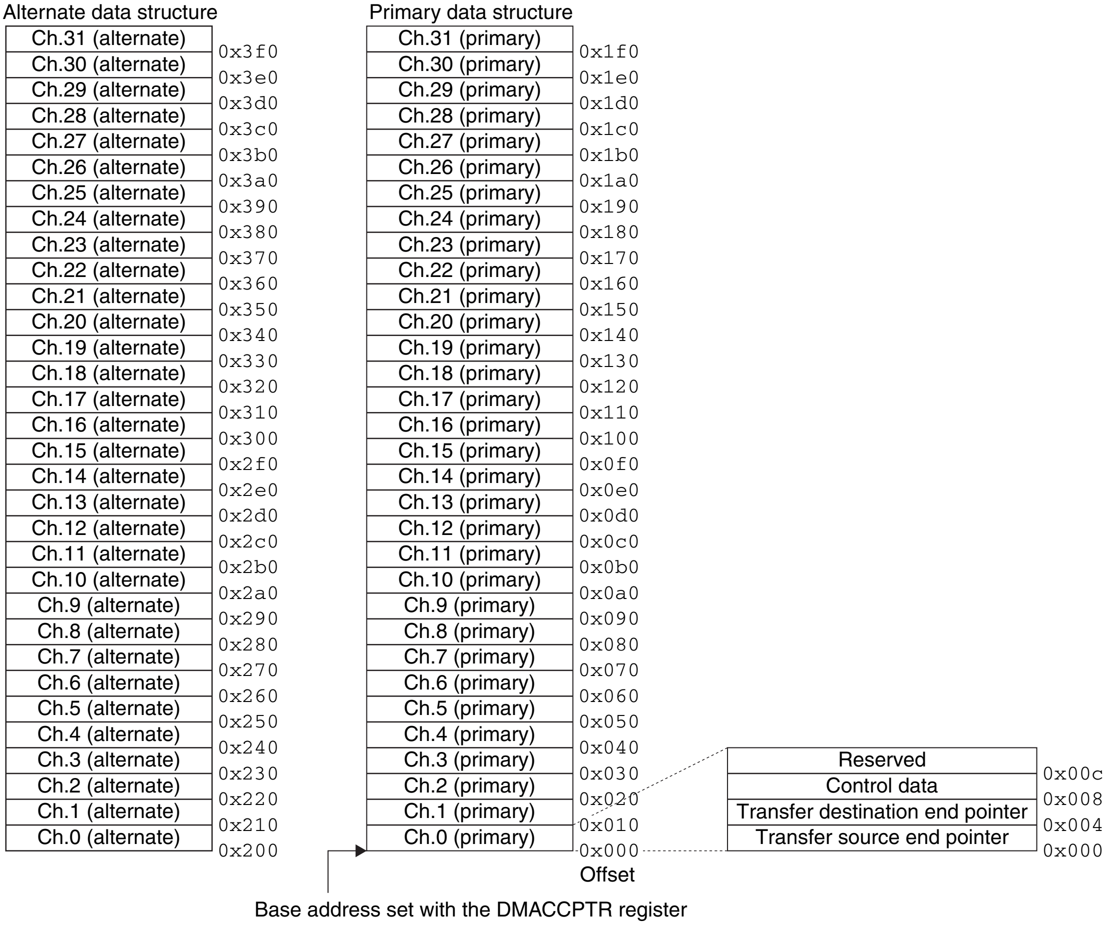


Figure 6.4.1 Data Structure Address Map (when 32 channels are implemented)

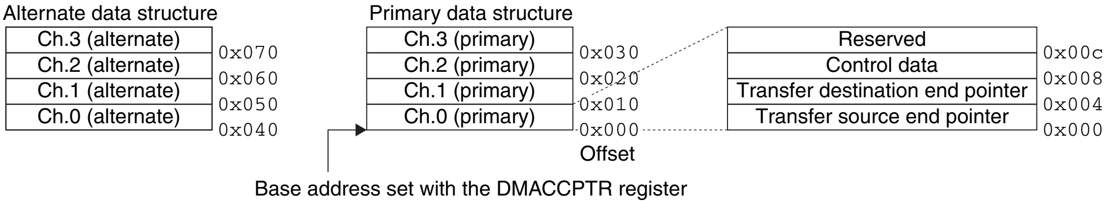


Figure 6.4.2 Data Structure Address Map (when 4 channels are implemented)

The alternate data structure base address can be determined from the DMACACPTR.ACPTR[31:0] bits.

6.4.1 Transfer Source End Pointer

Set the source data end address. The address of data to be transferred should be set as it is if the transfer source address is not incremented.

6.4.2 Transfer Destination End Pointer

Set the address to which the last transfer data is written. The address for writing transfer data should be set as it is if the transfer destination address is not incremented.

6.4.3 Control Data

Set the DMA transfer information. Figure 6.4.3.1 shows the constituent elements of the control data.

31–30	29–28	27–26	25–24	23–18	17–14	13–4	3	2–0
dst_inc	dst_size	src_inc	src_size	(Reserved)	R_power	n_minus_1	(Reserved)	cycle_ctrl

Reserved

Figure 6.4.3.1 Constituent Elements of Control Data

dst_inc

Set the increment value of the transfer destination address. The setting value must be equal to or larger than the transfer data size when the address is incremented.

Table 6.4.3.1 Increment Value of Transfer Destination Address

dst_inc	Increment value
0x3	No increment
0x2	+4
0x1	+2
0x0	+1

dst_size

Set the size of the data to be written to the transfer destination. It should be the same value as the src_size.

Table 6.4.3.2 Size of Data Written to Transfer Destination

dst_size	Data size
0x3	Reserved
0x2	Word
0x1	Halfword
0x0	Byte

src_inc

Set the increment value of the transfer source address. The setting value must be equal to or larger than the transfer data size when the address is incremented.

Table 6.4.3.3 Increment Value of Transfer Source Address

src_inc	Increment value
0x3	No increment
0x2	+4
0x1	+2
0x0	+1

src_size

Set the size of the data to be read from the transfer source. It should be the same value as the dst_size.

Table 6.4.3.4 Size of Data Read from Transfer Source

src_size	Data size
0x3	Reserved
0x2	Word
0x1	Halfword
0x0	Byte

R_power

Set the arbitration cycle during successive data transfer.

$$\text{Arbitration cycle } (2^{R}) = 2^{R\_power}$$

When the DMAC is performing a successive transfer, it suspends the data transfer at the cycle set with R_power. If DMA requests have been issued at that point, the DMAC re-arbitrates them according to their priorities and then performs a DMA transfer for the channel with the highest priority.
If the arbitration cycle setting value is larger than the number of successive data transfers, successive data transfers will not be suspended.

n_minus_1

Set the number of DMA transfers to be executed successively.

Number of successive transfers (N) = n_minus_1 + 1

When the set number of successive transfers has completed, a transfer completion interrupt occurs.

cycle_ctrl

Set the DMA transfer mode. For detailed information on each transfer mode, refer to Section 6.5, "DMA Transfer Mode."

Table 6.4.3.5 DMA Transfer Mode

cycle_ctrl	DMA transfer mode
0x7	Peripheral scatter-gather transfer (for alternate data structure)
0x6	Peripheral scatter-gather transfer (for primary data structure)
0x5	Memory scatter-gather transfer (for alternate data structure)
0x4	Memory scatter-gather transfer (for primary data structure)
0x3	Ping-pong transfer
0x2	Auto-request transfer
0x1	Basic transfer
0x0	Stop

6.5 DMA Transfer Mode

6.5.1 Basic Transfer

This is the basic DMA transfer mode. In this mode, DMA transfer starts when a DMA transfer request from a peripheral circuit or a software DMA request is issued, and it continues until it is completed for the set number of successive transfers or it is suspended at the arbitration cycle. To resume the DMA transfer suspended at the arbitration cycle, a DMA transfer request must be reissued.
When the set number of successive transfers has completed, a transfer completion interrupt occurs.

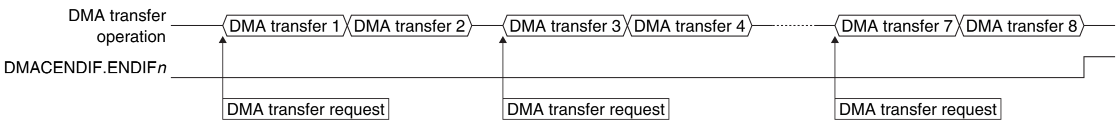


Figure 6.5.1.1 Basic Transfer Operation Example (N = 8, 2^R = 2)

6.5.2 Auto-Request Transfer

Similar to the basic transfer, DMA transfer starts when a DMA transfer request from a peripheral circuit or a software DMA request is issued, and it continues until it is completed for the set number of successive transfers or it is suspended at the arbitration cycle. The DMAC resumes the DMA transfer suspended at the arbitration cycle without a DMA transfer request being reissued.
When the set number of successive transfers has completed, a transfer completion interrupt occurs.

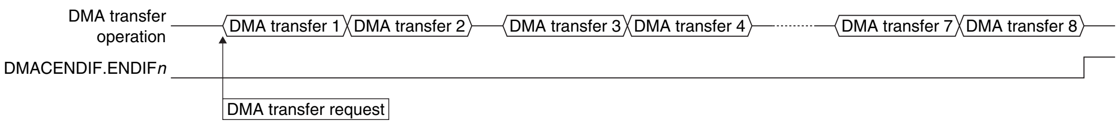


Figure 6.5.2.1 Auto-Request Transfer Operation Example (N = 8, 2^R = 2)

6.5.3 Ping-Pong Transfer

In ping-pong transfer mode, the DMAC performs basic transfers repeatedly while switching between the primary data structure and alternate data structure. The data structures are referred alternately, and DMA transfer is terminated when the control data with cycle_ctrl set to 0x0 is referred. A transfer completion interrupt occurs each time a transfer using a data structure is completed.

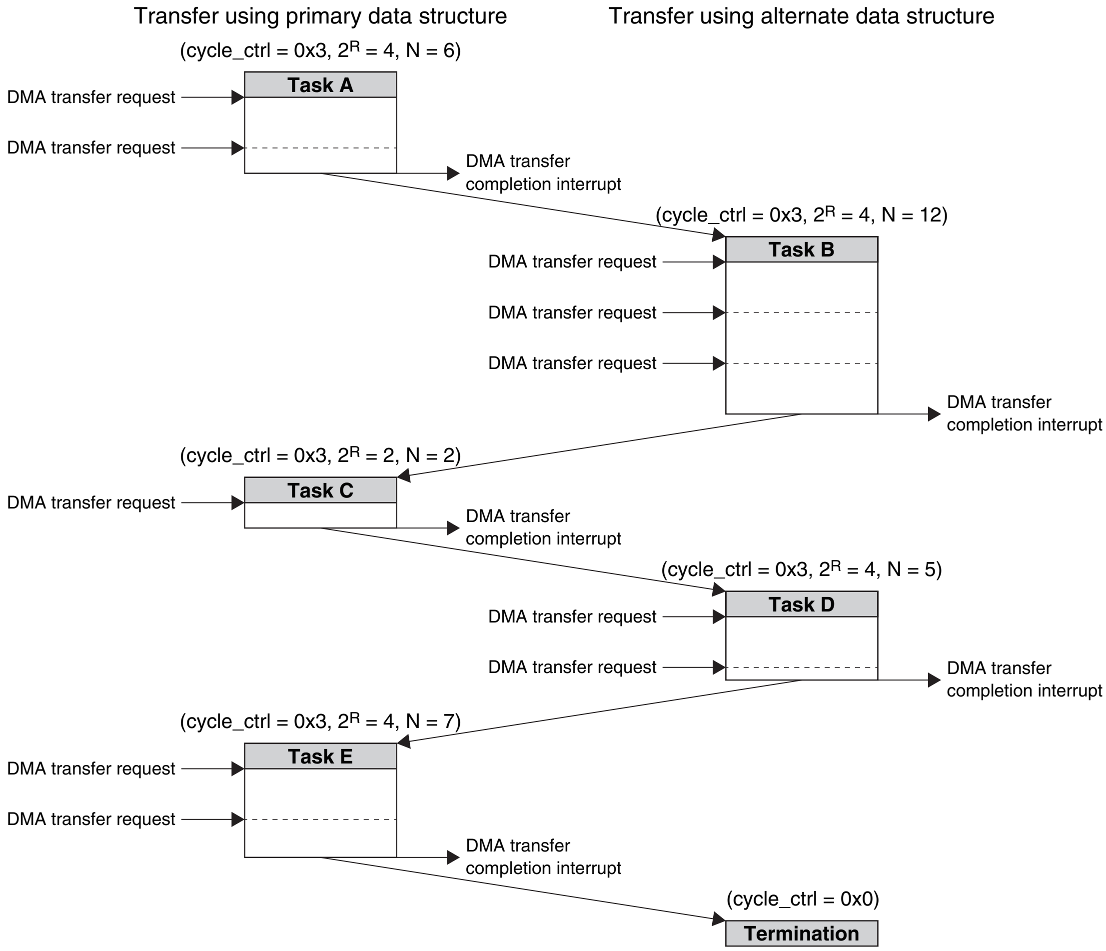


Figure 6.5.3.1 Ping-Pong Transfer Operation Example

DMA transfer procedure

1. Start data transfer by following the procedure shown in Section 6.2.1, "Initialization." In Step 2 of the initialization procedure, set Task A and Task B to the primary data structure and the alternate data structure, respectively.
2. Set Task C to the primary data structure after a DMA transfer completion interrupt has occurred by Task A.
3. Set Task D to the alternate data structure when a DMA transfer completion interrupt has occurred by Task B.
4. Repeat Steps 2 and 3.
5. Set cycle_ctrl to 0x0 after a DMA transfer completion interrupt has occurred by the next to last task.
6. The DMA transfer is completed when a DMA transfer completion interrupt occurs by the last task.

6.5.4 Memory Scatter-Gather Transfer

In scatter-gather transfer mode, first the DMAC, using the primary data structure, copies a data structure from the data structure table, which has been prepared with multiple data structures included in advance, to the alternate data structure, and then it performs DMA transfer using the alternate data structure. The DMAC performs this operation repeatedly. By programming the transfer mode of the data structure located at the end of the table as a basic transfer, the DMA transfer can be terminated with a transfer completion interrupt. This mode requires a DMA transfer request only for starting the first data transfer. Subsequent data transfers are performed by auto-requests.

Reserved	Data structure for Task D
Control data	
Transfer destination end pointer	
Transfer source end pointer	
Reserved	Data structure for Task C
Control data	
Transfer destination end pointer	
Transfer source end pointer	
Reserved	Data structure for Task B
Control data	
Transfer destination end pointer	
Transfer source end pointer	
Reserved	Data structure for Task A
Control data	
Transfer destination end pointer	
Transfer source end pointer	

Figure 6.5.4.1 Example of Data Structure Table for Scatter-Gather Transfer

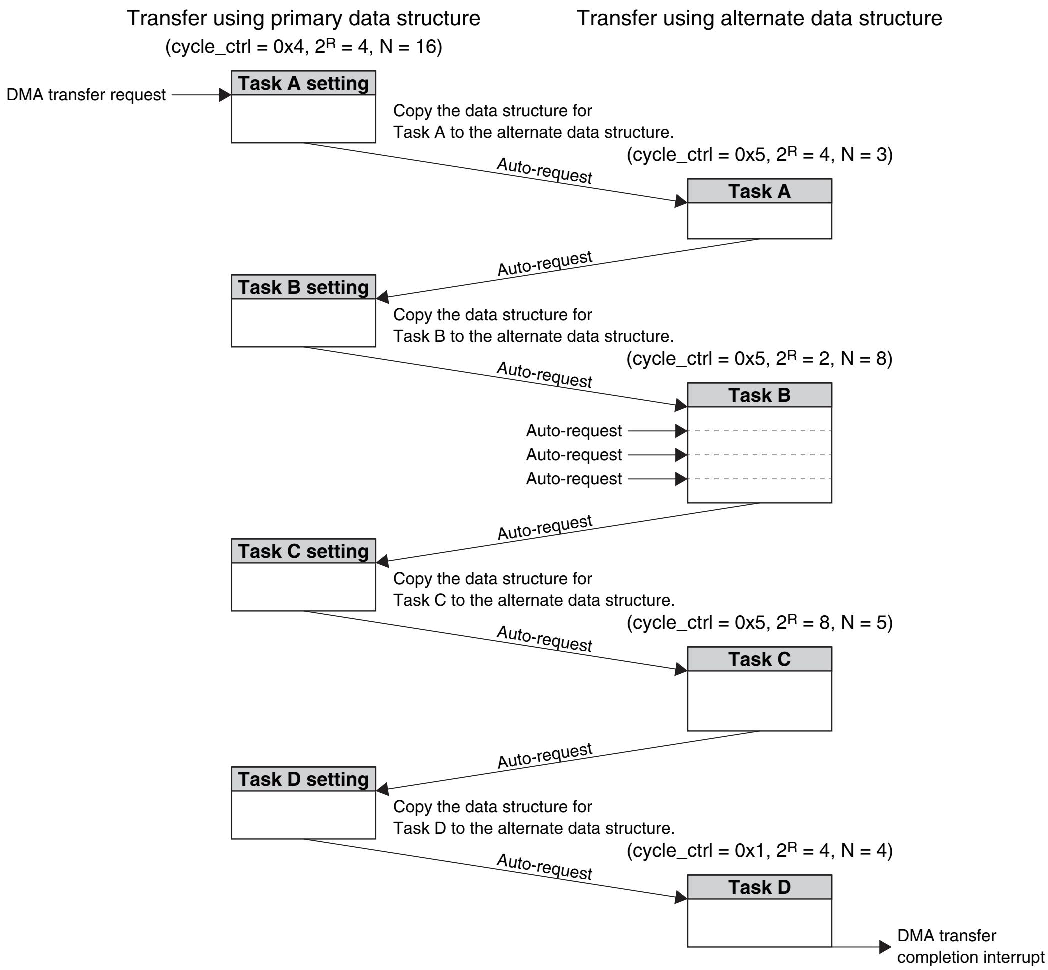


Figure 6.5.4.2 Memory Scatter-Gather Transfer Operation Example

DMA transfer procedure

1. Configure the data structure table for scatter-gather transfer.
   Set the cycle_ctrl for the last task to 0x1 and those for other tasks to 0x5.
2. Start data transfer by following the procedure shown in Section 6.2.1, "Initialization." In Step 2 of the initialization procedure, configure the primary data structure with the control data shown below.
   Transfer source end pointer = Data structure table end address
   Transfer destination end pointer = Alternate data structure end address
   dst_inc = 0x2
   dst_size = 0x2
   src_inc = 0x2
   src_size = 0x2
   R_power = 0x2
   n_minus_1 = Number of tasks × 4 - 1
   cycle_ctrl = 0x4
3. The DMA transfer is completed when a DMA transfer completion interrupt occurs.

6.5.5 Peripheral Scatter-Gather Transfer

In memory scatter-gather transfer mode, the second and subsequent DMA transfers are performed by auto-requests. On the other hand, in peripheral scatter-gather transfer mode, all DMA transfers are performed by a DMA transfer request issued by a peripheral circuit or a software DMA request.

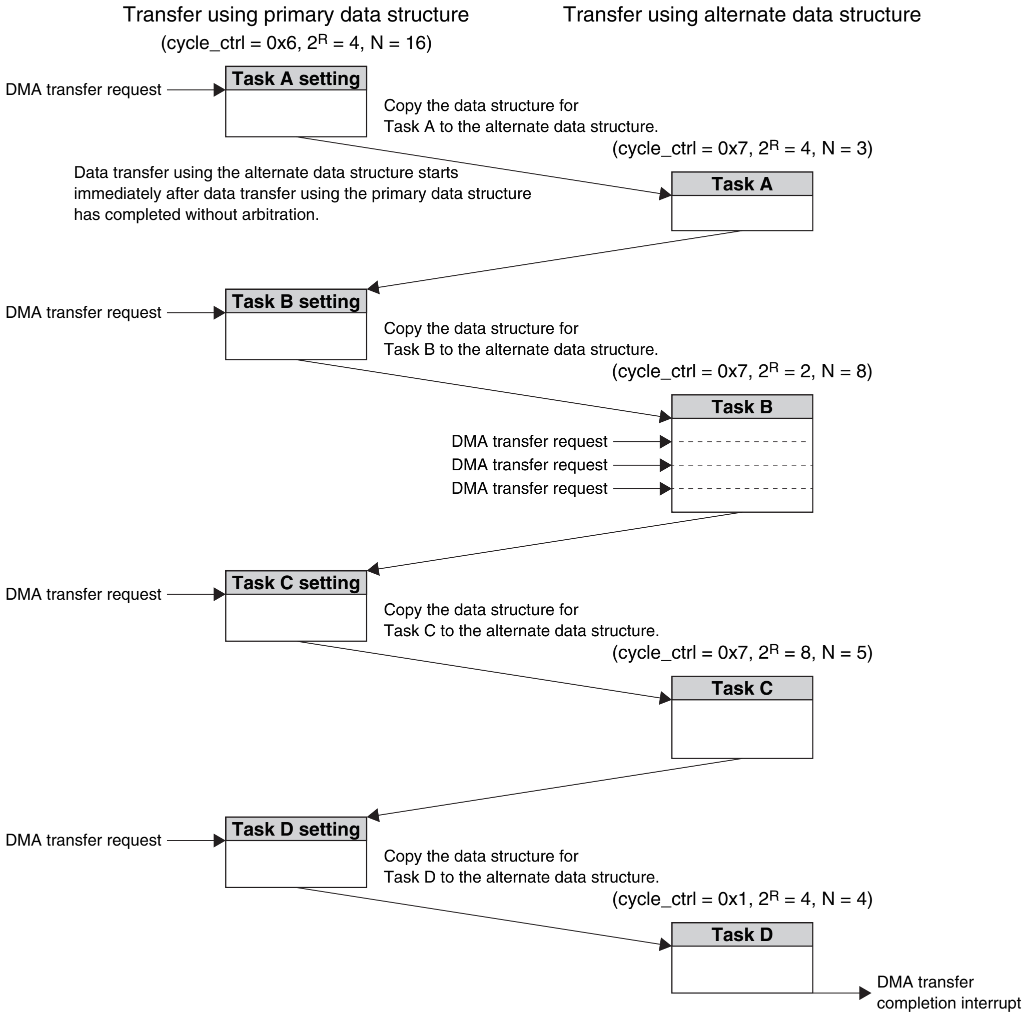


Figure 6.5.5.1 Peripheral Scatter-Gather Transfer Operation Example

DMA transfer procedure

1. Configure the data structure table for scatter-gather transfer.
   Set the cycle_ctrl for the last task to 0x1 and those for other tasks to 0x7.
2. Start data transfer by following the procedure shown in Section 6.2.1, "Initialization." In Step 2 of the initialization procedure, configure the primary data structure with the control data shown below.
   Transfer source end pointer = Data structure table end address
   Transfer destination end pointer = Alternate data structure end address
   dst_inc = 0x2
   dst_size = 0x2
   src_inc = 0x2
   src_size = 0x2
   R_power = 0x2
   n_minus_1 = Number of tasks × 4 - 1
   cycle_ctrl = 0x6
3. Issue a DMA transfer request in each task using a peripheral circuit or via software.
4. The DMA transfer is completed when a DMA transfer completion interrupt occurs.

6.6 DMA Transfer Cycle

A DMA transfer requires several clock cycles to execute. Figure 6.6.1 shows a detailed DAM transfer cycle. Note that the number of clock cycles for a DMA transfer may be increased due to a conflict with an access from the CPU or the Flash bus access cycle setting.

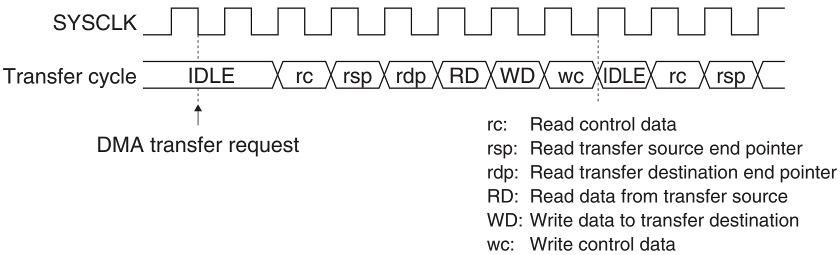


Figure 6.6.1 DMA Transfer Cycle

6.7 Interrupts

The DMAC has a function to generate the interrupts shown in Table 6.7.1.

Table 6.7.1 DMAC Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
DMA transfer completion	DMACENDIF.ENDIF*n*	When DMA transfers for a set number of successive transfers have completed	Writing 1
DMA transfer error	DMACERRIF.ERRIF	When an AHB bus error has occurred	Writing 1

The DMAC provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU 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" chapter.

6.8 Control Registers

DMAC Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACSTAT	31–24	–	0x00	–	R	–
	23–21	–	0x0	–	R	
	20–16	CHNLS[4:0]	*	H0	R	* Number of channels implemented - 1
	15–8	–	0x00	–	R	–
	7–4	STATE[3:0]	0x0	H0	R	
	3–1	–	0x0	–	R	
	0	MSTENSTAT	0	H0	R	

Bits 31–21 Reserved

Bits 20–16 CHNLS[4:0]

These bits show the number of DMAC channels implemented in this IC.

Number of channels implemented = CHNLS + 1

Bits 15–8 Reserved

Bits 7–4 STATE[3:0]

These bits indicates the DMA transfer status.

Table 6.8.1 DMA Transfer Status

DMACSTAT.STATE[3:0] bits	DMA transfer status
0xf–0xbf	Reserved
0xa	Peripheral scatter-gather transfer is in progress.
0x9	Transfer has completed.
0x8	Transfer has been suspended.
0x7	Control data is being written.
0x6	Standby for transfer request to be cleared.
0x5	Transfer data is being written.
0x4	Transfer data is being read.
0x3	Transfer destination end pointer is being read.
0x2	Transfer source end pointer is being read.
0x1	Control data is being read.
0x0	Idle

Bits 3–1 Reserved

Bit 0 MSTENSTAT

This bit indicates the DMA controller status.

1 (R): DMA controller is operating.

0 (R): DMA controller is idle.

DMAC Configuration Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACCFG	31–24	–	0x00	–	R	–
	23–16	–	0x00	–	R	
	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	MSTEN	–	–	W	

Bits 31–1 Reserved

Bit 0 MSTEN

This bit enables the DMA controller.

1 (W): Enable

0 (W): Disable

DMAC Control Data Base Pointer Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACCPTR	31–0	CPTR[31:0]	0x0000 0000	H0	R/W	–

Bits 31–0 CPTR[31:0]

These bits set the leading address of the data structure.

Depending on the number of channels implemented, low-order bits are configured for read only.

Table 6.8.2 CPTR Writable/Read-Only Bits Depending On Number of Channel Implemented

Number of channel implemented	Writable bits	Read-only bits
1	CPTR[31:5]	CPTR[4:0]
2	CPTR[31:6]	CPTR[5:0]
3–4	CPTR[31:7]	CPTR[6:0]
5–8	CPTR[31:8]	CPTR[7:0]
9–16	CPTR[31:9]	CPTR[8:0]
17–32	CPTR[31:10]	CPTR[9:0]

DMAC Alternate Control Data Base Pointer Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACACPTR	31–0	ACPTR[31:0]	–	H0	R	–

Bits 31–0 ACPTR[31:0]

These bits show the alternate data structure base address.

DMAC Software Request Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACSWREQ	31–0	SWREQ[31:0]	–	–	W	–

Bits 31–0 SWREQ [31:0]

These bits issue a software DMA transfer request to each channel.

1 (W): Issue a software DMA transfer request
0 (W): Ineffective

Each bit corresponds to a DMAC channel (e.g. bit *n* corresponds to Ch.*n*). The high-order bits for the unimplemented channels are ineffective.

DMAC Request Mask Set Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACRMSET	31–0	RMSET[31:0]	0x0000 0000	H0	R/W	–

Bits 31–0 RMSET[31:0]

These bits mask DMA transfer requests from peripheral circuits.

1 (W): Mask DMA transfer requests from peripheral circuits
0 (W): Ineffective
1 (R): DMA transfer requests from peripheral circuits have been disabled.
0 (R): DMA transfer requests from peripheral circuits have been enabled.

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Request Mask Clear Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACRMCLR	31–0	RMCLR[31:0]	–	–	W	–

Bits 31–0 RMCLR[31:0]

These bits cancel the mask state of DMA transfer requests from peripheral circuits

1 (W): Cancel mask state of DMA transfer requests from peripheral circuits (The DMACRMSET register is cleared to 0.)

0 (W): Ineffective

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Enable Set Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACENSET	31–0	ENSET[31:0]	0x0000 0000	H0	R/W	–

Bits 31–0 ENSET[31:0]

These bits enable each DMAC channel.

1 (W): Enable DMAC channel

0 (W): Ineffective

1 (R): Enabled

0 (R): Disabled

These bits are cleared after the DMA transfer has completed.

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Enable Clear Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACENCLR	31–0	ENCLR[31:0]	–	–	W	–

Bits 31–0 ENCLR[31:0]

These bits disable each DMAC channel.

1 (W): Disable DMAC channel (The DMACENSET register is cleared to 0.)

0 (W): Ineffective

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Primary-Alternate Set Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACPASET	31–0	PASET[31:0]	0x0000 0000	H0	R/W	–

Bits 31–0 PASET[31:0]

These bits enable the alternate data structures.

1 (W): Enable alternate data structure

0 (W): Ineffective

1 (R): The alternate data structure has been enabled.

0 (R): The primary data structure has been enabled.

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Primary-Alternate Clear Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACPACLR	31–0	PACLR[31:0]	–	–	W	–

Bits 31–0 PACLR[31:0]

These bits disable the alternate data structures.

1 (W): Disable alternate data structure (The DMACPASET register is cleared to 0.)
0 (W): Ineffective

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Priority Set Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACPRSET	31–0	PRSET[31:0]	0x0000 0000	H0	R/W	–

Bits 31–0 PRSET[31:0]

These bits increase the priority of each channel.

1 (W): Increase priority
0 (W): Ineffective
1 (R): Priority = High
0 (R): Priority = Normal

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Priority Clear Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACPRCLR	31–0	PRCLR[31:0]	–	–	W	–

Bits 31–0 PRCLR[31:0]

These bits decrease the priority of each channel.

1(W): Decrease priority (The DMACPRSET register is cleared to 0.)
0 (W): Ineffective

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Error Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACERRIF	31–24	–	0x00	–	R	–
	23–16	–	0x00	–	R	
	15–8	–	0x00	–	R	
	7–1	–	0x00	–	R	
	0	ERRIF	0	H0	R/W	Cleared by writing 1.

Bits 31–1 Reserved

Bit 0 ERRIF

This bit indicates the DMAC error interrupt cause occurrence status.

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

DMAC Transfer Completion Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACENDIF	31–0	ENDIF[31:0]	0x0000 0000	H0	R/W	Cleared by writing 1.

Bits 31–0 ENDIF[31:0]

These bits indicate the DMA transfer completion interrupt cause occurrence status of each DMAC channel.

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

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Transfer Completion Interrupt Enable Set Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACENDIESET	31–0	ENDIESET[31:0]	0x0000 0000	H0	R/W	–

Bits 31–0 ENDIESET[31:0]

These bits enable DMA transfer completion interrupts to be generated from each DMAC channel.

1 (W): Enable interrupt
0 (W): Ineffective
1 (R): Interrupt has been enabled.
0 (R): Interrupt has been disabled.

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Transfer Completion Interrupt Enable Clear Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACENDIECLR	31–0	ENDIECLR[31:0]	–	–	W	–

Bits 31–0 ENDIECLR[31:0]

These bits disable DMA transfer completion interrupts to be generated from each DMAC channel.

1 (W): Disable interrupt (The DMACENDIESET register is cleared to 0.)
0 (W): Ineffective

Each bit corresponds to a DMAC channel. The high-order bits for the unimplemented channels are ineffective.

DMAC Error Interrupt Enable Set Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACERRIESET	31–24	–	0x00	–	R	–
	23–16	–	0x00	–	R	
	15–8	–	0x00	–	R	
	7–1	–	0x00	–	R	
	0	ERRIESET	0	H0	R/W	

Bits 31–1 Reserved

Bit 0 ERRIESET

This bit enables DMA error interrupts.

1 (W): Enable interrupt
0 (W): Ineffective
1 (R): Interrupt has been enabled.
0 (R): Interrupt has been disabled.

DMAC Error Interrupt Enable Clear Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
DMACERRIECLR	31–24	–	0x00	–	R	–
	23–16	–	0x00	–	R	
	15–8	–	0x00	–	R	
	7–1	–	0x00	–	R	
	0	ERRIECLR	–	–	W	

Bits 31–1 Reserved

Bit 0 ERRIECLR

This bit disables DMA error interrupts.

1 (W): Disable interrupt (The DMACERRIESET register is cleared to 0.)
0 (W): Ineffective

7 I/O Ports (PPORT)

7.1 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.)

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 7.1.1 shows the configuration of PPORT.

Table 7.1.1 Port Configuration of S1C31D41

Item		32-pin package		48-pin package		64-pin package	
Port groups included	P0	–	(0)	P0[6:2]	(5) *1, *2	P0[7:0]	(8) *1, *2
	P1	P1[7:2]	(6) *1, *2	P1[7:2]	(6) *1, *2	P1[7:0]	(8) *1, *2
	P2	P20	(1) *1, *2	P2[3:0]	(4) *1, *2	P2[7:0]	(8) *1, *2
	P3	P3[2:1]	(2) *1, *2	P3[5:1]	(5) *1, *2	P3[7:0]	(8) *1, *2
	P4	P4[4:2]	(3) *1, *2	P4[5:2]	(4) *1, *2	P4[5:0]	(6) *1, *2
	P5	P54, P5[1:0]	(3) *1, *2	5[6:4], P5[1:0]	(5) *1, *2	P5[6:0]	(7) *1, *2
	P6	P6[5:0]	(6) *1, *2	P6[5:0]	(6) *1, *2	P6[5:0]	(6) *1, *2
	Pd	Pd[3:0]	(4) *1	Pd[3:0]	(4) *1	Pd[3:0]	(4) *1
Total number of ports		25 ports		39 ports		55 ports	
Ports for debug function		Pd[1:0]					
Key-entry reset function		Unavailable					

*1 Ports with general-purpose I/O function (GPIO)

*2 Ports with interrupt function

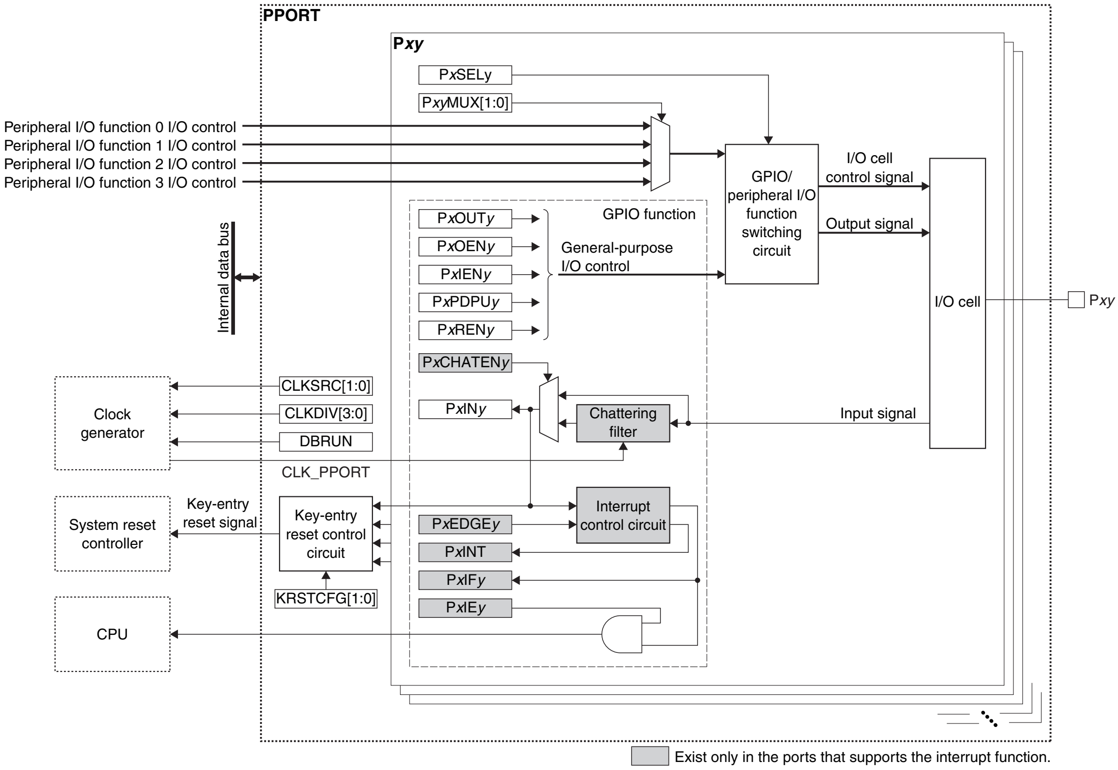


Figure 7.1.1 PPORT Configuration

7.2 I/O Cell Structure and Functions

Figure 7.2.1 shows the I/O cell Configuration.

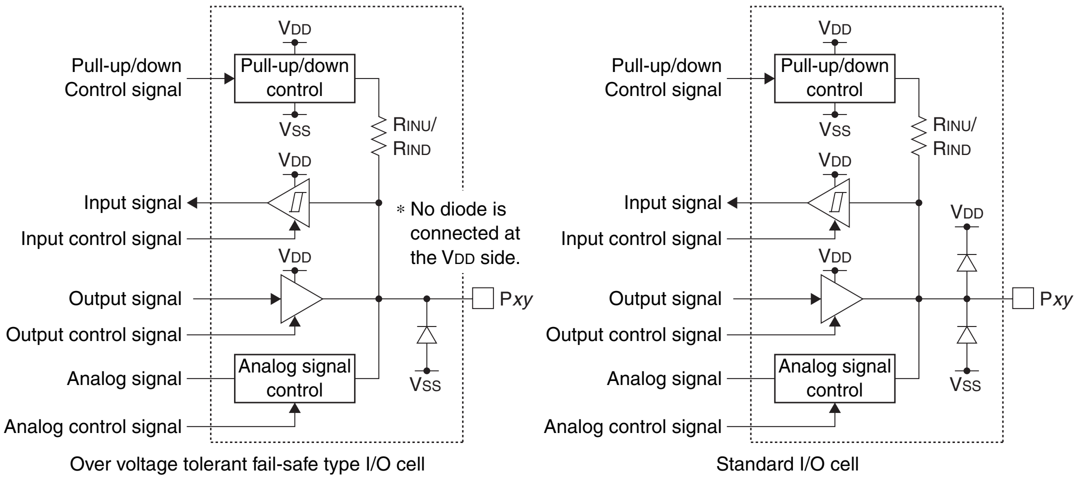


Figure 7.2.1 I/O Cell Configuration

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.

7.2.1 Schmitt Input

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

7.2.2 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.

7.2.3 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:

$$t_{PR} = -R_{INU} \times (C_{IN} + C_{BOARD}) \times \ln(1 - V_{T+}/V_{DD}) \qquad \text{(Eq. 7.1)}$$
$$t_{PF} = -R_{IND} \times (C_{IN} + C_{BOARD}) \times \ln(1 - V_{T-}/V_{DD})$$

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]

7.2.4 CMOS Output and High Impedance State

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

7.3 Clock Settings

7.3.1 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 SYSPROT.PROT[15:0] bits. (Remove system protection)
3. Set the following PPORTCLK register bits:
   - PPORTCLK.CLKSRC[1:0] bits (Clock source selection)
   - PPORTCLK.CLKDIV[3:0] bits (Clock division ratio selection = Clock frequency setting)
4. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

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

7.3.2 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 PPORTP*x*CHATEN.P*x*CHATEN*y* bit setting (chattering filter enabled/disabled).

7.3.3 Clock Supply During Debugging

The CLK_PPORT supply during debugging should be controlled using the PPORTCLK.DBRUN bit.
The CLK_PPORT supply to PPORT is suspended when the CPU enters debug state if the PPORTCLK.DBRUN bit = 0. After the CPU returns to normal operation, 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 PPORTCLK.DBRUN bit = 1, the CLK_PPORT supply is not suspended and the chattering filter will keep operating in a debug state.

7.4 Operations

7.4.1 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 PPORTP*x*IOEN register bits:
   - Set the PPORTP*x*IOEN.P*x*IEN*y* bit to 0. (Disable input)
   - Set the PPORTP*x*IOEN.P*x*OEN*y* bit to 0. (Disable output)
2. Set the PPORTP*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 PPORTP*x*FNCSEL.P*xy*MUX[1:0] bits. (Select peripheral I/O function)
5. Set the PPORTP*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 PPORTP*x*IOEN.P*x*OEN*y* bit to 1. (Enable output)
2. Set the PPORTP*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 PPORTP*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 PPORTP*x*CHATEN.P*x*CHATEN*y* bit to 1. *

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

   Set the PPORTP*x*RCTL.P*x*REN*y* bit to 0 if the internal pull-up/down resistors are not used.
4. Set the PPORTP*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 PPORTP*x*INTF.P*x*IF*y* bit. (Clear interrupt flag)
   - PPORTP*x*INTCTL.P*x*EDGE*y* bit (Select interrupt edge (input rising edge/falling edge))
   - Set the PPORTP*x*INTCTL.P*x*IE*y* bit to 1. (Enable interrupt)
6. Set the following PPORTP*x*IOEN register bits:
   - Set the PPORTP*x*IOEN.P*x*OEN*y* bit to 0. (Disable output)
   - Set the PPORTP*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 7.4.1.1 lists the port status according to the combination of data input/output control and pull-up/down control.

Table 7.4.1.1 GPIO Port Control List

PPORTP*x*IOEN. P*x*IEN*y* bit	PPORTP*x*IOEN. P*x*OEN*y* bit	PPORTP*x*RCTL. P*x*REN*y* bit	PPORTP*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 PPORTP*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.

7.4.2 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 PPORTP*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 PPORTP*x*DAT.P*x*IN*y* bit.

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 PPORTP*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 PPORTCLK register in common to all ports. The chattering filter removes pulses with a shorter width than the input sampling time.

$$\text{Input sampling time} = \frac{2 \text{ to } 3}{\text{CLK\_PPORT frequency [Hz]}} \text{ [second]} \qquad \text{(Eq. 7.2)}$$

Make sure the P*xy* port interrupt is disabled before altering the PPORTCLK register and PPORTP*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 PPORTCLK.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 PPORTCLK.KRSTCFG[1:0] bits.

PPORT issues a reset request immediately after all the input pins specified by the PPORTCLK.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.

7.5 Interrupts

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

Table 7.5.1 Port Input Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Port input interrupt	PPORTP*x*INTF.P*x*IF*y*	Rising or falling edge of the input signal	Writing 1
	PPORTINTFGRP.P*x*INT	Setting an interrupt flag in the port group	Clearing PPORTP*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 PPORTP*x*INTCTL. P*x*EDGE*y* bit to 1, or the rising edge when setting to 0.

Interrupt enable

PPORT provides interrupt enable bits (PPORTP*x*INTCTL.P*x*IE*y* bit) corresponding to each interrupt flag. An interrupt request is sent to the CPU 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" chapter.

Interrupt check in port group unit

When interrupts are enabled in two or more port groups, check the PPORTINTFGRP.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 PPORTP*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 PPORTP*x*INTF.P*x*IF*y* bit also clears the PPORTINTFGRP.P*x*INT bit. If the port is set to interrupt disabled status by the PPORTP*x*INTCTL. P*x*IE*y* bit, the PPORTINTFGRP.P*x*INT bit will not be set even if the PPORTP*x*INTF.P*x*IF*y* bit is set to 1.

7.6 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
PPORTP*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 (PPORTP*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 (PPORTP*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 (PPORTP*x*IOEN.P*x*IEN*y* bit = 1). When input is disabled (PPORTP*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
PPORTP*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
PPORTP*x*RCTL	15–8	P*x*PDPU[7:0]	0x00	H0	R/W	–
	7–0	PxREN[7:0]	0x00	H0	R/W	

*1: This register is effective when the GPIO function is selected.

*2: The bit configuration differs depending on the port group.

Bits 15–8 P*x*PDPU[7:0]

These bits select either the pull-up resistor or the pull-down resistor when using a resistor built into the port.

1 (R/W): Pull-up resistor

0 (R/W): Pull-down resistor

The selected pull-up/down resistor is enabled when the PPORTP*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 (PPORTP*x*IOEN.P*x*OEN*y* bit = 0). When output is enabled (PPORTP*x*IOEN.P*x*OEN*y* bit = 1), the PPORTP*x*RCTL.P*x*REN*y* bit setting is ineffective regardless of how the PPORTP*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
PPORTP*x*INTF	15–8	–	0x00	–	R	–
	7–0	PxIF[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
PPORTP*x*INTCTL	15–8	P*x*EDGE[7:0]	0x00	H0	R/W	–
	7–0	PxIE[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
PPORTP*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
PPORTP*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
PPORTP*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 7.6.1 Selecting Peripheral I/O Function

PPORTP*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 PPORTP*x*MODSEL.P*x*SEL*y* bit = 1.

P Port Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PPORTCLK	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 during debugging or not.

1 (R/WP): Clock supplied during debugging

0 (R/WP): No clock supplied during debugging

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 7.6.2 Key-Entry Reset Function Settings

PPORTCLK.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 PPORTCLK.CLKSRC[1:0] bits and the clock division ratio using the PPORTCLK.CLKDIV[3:0] bits as shown in Table 7.6.3. These settings determine the input sampling time of the chattering filter.

Table 7.6.3 Clock Source and Division Ratio Settings

PPORTCLK.CLKDIV[3:0] bits	PPORTCLK.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
PPORTINTFGRP	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 PPORTINTFGRP.P*x*INT bit is cleared when the interrupt flag for the port that has generated an interrupt is cleared.

7.7 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.

7.7.1 P0 Port Group

The P0 port group supports the GPIO and interrupt functions.

Table 7.7.1.1 Control Registers for P0 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P0DAT (P0 Port Data Register)	15	P0OUT7	0	H0	R/W	–	–	–	✓
	14	P0OUT6	0	H0	R/W		–	✓	✓
	13	P0OUT5	0	H0	R/W		–	✓	✓
	12	P0OUT4	0	H0	R/W		–	✓	✓
	11	P0OUT3	0	H0	R/W		–	✓	✓
	10	P0OUT2	0	H0	R/W		–	✓	✓
	9	P0OUT1	0	H0	R/W		–	–	✓
	8	P0OUT0	0	H0	R/W		–	–	✓
	7	P0IN7	0	H0	R	–	–	–	✓
	6	P0IN6	0	H0	R		–	✓	✓
	5	P0IN5	0	H0	R		–	✓	✓
	4	P0IN4	0	H0	R		–	✓	✓
	3	P0IN3	0	H0	R		–	✓	✓
	2	P0IN2	0	H0	R		–	✓	✓
	1	P0IN1	0	H0	R		–	–	✓
	0	P0IN0	0	H0	R		–	–	✓
P0IOEN (P0 Port Enable Register)	15	P0IEN7	0	H0	R/W	–	–	–	✓
	14	P0IEN6	0	H0	R/W		–	✓	✓
	13	P0IEN5	0	H0	R/W		–	✓	✓
	12	P0IEN4	0	H0	R/W		–	✓	✓
	11	P0IEN3	0	H0	R/W		–	✓	✓
	10	P0IEN2	0	H0	R/W		–	✓	✓
	9	P0IEN1	0	H0	R/W		–	–	✓
	8	P0IEN0	0	H0	R/W		–	–	✓
	7	P0OEN7	0	H0	R/W	–	–	–	✓
	6	P0OEN6	0	H0	R/W		–	✓	✓
	5	P0OEN5	0	H0	R/W		–	✓	✓
	4	P0OEN4	0	H0	R/W		–	✓	✓
	3	P0OEN3	0	H0	R/W		–	✓	✓
	2	P0OEN2	0	H0	R/W		–	✓	✓
	1	P0OEN1	0	H0	R/W		–	–	✓
	0	P0OEN0	0	H0	R/W		–	–	✓
P0RCTL (P0 Port Pull-up/down Control Register)	15	P0PDPU7	0	H0	R/W	–	–	–	✓
	14	P0PDPU6	0	H0	R/W		–	✓	✓
	13	P0PDPU5	0	H0	R/W		–	✓	✓
	12	P0PDPU4	0	H0	R/W		–	✓	✓
	11	P0PDPU3	0	H0	R/W		–	✓	✓
	10	P0PDPU2	0	H0	R/W		–	✓	✓
	9	P0PDPU1	0	H0	R/W		–	–	✓
	8	P0PDPU0	0	H0	R/W		–	–	✓
	7	P0REN7	0	H0	R/W	–	–	–	✓
	6	P0REN6	0	H0	R/W		–	✓	✓
	5	P0REN5	0	H0	R/W		–	✓	✓
	4	P0REN4	0	H0	R/W		–	✓	✓
	3	P0REN3	0	H0	R/W		–	✓	✓
	2	P0REN2	0	H0	R/W		–	✓	✓
	1	P0REN1	0	H0	R/W		–	–	✓
	0	P0REN0	0	H0	R/W		–	–	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P0INTF (P0 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P0IF7	0	H0	R/W	Cleared by writing 1.	–	–	✓
	6	P0IF6	0	H0	R/W		–	✓	✓
	5	P0IF5	0	H0	R/W		–	✓	✓
	4	P0IF4	0	H0	R/W		–	✓	✓
	3	P0IF3	0	H0	R/W		–	✓	✓
	2	P0IF2	0	H0	R/W		–	✓	✓
	1	P0IF1	0	H0	R/W		–	–	✓
	0	P0IF0	0	H0	R/W		–	–	✓
P0INTCTL (P0 Port Interrupt Control Register)	15	P0EDGE7	0	H0	R/W	–	–	–	✓
	14	P0EDGE6	0	H0	R/W		–	✓	✓
	13	P0EDGE5	0	H0	R/W		–	✓	✓
	12	P0EDGE4	0	H0	R/W		–	✓	✓
	11	P0EDGE3	0	H0	R/W		–	✓	✓
	10	P0EDGE2	0	H0	R/W		–	✓	✓
	9	P0EDGE1	0	H0	R/W		–	–	✓
	8	P0EDGE0	0	H0	R/W		–	–	✓
	7	P0IE7	0	H0	R/W	–	–	–	✓
	6	P0IE6	0	H0	R/W		–	✓	✓
	5	P0IE5	0	H0	R/W		–	✓	✓
	4	P0IE4	0	H0	R/W		–	✓	✓
	3	P0IE3	0	H0	R/W		–	✓	✓
	2	P0IE2	0	H0	R/W		–	✓	✓
	1	P0IE1	0	H0	R/W		–	–	✓
	0	P0IE0	0	H0	R/W		–	–	✓
P0CHATEN (P0 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P0CHATEN7	0	H0	R/W	–	–	–	✓
	6	P0CHATEN6	0	H0	R/W		–	✓	✓
	5	P0CHATEN5	0	H0	R/W		–	✓	✓
	4	P0CHATEN4	0	H0	R/W		–	✓	✓
	3	P0CHATEN3	0	H0	R/W		–	✓	✓
	2	P0CHATEN2	0	H0	R/W		–	✓	✓
	1	P0CHATEN1	0	H0	R/W		–	–	✓
	0	P0CHATEN0	0	H0	R/W		–	–	✓
P0MODSEL (P0 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P0SEL7	0	H0	R/W	–	–	–	✓
	6	P0SEL6	0	H0	R/W		–	✓	✓
	5	P0SEL5	0	H0	R/W		–	✓	✓
	4	P0SEL4	0	H0	R/W		–	✓	✓
	3	P0SEL3	0	H0	R/W		–	✓	✓
	2	P0SEL2	0	H0	R/W		–	✓	✓
	1	P0SEL1	0	H0	R/W		–	–	✓
	0	P0SEL0	0	H0	R/W		–	–	✓
P0FNCSEL (P0 Port Function Select Register)	15–14	P07MUX[1:0]	0x0	H0	R/W	–	–	–	✓
	13–12	P06MUX[1:0]	0x0	H0	R/W		–	✓	✓
	11–10	P05MUX[1:0]	0x0	H0	R/W		–	✓	✓
	9–8	P04MUX[1:0]	0x0	H0	R/W		–	✓	✓
	7–6	P03MUX[1:0]	0x0	H0	R/W		–	✓	✓
	5–4	P02MUX[1:0]	0x0	H0	R/W		–	✓	✓
	3–2	P01MUX[1:0]	0x0	H0	R/W		–	–	✓
	1–0	P00MUX[1:0]	0x0	H0	R/W		–	–	✓

Table 7.7.1.2 P0 Port Group Function Assignment

Port name	P0SEL*y* = 0	P0SEL*y* = 1								32 pin	48 pin	64 pin
	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	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P01	P01	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P02	P02	CLG	FOUT	UPMUX	*1	–	–	–	–	–	✓	✓
P03	P03	SDAC2	SDACOUT_P2	UPMUX	*1	–	–	–	–	–	✓	✓
P04	P04	SDAC2	SDACOUT_P	UPMUX	*1	–	–	–	–	–	✓	✓
P05	P05	SDAC2	SDACOUT_N	UPMUX	*1	–	–	–	–	–	✓	✓
P06	P06	SDAC2	SDACOUT_N2	UPMUX	*1	–	–	–	–	–	✓	✓
P07	P07	–	–	UPMUX	*1	–	–	–	–	–	–	✓

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

7.7.2 P1 Port Group

The P1 port group supports the GPIO and interrupt functions.

Table 7.7.2.1 Control Registers for P1 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P1DAT (P1 Port Data Register)	15	P1OUT7	0	H0	R/W	–	✓	✓	✓
	14	P1OUT6	0	H0	R/W		✓	✓	✓
	13	P1OUT5	0	H0	R/W		✓	✓	✓
	12	P1OUT4	0	H0	R/W		✓	✓	✓
	11	P1OUT3	0	H0	R/W		✓	✓	✓
	10	P1OUT2	0	H0	R/W		✓	✓	✓
	9	P1OUT1	0	H0	R/W		–	–	✓
	8	P1OUT0	0	H0	R/W		–	–	✓
	7	P1IN7	0	H0	R	–	✓	✓	✓
	6	P1IN6	0	H0	R		✓	✓	✓
	5	P1IN5	0	H0	R		✓	✓	✓
	4	P1IN4	0	H0	R		✓	✓	✓
	3	P1IN3	0	H0	R		✓	✓	✓
	2	P1IN2	0	H0	R		✓	✓	✓
	1	P1IN1	0	H0	R		–	–	✓
	0	P1IN0	0	H0	R		–	–	✓
P1IOEN (P1 Port Enable Register)	15	P1IEN7	0	H0	R/W	–	✓	✓	✓
	14	P1IEN6	0	H0	R/W		✓	✓	✓
	13	P1IEN5	0	H0	R/W		✓	✓	✓
	12	P1IEN4	0	H0	R/W		✓	✓	✓
	11	P1IEN3	0	H0	R/W		✓	✓	✓
	10	P1IEN2	0	H0	R/W		✓	✓	✓
	9	P1IEN1	0	H0	R/W		–	–	✓
	8	P1IEN0	0	H0	R/W		–	–	✓
	7	P1OEN7	0	H0	R/W	–	✓	✓	✓
	6	P1OEN6	0	H0	R/W		✓	✓	✓
	5	P1OEN5	0	H0	R/W		✓	✓	✓
	4	P1OEN4	0	H0	R/W		✓	✓	✓
	3	P1OEN3	0	H0	R/W		✓	✓	✓
	2	P1OEN2	0	H0	R/W		✓	✓	✓
	1	P1OEN1	0	H0	R/W		–	–	✓
	0	P1OEN0	0	H0	R/W		–	–	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P1RCTL (P1 Port Pull-up/down Control Register)	15	P1PDPU7	0	H0	R/W	–	✓	✓	✓
	14	P1PDPU6	0	H0	R/W		✓	✓	✓
	13	P1PDPU5	0	H0	R/W		✓	✓	✓
	12	P1PDPU4	0	H0	R/W		✓	✓	✓
	11	P1PDPU3	0	H0	R/W		✓	✓	✓
	10	P1PDPU2	0	H0	R/W		✓	✓	✓
	9	P1PDPU1	0	H0	R/W		–	–	✓
	8	P1PDPU0	0	H0	R/W		–	–	✓
	7	P1REN7	0	H0	R/W	–	✓	✓	✓
	6	P1REN6	0	H0	R/W		✓	✓	✓
	5	P1REN5	0	H0	R/W		✓	✓	✓
	4	P1REN4	0	H0	R/W		✓	✓	✓
	3	P1REN3	0	H0	R/W		✓	✓	✓
	2	P1REN2	0	H0	R/W		✓	✓	✓
	1	P1REN1	0	H0	R/W		–	–	✓
	0	P1REN0	0	H0	R/W		–	–	✓
P1INTF (P1 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P1IF7	0	H0	R/W	Cleared by writing 1.	✓	✓	✓
	6	P1IF6	0	H0	R/W		✓	✓	✓
	5	P1IF5	0	H0	R/W		✓	✓	✓
	4	P1IF4	0	H0	R/W		✓	✓	✓
	3	P1IF3	0	H0	R/W		✓	✓	✓
	2	P1IF2	0	H0	R/W		✓	✓	✓
	1	P1IF1	0	H0	R/W		–	–	✓
	0	P1IF0	0	H0	R/W		–	–	✓
P1INTCTL (P1 Port Interrupt Control Register)	15	P1EDGE7	0	H0	R/W	–	✓	✓	✓
	14	P1EDGE6	0	H0	R/W		✓	✓	✓
	13	P1EDGE5	0	H0	R/W		✓	✓	✓
	12	P1EDGE4	0	H0	R/W		✓	✓	✓
	11	P1EDGE3	0	H0	R/W		✓	✓	✓
	10	P1EDGE2	0	H0	R/W		✓	✓	✓
	9	P1EDGE1	0	H0	R/W		–	–	✓
	8	P1EDGE0	0	H0	R/W		–	–	✓
	7	P1IE7	0	H0	R/W	–	✓	✓	✓
	6	P1IE6	0	H0	R/W		✓	✓	✓
	5	P1IE5	0	H0	R/W		✓	✓	✓
	4	P1IE4	0	H0	R/W		✓	✓	✓
	3	P1IE3	0	H0	R/W		✓	✓	✓
	2	P1IE2	0	H0	R/W		✓	✓	✓
	1	P1IE1	0	H0	R/W		–	–	✓
	0	P1IE0	0	H0	R/W		–	–	✓
P1CHATEN (P1 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P1CHATEN7	0	H0	R/W	–	✓	✓	✓
	6	P1CHATEN6	0	H0	R/W		✓	✓	✓
	5	P1CHATEN5	0	H0	R/W		✓	✓	✓
	4	P1CHATEN4	0	H0	R/W		✓	✓	✓
	3	P1CHATEN3	0	H0	R/W		✓	✓	✓
	2	P1CHATEN2	0	H0	R/W		✓	✓	✓
	1	P1CHATEN1	0	H0	R/W		–	–	✓
	0	P1CHATEN0	0	H0	R/W		–	–	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P1MODSEL (P1 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P1SEL7	0	H0	R/W	–	✓	✓	✓
	6	P1SEL6	0	H0	R/W		✓	✓	✓
	5	P1SEL5	0	H0	R/W		✓	✓	✓
	4	P1SEL4	0	H0	R/W		✓	✓	✓
	3	P1SEL3	0	H0	R/W		✓	✓	✓
	2	P1SEL2	0	H0	R/W		✓	✓	✓
	1	P1SEL1	0	H0	R/W		–	–	✓
	0	P1SEL0	0	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 7.7.2.2 P1 Port Group Function Assignment

Port name	P1SEL*y* = 0	P1SEL*y* = 1								32 pin	48 pin	64 pin
	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	–	–	UPMUX	*1	ADC12A	ADIN06	–	–	–	–	✓
P11	P11	–	–	UPMUX	*1	ADC12A	ADIN05	–	–	–	–	✓
P12	P12	–	–	UPMUX	*1	ADC12A	ADIN04	–	–	✓	✓	✓
P13	P13	CLG	FOUT	UPMUX	*1	ADC12A	ADIN03	–	–	✓	✓	✓
P14	P14	–	–	UPMUX	*1	ADC12A	ADIN02	–	–	✓	✓	✓
P15	P15	–	–	UPMUX	*1	ADC12A	ADIN01	–	–	✓	✓	✓
P16	P16	–	–	UPMUX	*1	ADC12A	ADIN00	–	–	✓	✓	✓
P17	P17	–	–	UPMUX	*1	ADC12A/ TSRVR	VREFA0	–	–	✓	✓	✓

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

7.7.3 P2 Port Group

The P2 port group supports the GPIO and interrupt functions.

Table 7.7.3.1 Control Registers for P2 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P2DAT (P2 Port Data Register)	15	P2OUT7	0	H0	R/W	–	–	–	✓
	14	P2OUT6	0	H0	R/W		–	–	✓
	13	P2OUT5	0	H0	R/W		–	–	✓
	12	P2OUT4	0	H0	R/W		–	–	✓
	11	P2OUT3	0	H0	R/W		–	✓	✓
	10	P2OUT2	0	H0	R/W		–	✓	✓
	9	P2OUT1	0	H0	R/W		–	✓	✓
	8	P2OUT0	0	H0	R/W		✓	✓	✓
	7	P2IN7	0	H0	R	–	–	–	✓
	6	P2IN6	0	H0	R		–	–	✓
	5	P2IN5	0	H0	R		–	–	✓
	4	P2IN4	0	H0	R		–	–	✓
	3	P2IN3	0	H0	R		–	✓	✓
	2	P2IN2	0	H0	R		–	✓	✓
	1	P2IN1	0	H0	R		–	✓	✓
	0	P2IN0	0	H0	R		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P2IOEN (P2 Port Enable Register)	15	P2IEN7	0	H0	R/W	–	–	–	✓
	14	P2IEN6	0	H0	R/W		–	–	✓
	13	P2IEN5	0	H0	R/W		–	–	✓
	12	P2IEN4	0	H0	R/W		–	–	✓
	11	P2IEN3	0	H0	R/W		–	✓	✓
	10	P2IEN2	0	H0	R/W		–	✓	✓
	9	P2IEN1	0	H0	R/W		–	✓	✓
	8	P2IEN0	0	H0	R/W		✓	✓	✓
	7	P2OEN7	0	H0	R/W	–	–	–	✓
	6	P2OEN6	0	H0	R/W		–	–	✓
	5	P2OEN5	0	H0	R/W		–	–	✓
	4	P2OEN4	0	H0	R/W		–	–	✓
	3	P2OEN3	0	H0	R/W		–	✓	✓
	2	P2OEN2	0	H0	R/W		–	✓	✓
	1	P2OEN1	0	H0	R/W		–	✓	✓
	0	P2OEN0	0	H0	R/W		✓	✓	✓
P2RCTL (P2 Port Pull-up/down Control Register)	15	P2PDPU7	0	H0	R/W	–	–	–	✓
	14	P2PDPU6	0	H0	R/W		–	–	✓
	13	P2PDPU5	0	H0	R/W		–	–	✓
	12	P2PDPU4	0	H0	R/W		–	–	✓
	11	P2PDPU3	0	H0	R/W		–	✓	✓
	10	P2PDPU2	0	H0	R/W		–	✓	✓
	9	P2PDPU1	0	H0	R/W		–	✓	✓
	8	P2PDPU0	0	H0	R/W		✓	✓	✓
	7	P2REN7	0	H0	R/W	–	–	–	✓
	6	P2REN6	0	H0	R/W		–	–	✓
	5	P2REN5	0	H0	R/W		–	–	✓
	4	P2REN4	0	H0	R/W		–	–	✓
	3	P2REN3	0	H0	R/W		–	✓	✓
	2	P2REN2	0	H0	R/W		–	✓	✓
	1	P2REN1	0	H0	R/W		–	✓	✓
	0	P2REN0	0	H0	R/W		✓	✓	✓
P2INTF (P2 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P2IF7	0	H0	R/W	Cleared by writing 1.	–	–	✓
	6	P2IF6	0	H0	R/W		–	–	✓
	5	P2IF5	0	H0	R/W		–	–	✓
	4	P2IF4	0	H0	R/W		–	–	✓
	3	P2IF3	0	H0	R/W		–	✓	✓
	2	P2IF2	0	H0	R/W		–	✓	✓
	1	P2IF1	0	H0	R/W		–	✓	✓
	0	P2IF0	0	H0	R/W		✓	✓	✓
P2INTCTL (P2 Port Interrupt Control Register)	15	P2EDGE7	0	H0	R/W	–	–	–	✓
	14	P2EDGE6	0	H0	R/W		–	–	✓
	13	P2EDGE5	0	H0	R/W		–	–	✓
	12	P2EDGE4	0	H0	R/W		–	–	✓
	11	P2EDGE3	0	H0	R/W		–	✓	✓
	10	P2EDGE2	0	H0	R/W		–	✓	✓
	9	P2EDGE1	0	H0	R/W		–	✓	✓
	8	P2EDGE0	0	H0	R/W		✓	✓	✓
	7	P2IE7	0	H0	R/W	–	–	–	✓
	6	P2IE6	0	H0	R/W		–	–	✓
	5	P2IE5	0	H0	R/W		–	–	✓
	4	P2IE4	0	H0	R/W		–	–	✓
	3	P2IE3	0	H0	R/W		–	✓	✓
	2	P2IE2	0	H0	R/W		–	✓	✓
	1	P2IE1	0	H0	R/W		–	✓	✓
	0	P2IE0	0	H0	R/W		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P2CHATEN (P2 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P2CHATEN7	0	H0	R/W	–	–	–	✓
	6	P2CHATEN6	0	H0	R/W		–	–	✓
	5	P2CHATEN5	0	H0	R/W		–	–	✓
	4	P2CHATEN4	0	H0	R/W		–	–	✓
	3	P2CHATEN3	0	H0	R/W		–	✓	✓
	2	P2CHATEN2	0	H0	R/W		–	✓	✓
	1	P2CHATEN1	0	H0	R/W		–	✓	✓
	0	P2CHATEN0	0	H0	R/W		✓	✓	✓
P2MODSEL (P2 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P2SEL7	0	H0	R/W	–	–	–	✓
	6	P2SEL6	0	H0	R/W		–	–	✓
	5	P2SEL5	0	H0	R/W		–	–	✓
	4	P2SEL4	0	H0	R/W		–	–	✓
	3	P2SEL3	0	H0	R/W		–	✓	✓
	2	P2SEL2	0	H0	R/W		–	✓	✓
	1	P2SEL1	0	H0	R/W		–	✓	✓
	0	P2SEL0	0	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 7.7.3.2 P2 Port Group Function Assignment

Port name	P2SEL*y* = 0	P2SEL*y* = 1								32 pin	48 pin	64 pin
	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	SENB0	UPMUX	*1	–	–	–	–	✓	✓	✓
P21	P21	RFC	SENA0	UPMUX	*1	–	–	–	–	–	✓	✓
P22	P22	RFC	REF0	UPMUX	*1	–	–	–	–	–	✓	✓
P23	P23	RFC	RFIN0	UPMUX	*1	–	–	–	–	–	✓	✓
P24	P24	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P25	P25	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P26	P26	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P27	P27	–	–	UPMUX	*1	–	–	–	–	–	–	✓

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

7.7.4 P3 Port Group

The P3 port group supports the GPIO and interrupt functions.

Table 7.7.4.1 Control Registers for P3 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P3DAT (P3 Port Data Register)	15	P3OUT7	0	H0	R/W	–	–	–	✓
	14	P3OUT6	0	H0	R/W		–	–	✓
	13	P3OUT5	0	H0	R/W		–	✓	✓
	12	P3OUT4	0	H0	R/W		–	✓	✓
	11	P3OUT3	0	H0	R/W		–	✓	✓
	10	P3OUT2	0	H0	R/W		✓	✓	✓
	9	P3OUT1	0	H0	R/W		✓	✓	✓
	8	P3OUT0	0	H0	R/W		–	–	✓
	7	P3IN7	0	H0	R	–	–	–	✓
	6	P3IN6	0	H0	R		–	–	✓
	5	P3IN5	0	H0	R		–	✓	✓
	4	P3IN4	0	H0	R		–	✓	✓
	3	P3IN3	0	H0	R		–	✓	✓
	2	P3IN2	0	H0	R		✓	✓	✓
	1	P3IN1	0	H0	R		✓	✓	✓
	0	P3IN0	0	H0	R		–	–	✓
P3IOEN (P3 Port Enable Register)	15	P3IEN7	0	H0	R/W	–	–	–	✓
	14	P3IEN6	0	H0	R/W		–	–	✓
	13	P3IEN5	0	H0	R/W		–	✓	✓
	12	P3IEN4	0	H0	R/W		–	✓	✓
	11	P3IEN3	0	H0	R/W		–	✓	✓
	10	P3IEN2	0	H0	R/W		✓	✓	✓
	9	P3IEN1	0	H0	R/W		✓	✓	✓
	8	P3IEN0	0	H0	R/W		–	–	✓
	7	P3OEN7	0	H0	R/W	–	–	–	✓
	6	P3OEN6	0	H0	R/W		–	–	✓
	5	P3OEN5	0	H0	R/W		–	✓	✓
	4	P3OEN4	0	H0	R/W		–	✓	✓
	3	P3OEN3	0	H0	R/W		–	✓	✓
	2	P3OEN2	0	H0	R/W		✓	✓	✓
	1	P3OEN1	0	H0	R/W		✓	✓	✓
	0	P3OEN0	0	H0	R/W		–	–	✓
P3RCTL (P3 Port Pull-up/down Control Register)	15	P3PDPU7	0	H0	R/W	–	–	–	✓
	14	P3PDPU6	0	H0	R/W		–	–	✓
	13	P3PDPU5	0	H0	R/W		–	✓	✓
	12	P3PDPU4	0	H0	R/W		–	✓	✓
	11	P3PDPU3	0	H0	R/W		–	✓	✓
	10	P3PDPU2	0	H0	R/W		✓	✓	✓
	9	P3PDPU1	0	H0	R/W		✓	✓	✓
	8	P3PDPU0	0	H0	R/W		–	–	✓
	7	P3REN7	0	H0	R/W	–	–	–	✓
	6	P3REN6	0	H0	R/W		–	–	✓
	5	P3REN5	0	H0	R/W		–	✓	✓
	4	P3REN4	0	H0	R/W		–	✓	✓
	3	P3REN3	0	H0	R/W		–	✓	✓
	2	P3REN2	0	H0	R/W		✓	✓	✓
	1	P3REN1	0	H0	R/W		✓	✓	✓
	0	P3REN0	0	H0	R/W		–	–	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P3INTF (P3 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P3IF7	0	H0	R/W	Cleared by writing 1.	–	–	✓
	6	P3IF6	0	H0	R/W		–	–	✓
	5	P3IF5	0	H0	R/W		–	✓	✓
	4	P3IF4	0	H0	R/W		–	✓	✓
	3	P3IF3	0	H0	R/W		–	✓	✓
	2	P3IF2	0	H0	R/W		✓	✓	✓
	1	P3IF1	0	H0	R/W		✓	✓	✓
	0	P3IF0	0	H0	R/W		–	–	✓
P3INTCTL (P3 Port Interrupt Control Register)	15	P3EDGE7	0	H0	R/W	–	–	–	✓
	14	P3EDGE6	0	H0	R/W		–	–	✓
	13	P3EDGE5	0	H0	R/W		–	✓	✓
	12	P3EDGE4	0	H0	R/W		–	✓	✓
	11	P3EDGE3	0	H0	R/W		–	✓	✓
	10	P3EDGE2	0	H0	R/W		✓	✓	✓
	9	P3EDGE1	0	H0	R/W		✓	✓	✓
	8	P3EDGE0	0	H0	R/W		–	–	✓
	7	P3IE7	0	H0	R/W	–	–	–	✓
	6	P3IE6	0	H0	R/W		–	–	✓
	5	P3IE5	0	H0	R/W		–	✓	✓
	4	P3IE4	0	H0	R/W		–	✓	✓
	3	P3IE3	0	H0	R/W		–	✓	✓
	2	P3IE2	0	H0	R/W		✓	✓	✓
	1	P3IE1	0	H0	R/W		✓	✓	✓
	0	P3IE0	0	H0	R/W		–	–	✓
P3CHATEN (P3 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P3CHATEN7	0	H0	R/W	–	–	–	✓
	6	P3CHATEN6	0	H0	R/W		–	–	✓
	5	P3CHATEN5	0	H0	R/W		–	✓	✓
	4	P3CHATEN4	0	H0	R/W		–	✓	✓
	3	P3CHATEN3	0	H0	R/W		–	✓	✓
	2	P3CHATEN2	0	H0	R/W		✓	✓	✓
	1	P3CHATEN1	0	H0	R/W		✓	✓	✓
	0	P3CHATEN0	0	H0	R/W		–	–	✓
P3MODSEL (P3 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7	P3SEL7	0	H0	R/W	–	–	–	✓
	6	P3SEL6	0	H0	R/W		–	–	✓
	5	P3SEL5	0	H0	R/W		–	✓	✓
	4	P3SEL4	0	H0	R/W		–	✓	✓
	3	P3SEL3	0	H0	R/W		–	✓	✓
	2	P3SEL2	0	H0	R/W		✓	✓	✓
	1	P3SEL1	0	H0	R/W		✓	✓	✓
	0	P3SEL0	0	H0	R/W		–	–	✓
P3FNCSEL (P3 Port Function Select Register)	15–14	P37MUX[1:0]	0x0	H0	R/W	–	–	–	✓
	13–12	P36MUX[1:0]	0x0	H0	R/W		–	–	✓
	11–10	P35MUX[1:0]	0x0	H0	R/W		–	✓	✓
	9–8	P34MUX[1:0]	0x0	H0	R/W		–	✓	✓
	7–6	P33MUX[1:0]	0x0	H0	R/W		–	✓	✓
	5–4	P32MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	3–2	P31MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	1–0	P30MUX[1:0]	0x0	H0	R/W		–	–	✓

Table 7.7.4.2 P3 Port Group Function Assignment

Port name	P3SELy = 0	P3SELy = 1								32 pin	48 pin	64 pin
	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	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P31	P31	T16B Ch.1	EXCL10	UPMUX	*1	–	–	–	–	✓	✓	✓
P32	P32	T16B Ch.1	EXCL11	UPMUX	*1	–	–	–	–	✓	✓	✓
P33	P33	RFC	RFCLKO0	UPMUX	*1	–	–	–	–	–	✓	✓
P34	P34	REMC3	REMO	UPMUX	*1	–	–	–	–	–	✓	✓
P35	P35	REMC3	CLPLS	UPMUX	*1	–	–	–	–	–	✓	✓
P36	P36	–	–	UPMUX	*1	–	–	–	–	–	–	✓
P37	P37	–	–	UPMUX	*1	–	–	–	–	–	–	✓

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

7.7.5 P4 Port Group

The P4 port group supports the GPIO and interrupt functions.

Table 7.7.5.1 Control Registers for P4 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P4DAT (P4 Port Data Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P4OUT5	0	H0	R/W	–	–	✓	✓
	12	P4OUT4	0	H0	R/W		✓	✓	✓
	11	P4OUT3	0	H0	R/W		✓	✓	✓
	10	P4OUT2	0	H0	R/W		✓	✓	✓
	9	P4OUT1	0	H0	R/W		–	–	✓
	8	P4OUT0	0	H0	R/W		–	–	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P4IN5	0	H0	R	–	–	✓	✓
	4	P4IN4	0	H0	R		✓	✓	✓
	3	P4IN3	0	H0	R		✓	✓	✓
	2	P4IN2	0	H0	R		✓	✓	✓
	1	P4IN1	0	H0	R		–	–	✓
	0	P4IN0	0	H0	R		–	–	✓
P4IOEN (P4 Port Enable Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P4IEN5	0	H0	R/W	–	–	✓	✓
	12	P4IEN4	0	H0	R/W		✓	✓	✓
	11	P4IEN3	0	H0	R/W		✓	✓	✓
	10	P4IEN2	0	H0	R/W		✓	✓	✓
	9	P4IEN1	0	H0	R/W		–	–	✓
	8	P4IEN0	0	H0	R/W		–	–	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P4OEN5	0	H0	R/W	–	–	✓	✓
	4	P4OEN4	0	H0	R/W		✓	✓	✓
	3	P4OEN3	0	H0	R/W		✓	✓	✓
	2	P4OEN2	0	H0	R/W		✓	✓	✓
	1	P4OEN1	0	H0	R/W		–	–	✓
	0	P4OEN0	0	H0	R/W		–	–	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P4RCTL (P4 Port Pull-up/down Control Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P4PDPU5	0	H0	R/W	–	–	✓	✓
	12	P4PDPU4	0	H0	R/W		✓	✓	✓
	11	P4PDPU3	0	H0	R/W		✓	✓	✓
	10	P4PDPU2	0	H0	R/W		✓	✓	✓
	9	P4PDPU1	0	H0	R/W		–	–	✓
	8	P4PDPU0	0	H0	R/W		–	–	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P4REN5	0	H0	R/W	–	–	✓	✓
	4	P4REN4	0	H0	R/W		✓	✓	✓
	3	P4REN3	0	H0	R/W		✓	✓	✓
	2	P4REN2	0	H0	R/W		✓	✓	✓
	1	P4REN1	0	H0	R/W		–	–	✓
	0	P4REN0	0	H0	R/W		–	–	✓
P4INTF (P4 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7–6	–	0x0	–	R		–	–	–
	5	P4IF5	0	H0	R/W	Cleared by writing 1.	–	✓	✓
	4	P4IF4	0	H0	R/W		✓	✓	✓
	3	P4IF3	0	H0	R/W		✓	✓	✓
	2	P4IF2	0	H0	R/W		✓	✓	✓
	1	P4IF1	0	H0	R/W		–	–	✓
	0	P4IF0	0	H0	R/W		–	–	✓
P4INTCTL (P4 Port Interrupt Control Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P4EDGE5	0	H0	R/W	–	–	✓	✓
	12	P4EDGE4	0	H0	R/W		✓	✓	✓
	11	P4EDGE3	0	H0	R/W		✓	✓	✓
	10	P4EDGE2	0	H0	R/W		✓	✓	✓
	9	P4EDGE1	0	H0	R/W		–	–	✓
	8	P4EDGE0	0	H0	R/W		–	–	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P4IE5	0	H0	R/W	–	–	✓	✓
	4	P4IE4	0	H0	R/W		✓	✓	✓
	3	P4IE3	0	H0	R/W		✓	✓	✓
	2	P4IE2	0	H0	R/W		✓	✓	✓
	1	P4IE1	0	H0	R/W		–	–	✓
	0	P4IE0	0	H0	R/W		–	–	✓
P4CHATEN (P4 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7–6	–	0x0	–	R		–	–	–
	5	P4CHATEN5	0	H0	R/W	–	–	✓	✓
	4	P4CHATEN4	0	H0	R/W		✓	✓	✓
	3	P4CHATEN3	0	H0	R/W		✓	✓	✓
	2	P4CHATEN2	0	H0	R/W		✓	✓	✓
	1	P4CHATEN1	0	H0	R/W		–	–	✓
	0	P4CHATEN0	0	H0	R/W		–	–	✓
P4MODSEL (P4 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7–6	–	0x0	–	R		–	–	–
	5	P4SEL5	0	H0	R/W	–	–	✓	✓
	4	P4SEL4	0	H0	R/W		✓	✓	✓
	3	P4SEL3	0	H0	R/W		✓	✓	✓
	2	P4SEL2	0	H0	R/W		✓	✓	✓
	1	P4SEL1	0	H0	R/W		–	–	✓
	0	P4SEL0	0	H0	R/W		–	–	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P4FNCSEL (P4 Port Function Select Register)	15–12	–	0x0	H0	R/W	–	–	–	–
	11–10	P45MUX[1:0]	0x0	H0	R/W	–	–	✓	✓
	9–8	P44MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	7–6	P43MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	5–4	P42MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	3–2	P41MUX[1:0]	0x0	H0	R/W		–	–	✓
	1–0	P40MUX[1:0]	0x0	H0	R/W		–	–	✓

Table 7.7.5.2 P4 Port Group Function Assignment

Port name	P4SELy = 0	P4SELy = 1								32 pin	48 pin	64 pin
	GPIO	P4yMUX = 0x0 (Function 0)		P4yMUX = 0x1 (Function 1)		P4yMUX = 0x2 (Function 2)		P4yMUX = 0x3 (Function 3)				
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin			
P40	P40	–	–	–	–	–	–	–	–	–	–	✓
P41	P41	–	–	–	–	–	–	–	–	–	–	✓
P42	P42	ADC12A	#ADTRG0	–	–	–	–	–	–	✓	✓	✓
P43	P43	RTCA	RTC1S	–	–	–	–	–	–	✓	✓	✓
P44	P44	–	–	–	–	SDV3	EXSVD0	–	–	✓	✓	✓
P45	P45	–	–	–	–	SDV3	EXSVD1	–	–	–	✓	✓

7.7.6 P5 Port Group

The P5 port group supports the GPIO and interrupt functions.

Table 7.7.6.1 Control Registers for P5 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P5DAT (P5 Port Data Register)	15	–	0	–	R	–	–	–	–
	14	P5OUT6	0	H0	R/W	–	–	✓	✓
	13	P5OUT5	0	H0	R/W		–	✓	✓
	12	P5OUT4	0	H0	R/W		✓	✓	✓
	11	P5OUT3	0	H0	R/W		–	–	✓
	10	P5OUT2	0	H0	R/W		–	–	✓
	9	P5OUT1	0	H0	R/W		✓	✓	✓
	8	P5OUT0	0	H0	R/W		✓	✓	✓
	7	–	0	–	R	–	–	–	–
	6	P5IN6	0	H0	R	–	–	✓	✓
	5	P5IN5	0	H0	R		–	✓	✓
	4	P5IN4	0	H0	R		✓	✓	✓
	3	P5IN3	0	H0	R		–	–	✓
	2	P5IN2	0	H0	R		–	–	✓
	1	P5IN1	0	H0	R		✓	✓	✓
	0	P5IN0	0	H0	R		✓	✓	✓
P5IOEN (P5 Port Enable Register)	15	–	0	–	R	–	–	–	–
	14	P5IEN6	0	H0	R/W	–	–	✓	✓
	13	P5IEN5	0	H0	R/W		–	✓	✓
	12	P5IEN4	0	H0	R/W		✓	✓	✓
	11	P5IEN3	0	H0	R/W		–	–	✓
	10	P5IEN2	0	H0	R/W		–	–	✓
	9	P5IEN1	0	H0	R/W		✓	✓	✓
	8	P5IEN0	0	H0	R/W		✓	✓	✓
	7	–	0	–	R	–	–	–	–
	6	P5OEN6	0	H0	R/W	–	–	✓	✓
	5	P5OEN5	0	H0	R/W		–	✓	✓
	4	P5OEN4	0	H0	R/W		✓	✓	✓
	3	P5OEN3	0	H0	R/W		–	–	✓
	2	P5OEN2	0	H0	R/W		–	–	✓
	1	P5OEN1	0	H0	R/W		✓	✓	✓
	0	P5OEN0	0	H0	R/W		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P5RCTL (P5 Port Pull-up/down Control Register)	15	–	0	–	R	–	–	–	–
	14	P5PDPU6	0	H0	R/W	–	–	✓	✓
	13	P5PDPU5	0	H0	R/W		–	✓	✓
	12	P5PDPU4	0	H0	R/W		✓	✓	✓
	11	P5PDPU3	0	H0	R/W		–	–	✓
	10	P5PDPU2	0	H0	R/W		–	–	✓
	9	P5PDPU1	0	H0	R/W		✓	✓	✓
	8	P5PDPU0	0	H0	R/W		✓	✓	✓
	7	–	0	–	R	–	–	–	–
	6	P5REN6	0	H0	R/W	–	–	✓	✓
	5	P5REN5	0	H0	R/W		–	✓	✓
	4	P5REN4	0	H0	R/W		✓	✓	✓
	3	P5REN3	0	H0	R/W		–	–	✓
	2	P5REN2	0	H0	R/W		–	–	✓
	1	P5REN1	0	H0	R/W		✓	✓	✓
	0	P5REN0	0	H0	R/W		✓	✓	✓
P5INTF (P5 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7	–	0	–	R	–	–	–	–
	6	P5IF6	0	H0	R/W	Cleared by writing 1.	–	✓	✓
	5	P5IF5	0	H0	R/W		–	✓	✓
	4	P5IF4	0	H0	R/W		✓	✓	✓
	3	P5IF3	0	H0	R/W		–	–	✓
	2	P5IF2	0	H0	R/W		–	–	✓
	1	P5IF1	0	H0	R/W		✓	✓	✓
	0	P5IF0	0	H0	R/W		✓	✓	✓
P5INTCTL (P5 Port Interrupt Control Register)	15	–	0	–	R	–	–	–	–
	14	P5EDGE6	0	H0	R/W	–	–	✓	✓
	13	P5EDGE5	0	H0	R/W		–	✓	✓
	12	P5EDGE4	0	H0	R/W		✓	✓	✓
	11	P5EDGE3	0	H0	R/W		–	–	✓
	10	P5EDGE2	0	H0	R/W		–	–	✓
	9	P5EDGE1	0	H0	R/W		✓	✓	✓
	8	P5EDGE0	0	H0	R/W		✓	✓	✓
	7	–	0	–	R	–	–	–	–
	6	P5IE6	0	H0	R/W	–	–	✓	✓
	5	P5IE5	0	H0	R/W		–	✓	✓
	4	P5IE4	0	H0	R/W		✓	✓	✓
	3	P5IE3	0	H0	R/W		–	–	✓
	2	P5IE2	0	H0	R/W		–	–	✓
	1	P5IE1	0	H0	R/W		✓	✓	✓
	0	P5IE0	0	H0	R/W		✓	✓	✓
P5CHATEN (P5 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7	–	0	–	R	–	–	–	–
	6	P5CHATEN6	0	H0	R/W	–	–	✓	✓
	5	P5CHATEN5	0	H0	R/W		–	✓	✓
	4	P5CHATEN4	0	H0	R/W		✓	✓	✓
	3	P5CHATEN3	0	H0	R/W		–	–	✓
	2	P5CHATEN2	0	H0	R/W		–	–	✓
	1	P5CHATEN1	0	H0	R/W		✓	✓	✓
	0	P5CHATEN0	0	H0	R/W		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P5MODSEL (P5 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7	–	0	–	R	–	–	–	–
	6	P5SEL6	0	H0	R/W	–	–	✓	✓
	5	P5SEL5	0	H0	R/W		–	✓	✓
	4	P5SEL4	0	H0	R/W		✓	✓	✓
	3	P5SEL3	0	H0	R/W		–	–	✓
	2	P5SEL2	0	H0	R/W		–	–	✓
	1	P5SEL1	1	H0	R/W		✓	✓	✓
	0	P5SEL0	1	H0	R/W		✓	✓	✓
P5FNCSEL (P5 Port Function Select Register)	15–14	–	0x0	–	R	–	–	–	–
	13–12	P56MUX[1:0]	0x0	H0	R/W	–	–	✓	✓
	11–10	P55MUX[1:0]	0x0	H0	R/W		–	✓	✓
	9–8	P54MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	7–6	P53MUX[1:0]	0x0	H0	R/W		–	–	✓
	5–4	P52MUX[1:0]	0x0	H0	R/W		–	–	✓
	3–2	P51MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	1–0	P50MUX[1:0]	0x0	H0	R/W		✓	✓	✓

Table 7.7.6.2 P5 Port Group Function Assignment

Port name	P5SEL*y* = 0	P5SEL*y* = 1								32 pin	48 pin	64 pin
	GPIO	P5*y*MUX = 0x0 (Function 0)		P5*y*MUX = 0x1 (Function 1)		P5*y*MUX = 0x2 (Function 2)		P5*y*MUX = 0x3 (Function 3)				
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin			
P50	P50	SDAC2	SDACOUT_P	–	–	–	–	–	–	✓	✓	✓
P51	P51	SDAC2	SDACOUT_N	–	–	–	–	–	–	✓	✓	✓
P52	P52	–	–	–	–	–	–	–	–	–	–	✓
P53	P53	–	–	–	–	–	–	–	–	–	–	✓
P54	P54	CLG	EXOSC	–	–	–	–	–	–	✓	✓	✓
P55	P55	T16B Cn.0	EXCL00	–	–	–	–	–	–	–	✓	✓
P56	P56	T16B Cn.0	EXCL01	–	–	–	–	–	–	–	✓	✓

7.7.7 P6 Port Group

The P6 port group supports the GPIO and interrupt functions.

Table 7.7.7.1 Control Registers for P6 Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P6DAT (P6 Port Data Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P6OUT5	0	H0	R/W	–	✓	✓	✓
	12	P6OUT4	0	H0	R/W		✓	✓	✓
	11	P6OUT3	0	H0	R/W		✓	✓	✓
	10	P6OUT2	0	H0	R/W		✓	✓	✓
	9	P6OUT1	0	H0	R/W		✓	✓	✓
	8	P6OUT0	0	H0	R/W		✓	✓	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P6IN5	0	H0	R	–	✓	✓	✓
	4	P6IN4	0	H0	R		✓	✓	✓
	3	P6IN3	0	H0	R		✓	✓	✓
	2	P6IN2	0	H0	R		✓	✓	✓
	1	P6IN1	0	H0	R		✓	✓	✓
	0	P6IN0	0	H0	R		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P6IOEN (P6 Port Enable Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P6IEN5	0	H0	R/W	–	✓	✓	✓
	12	P6IEN4	0	H0	R/W		✓	✓	✓
	11	P6IEN3	0	H0	R/W		✓	✓	✓
	10	P6IEN2	0	H0	R/W		✓	✓	✓
	9	P6IEN1	0	H0	R/W		✓	✓	✓
	8	P6IEN0	0	H0	R/W		✓	✓	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P6OEN5	0	H0	R/W	–	✓	✓	✓
	4	P6OEN4	0	H0	R/W		✓	✓	✓
	3	P6OEN3	0	H0	R/W		✓	✓	✓
	2	P6OEN2	0	H0	R/W		✓	✓	✓
	1	P6OEN1	0	H0	R/W		✓	✓	✓
	0	P6OEN0	0	H0	R/W		✓	✓	✓
P6RCTL (P6 Port Pull-up/down Control Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P6PDPU5	0	H0	R/W	–	✓	✓	✓
	12	P6PDPU4	0	H0	R/W		✓	✓	✓
	11	P6PDPU3	0	H0	R/W		✓	✓	✓
	10	P6PDPU2	0	H0	R/W		✓	✓	✓
	9	P6PDPU1	0	H0	R/W		✓	✓	✓
	8	P6PDPU0	0	H0	R/W		✓	✓	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P6REN5	0	H0	R/W	–	✓	✓	✓
	4	P6REN4	0	H0	R/W		✓	✓	✓
	3	P6REN3	0	H0	R/W		✓	✓	✓
	2	P6REN2	0	H0	R/W		✓	✓	✓
	1	P6REN1	0	H0	R/W		✓	✓	✓
	0	P6REN0	0	H0	R/W		✓	✓	✓
P6INTF (P6 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
	7–6	–	0x0	–	R		–	–	–
	5	P6IF5	0	H0	R/W	Cleared by writing 1.	✓	✓	✓
	4	P6IF4	0	H0	R/W		✓	✓	✓
	3	P6IF3	0	H0	R/W		✓	✓	✓
	2	P6IF2	0	H0	R/W		✓	✓	✓
	1	P6IF1	0	H0	R/W		✓	✓	✓
	0	P6IF0	0	H0	R/W		✓	✓	✓
P6INTCTL (P6 Port Interrupt Control Register)	15–14	–	0x0	–	R	–	–	–	–
	13	P6EDGE5	0	H0	R/W	–	✓	✓	✓
	12	P6EDGE4	0	H0	R/W		✓	✓	✓
	11	P6EDGE3	0	H0	R/W		✓	✓	✓
	10	P6EDGE2	0	H0	R/W		✓	✓	✓
	9	P6EDGE1	0	H0	R/W		✓	✓	✓
	8	P6EDGE0	0	H0	R/W		✓	✓	✓
	7–6	–	0x0	–	R	–	–	–	–
	5	P6IE5	0	H0	R/W	–	✓	✓	✓
	4	P6IE4	0	H0	R/W		✓	✓	✓
	3	P6IE3	0	H0	R/W		✓	✓	✓
	2	P6IE2	0	H0	R/W		✓	✓	✓
	1	P6IE1	0	H0	R/W		✓	✓	✓
	0	P6IE0	0	H0	R/W		✓	✓	✓
P6CHATEN (P6 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
	7–6	–	0x0	–	R		–	–	–
	5	P6CHATEN5	0	H0	R/W	–	✓	✓	✓
	4	P6CHATEN4	0	H0	R/W		✓	✓	✓
	3	P6CHATEN3	0	H0	R/W		✓	✓	✓
	2	P6CHATEN2	0	H0	R/W		✓	✓	✓
	1	P6CHATEN1	0	H0	R/W		✓	✓	✓
	0	P6CHATEN0	0	H0	R/W		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
P6MODSEL (P6 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7–6	–	0x0	–	R		–	–	–
	5	P6SEL5	0	H0	R/W	–	✓	✓	✓
	4	P6SEL4	0	H0	R/W		✓	✓	✓
	3	P6SEL3	0	H0	R/W		✓	✓	✓
	2	P6SEL2	0	H0	R/W		✓	✓	✓
	1	P6SEL1	0	H0	R/W		✓	✓	✓
	0	P6SEL0	0	H0	R/W		✓	✓	✓
P6FNCSEL (P6 Port Function Select Register)	15–12	–	0x0	H0	R/W	–	–	–	–
	11–10	P65MUX[1:0]	0x0	H0	R/W	–	✓	✓	✓
	9–8	P64MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	7–6	P63MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	5–4	P62MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	3–2	P61MUX[1:0]	0x0	H0	R/W		✓	✓	✓
	1–0	P60MUX[1:0]	0x0	H0	R/W		✓	✓	✓

Table 7.7.7.2 P6 Port Group Function Assignment

Port name	P6SEL*y* = 0	P6SEL*y* = 1								32 pin	48 pin	64 pin
	GPIO	P6*y*MUX = 0x0 (Function 0)		P6*y*MUX = 0x1 (Function 1)		P6*y*MUX = 0x2 (Function 2)		P6*y*MUX = 0x3 (Function 3)				
		Peripheral	Pin	Peripheral	Pin	Peripheral	Pin	Peripheral	Pin			
P60	P60	QSPI Ch.0	QSPICLK0	–	–	–	–	–	–	✓	✓	✓
P61	P61	QSPI Ch.0	QSDIO00	–	–	–	–	–	–	✓	✓	✓
P62	P62	QSPI Ch.0	QSDIO01	–	–	–	–	–	–	✓	✓	✓
P63	P63	QSPI Ch.0	QSDIO02	–	–	–	–	–	–	✓	✓	✓
P64	P64	QSPI Ch.0	QSDIO03	–	–	–	–	–	–	✓	✓	✓
P65	P65	QSPI Ch.0	#QSPISS0	–	–	–	–	–	–	✓	✓	✓

7.7.8 Pd Port Group

The Pd port group support the GPIO function. The Pd0 and Pd1 ports are configured as debugging function ports at initialization.

Table 7.7.8.1 Control Registers for Pd Port Group

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
PDDAT (Pd Port Data Register)	15–12	–	0x0	–	R	–	–	–	–
	11	PDOUT3	0	H0	R/W	–	✓	✓	✓
	10	PDOUT2	0	H0	R/W		✓	✓	✓
	9	PDOUT1	0	H0	R/W		✓	✓	✓
	8	PDOUT0	0	H0	R/W		✓	✓	✓
	7–4	–	0x0	–	R	–	–	–	–
	3	PDIN3	0	H0	R	–	✓	✓	✓
	2	PDIN2	0	H0	R		✓	✓	✓
	1	PDIN1	0	H0	R		✓	✓	✓
	0	PDIN0	0	H0	R		✓	✓	✓
PDIOEN (Pd Port Enable Register)	15–12	–	0x0	–	R	–	–	–	–
	11	PDIEN3	0	H0	R/W	–	✓	✓	✓
	10	PDIEN2	0	H0	R/W		✓	✓	✓
	9	PDIEN1	0	H0	R/W		✓	✓	✓
	8	PDIEN0	0	H0	R/W		✓	✓	✓
	7–4	–	0x0	–	R	–	–	–	–
	3	PDOEN3	0	H0	R/W	–	✓	✓	✓
	2	PDOEN2	0	H0	R/W		✓	✓	✓
	1	PDOEN1	0	H0	R/W		✓	✓	✓
	0	PDOEN0	0	H0	R/W		✓	✓	✓

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
PDRCTL (Pd Port Pull-up/down Control Register)	15–12	–	0x0	–	R	–	–	–	–
	11	PDPDPU3	0	H0	R/W	–	✓	✓	✓
	10	PDPDPU2	0	H0	R/W		✓	✓	✓
	9	PDPDPU1	0	H0	R/W		✓	✓	✓
	8	PDPDPU0	0	H0	R/W		✓	✓	✓
	7–4	–	0x0	–	R	–	–	–	–
	3	PDREN3	0	H0	R/W	–	✓	✓	✓
	2	PDREN2	0	H0	R/W		✓	✓	✓
	1	PDREN1	0	H0	R/W		✓	✓	✓
	0	PDREN0	0	H0	R/W		✓	✓	✓
PDMODSEL (Pd Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
	7–4	–	0x0	–	R		–	–	–
	3	PDSEL3	0	H0	R/W	–	✓	✓	✓
	2	PDSEL2	0	H0	R/W		✓	✓	✓
	1	PDSEL1	1	H0	R/W		✓	✓	✓
	0	PDSEL0	1	H0	R/W		✓	✓	✓
PDFNCSEL (Pd Port Function Select Register)	15–8	–	0x00	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 7.7.8.2 Pd Port Group Function Assignment

Port name	PDSEL*y* = 0	PDSEL*y* = 1								32 pin	48 pin	64 pin
	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	CPU	SWCLK	–	–	–	–	–	–	✓	✓	✓
Pd1	Pd1	CPU	SWD	–	–	–	–	–	–	✓	✓	✓
Pd2	Pd2	–	–	–	–	CLG	OSC4	–	–	✓	✓	✓
Pd3	Pd3	–	–	–	–	CLG	OSC3	–	–	✓	✓	✓

7.7.9 Common Registers between Port Groups

Table 7.7.9.1 Control Registers for Common Use with Port Groups

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
PPORTCLK (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	–	0x0	–	R	–	–	–	–
	1–0	CLKSRC[1:0]	0x0	H0	R/WP	–	✓	✓	✓
PPORTINTFGRP (P Port Interrupt Flag Group Register)	15–8	–	0x00	–	R	–	–	–	–
	7	–	0	–	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		–	✓	✓

8 Universal Port Multiplexer (UPMUX)

8.1 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 I^{2}C, UART, synchronous serial interface, and 16-bit PWM timer peripheral I/O functions to the P0, P1, P2, and P3 port groups.
- The peripheral I/O function assigned via UPMUX is enabled by setting the PPORTP*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, 3) and '*y*' refers to a port number (*y* = 0, 1, 2, ··· , 7).

Figure 8.1.1 shows the configuration of UPMUX.

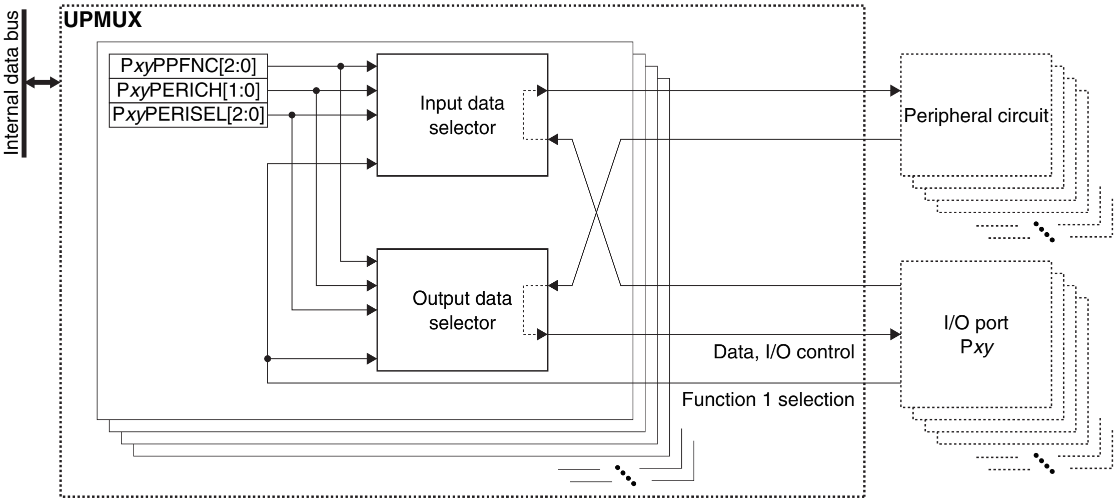


Figure 8.1.1 UPMUX Configuration

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

8.3 Control Registers

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

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UPMUXP*x*MUX*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 (0–3).
*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 8.3.1.)

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 8.3.1.)

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

These bits specify a peripheral circuit. (See Table 8.3.1.)

Table 8.3.1 Peripheral I/O Function Selections

UPMUXP*x*MUX*n*.P*xy*PPFNC[2:0] bits (Peripheral I/O function)	UPMUXP*x*MUX*n*.P*xy*PERISEL[2:0] bits (Peripheral circuit)							
	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
	None *	I2C	SPIA	UART3	T16B	Reserved	Reserved	Reserved
	UPMUXP*x*MUX*n*.P*xy*PERICH[1:0] bits (Peripheral circuit channel)							
	–	0x0–0x2	0x0–0x2	0x0–0x2	0x0–0x1	–	–	–
	–	Ch.0–2	Ch.0–2	Ch.0–2	Ch.0–1	–	–	–
0x0	None *	None *	None *	None *	None *	None *	None *	None *
0x1	Reserved	SCL*n*	SDI*n*	USIN*n*	TOUT*n*0/ CAP*n*0	Reserved	Reserved	Reserved
0x2		SDA*n*	SDO*n*	USOUT*n*	TOUT*n*1/ CAP*n*1			
0x3		Reserved	SPICLK*n*	Reserved	TOUT*n*2/ CAP*n*2			
0x4			#SPISS*n*		TOUT*n*3/ CAP*n*3			
0x5			Reserved		Reserved			
0x6								
0x7								

* "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.

9 Watchdog Timer (WDT2)

9.1 Overview

WDT2 restarts the system if a problem occurs, such as when the program cannot be executed normally.
The features of WDT2 are listed below.

- Includes a 10-bit up counter to count NMI/reset generation cycle.
- A counter clock source and clock division ratio are selectable.
- Can generate a reset or NMI in a cycle given via software.
- Can generate a reset at the next NMI generation cycle after an NMI is generated.

Figure 9.1.1 shows the configuration of WDT2.

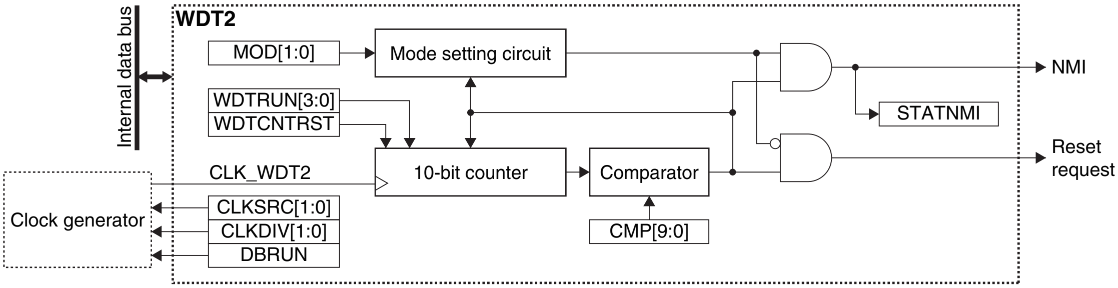


Figure 9.1.1 WDT2 Configuration

9.2 Clock Settings

9.2.1 WDT2 Operating Clock

When using WDT2, the WDT2 operating clock CLK_WDT2 must be supplied to WDT2 from the clock generator.
The CLK_WDT2 supply should be controlled as in the procedure shown below.

1. Write 0x0096 to the SYSPROT.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 WDT2CLK register bits:
   WDT2CLK.CLKSRC[1:0] bits (Clock source selection)
   WDT2CLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)
4. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

9.2.2 Clock Supply in DEBUG Mode

The CLK_WDT2 supply during DEBUG mode should be controlled using the WDT2CLK.DBRUN bit.
The CLK_WDT2 supply to WDT2 is suspended when the CPU enters DEBUG mode if the WDT2CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_WDT2 supply resumes. Although WDT2 stops operating when the CLK_WDT2 supply is suspended, the register retains the status before DEBUG mode was entered.
If the WDT2CLK.DBRUN bit = 1, the CLK_WDT2 supply is not suspended and WDT2 will keep operating in DEBUG mode.

9.3 Operations

9.3.1 WDT2 Control

Activating WDT2

WDT2 should be initialized and started up with the procedure listed below.

1. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
2. Configure the WDT2 operating clock.
3. Set the WDT2CTL.MOD[1:0] bits. (Select WDT2 operating mode)
4. Set the WDT2CMP.CMP[9:0] bits. (Set NMI/reset generation cycle)
5. Write 1 to the WDT2CTL.WDTCNTRST bit. (Reset WDT2 counter)
6. Write a value other than 0xa to the WDT2CTL.WDTRUN[3:0] bits. (Start up WDT2)
7. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

NMI/reset generation cycle

Use the following equation to calculate the WDT2 NMI/reset generation cycle.

$$t_{WDT} = \frac{CMP + 1}{CLK\_WDT2} \qquad \text{(Eq. 9.1)}$$

Where

t_{WDT}: NMI/reset generation cycle [second]
CLK_WDT2: WDT2 operating clock frequency [Hz]
CMP: Setting value of the WDT2CMP.CMP[9:0] bits

Example) t_{WDT} = 2.5 seconds when CLK_WDT2 = 256 Hz and the WDT2CMP.CMP[9:0] bits = 639

Resetting WDT2 counter

To prevent an unexpected NMI/reset to be generated by WDT2, its embedded counter must be reset periodically via software while WDT2 is running.

1. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
2. Write 1 to the WDT2CTL.WDTCNTRST bit. (Reset WDT2 counter)
3. Write a value other than 0x0096 to the SYSPROT.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, WDT2 starts counting with a new NMI/reset generation cycle.

Occurrence of counter compare match

If WDT2 is not reset within the t_{WDT} cycle for any reason and the counter reaches the setting value of the WDT2CMP.CMP[9:0] bits, a compare match occurs to cause WDT2 to issue an NMI or reset according to the setting of the WDT2CTL.MOD[1:0] bits.
If an NMI is issued, the WDT2CTL.STATNMI bit is set to 1. This bit can be cleared to 0 by writing 1 to the WDT2CTL.WDTCNTRST bit. Be sure to clear the WDT2CTL.STATNMI bit in the NMI handler routine,
If a compare match occurs, the counter is automatically reset to 0 and it continues counting.

Deactivating WDT2

WDT2 should be stopped with the procedure listed below.

1. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
2. Write 0xa to the WDT2CTL.WDTRUN[3:0] bits. (Stop WDT2)
3. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

9.3.2 Operations in HALT and SLEEP Modes

During HALT mode

WDT2 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 CPU executes the interrupt handler. To disable WDT2 in HALT mode, stop WDT2 by writing 0xa to the WDT2CTL.WDTRUN[3:0] bits before setting to HALT mode. Reset WDT2 before resuming operations after HALT mode is cleared.

During SLEEP mode

WDT2 operates in SLEEP mode if the selected clock source is running. SLEEP mode is cleared by an NMI or reset if it continues for more than the NMI/reset generation cycle and the CPU executes the interrupt handler. Therefore, stop WDT2 by setting the WDT2CTL.WDTRUN[3:0] bits before setting to SLEEP mode.

If the clock source stops in SLEEP mode, WDT2 stops. To prevent generation of an unnecessary NMI or reset after clearing SLEEP mode, reset WDT2 before executing the slp instruction. WDT2 should also be stopped as required using the WDT2CTL.WDTRUN[3:0] bits.

9.4 Control Registers

WDT2 Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
WDT2CLK	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 WDT2 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 WDT2 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 WDT2.

Table 9.4.1 Clock Source and Division Ratio Settings

WDT2CLK. CLKDIV[1:0] bits	WDT2CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x3	1/65,536	1/128	1/65,536	1/1
0x2	1/32,768		1/32,768	
0x1	1/16,384		1/16,384	
0x0	1/8,192		1/8,192	

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

WDT2 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
WDT2CTL	15–11	–	0x00	–	R	–
	10–9	MOD[1:0]	0x0	H0	R/WP	
	8	STATNMI	0	H0	R	
	7–5	–	0x0	–	R	
	4	WDTCNTRST	0	H0	WP	Always read as 0.
	3–0	WDTRUN[3:0]	0xa	H0	R/WP	–

Bits 15–11 Reserved

Bits 10–9 MOD[1:0]

These bits set the WDT2 operating mode.

Table 9.4.2 Operating Mode Setting

WDT2CTL. MOD[1:0] bits	Operating mode	Description
0x3	Reserved	–
0x2	RESET after NMI mode	If the WDT2CTL.STATNMI bit is not cleared to 0 after an NMI has occurred due to a counter compare match, WDT2 issues a reset when the next compare match occurs.
0x1	NMI mode	WDT2 issues an NMI when a counter compare match occurs.
0x0	RESET mode	WDT2 issues a reset when a counter compare match occurs.

Bit 8 STATNMI

This bit indicates that a counter compare match and NMI have occurred.

1 (R): NMI (counter compare match) occurred
0 (R): NMI not occurred

When the NMI generation function of WDT2 is used, read this bit in the NMI handler routine to confirm that WDT2 was the source of the NMI.
The WDT2CTL.STATNMI bit set to 1 is cleared to 0 by writing 1 to the WDT2CTL.WDTCNTRST bit.

Bits 7–5 Reserved

Bit 4 WDTCNTRST

This bit resets the 10-bit counter and the WDT2CTL.STATNMI bit.

1 (WP): Reset
0 (WP): Ignored
0 (R): Always 0 when being read

Bits 3–0 WDTRUN[3:0]

These bits control WDT2 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 an NMI or reset may be generated immediately after running depending on the counter value, WDT2 should also be reset concurrently when running WDT2.

WDT2 Counter Compare Match Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
WDT2CMP	15–10	–	0x00	–	R	–
	9–0	CMP[9:0]	0x3ff	H0	R/WP	

Bits 15–10 Reserved

Bits 9–0 CMP[9:0]

These bits set the NMI/reset generation cycle.

The value set in this register is compared with the 10-bit counter value while WDT2 is running, and an NMI or reset is generated when they are matched.

10 Real-Time Clock (RTCA)

10.1 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 10.1.1 shows the configuration of RTCA.

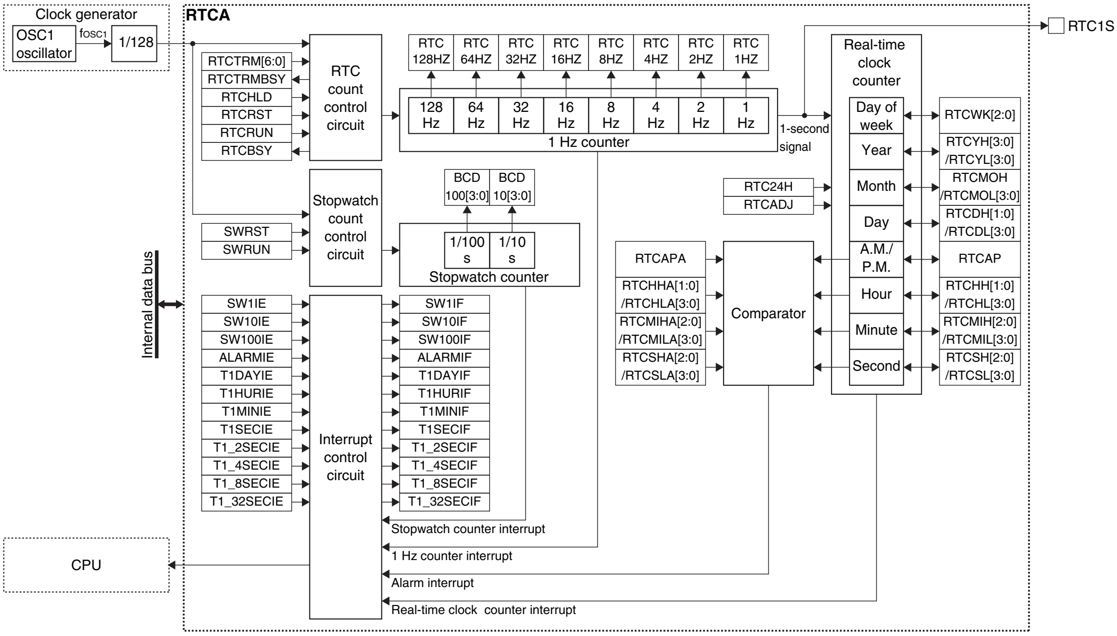


Figure 10.1.1 RTCA Configuration

10.2 Output Pin and External Connection

10.2.1 Output Pin

Table 10.2.1.1 shows the RTCA pin.

Table 10.2.1.1 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.

10.3 Clock Settings

10.3.1 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.

10.3.2 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 RTCACTLH.RTCTRM[6:0] bits from the results in Steps 1 and 2.
4. Write the value determined in Step 3 to the RTCACTLH.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 RTCACTLH.RTCTRM[6:0] bits as a two's-complement number. Use Eq. 10.1 to calculate the correction value.

$$\text{RTCTRM[6:0]} = \frac{m}{10^6} \times 256 \times n \quad \text{(However, RTCTRM[6:0] is an integer after rounding off to -64 to +63.)} \qquad \text{(Eq. 10.1)}$$

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

Figure 10.3.2.1 shows the RTC1S signal waveform.

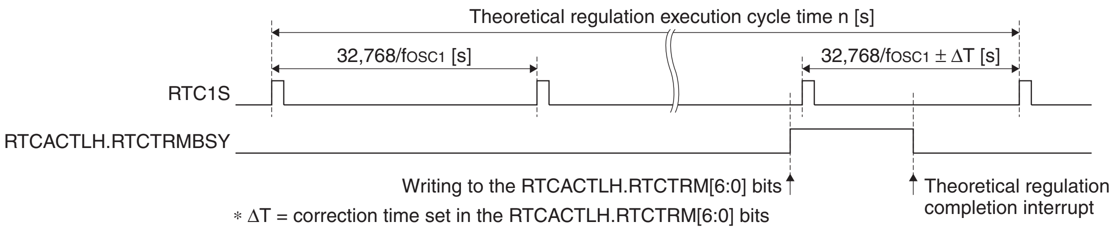


Figure 10.3.2.1 RTC1S Signal Waveform

Table 10.3.2.1 lists the frequency tolerance correction rates when the theoretical regulation execution cycle time n is 4,096 seconds as an example.

Table 10.3.2.1 Correction Rates when Theoretical Regulation Execution Cycle Time n = 4,096 Seconds

RTCACTLH.RTCTRM[6:0] bits (two's-complement)	Correction value (decimal)	Correction rate [ppm]	RTCACTLH.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 RTCACTLH.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.

10.4 Operations

10.4.1 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 RTCACTLL.RTC24H bit.
2. Write 1 to the RTCACTLL.RTCRUN bit to enable for the real-time clock counter to start counting up.
3. Check to see if the RTCACTLL.RTCBSY bit = 0 that indicates the counter is ready to rewrite. If the RTCACTLL.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.
   RTCASEC.RTCSH[2:0]/RTCSL[3:0] bits (second)
   RTCAHUR.RTCMIH[2:0]/RTCMIL[3:0] bits (minute)
   RTCAHUR.RTCHH[1:0]/RTCHL[3:0] bits (hour)
   RTCAHUR.RTCAP bit (AM/PM) (effective when RTCACTLL.RTC24H bit = 0)
   RTCAMON.RTCDH[1:0]/RTCDL[3:0] bits (day)
   RTCAMON.RTCMOH/RTCMOL[3:0] bits (month)
   RTCAYAR.RTCYH[3:0]/RTCYL[3:0] bits (year)
   RTCAYAR.RTCWK[2:0] bits (day of the week)
5 Write 1 to the RTCACTLL.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 RTCAINTF register to clear them.
7. Write 1 to the interrupt enable bits in the RTCAINTE register to enable real-time clock counter interrupts.

Time read

1. Check to see if the RTCACTLL.RTCBSY bit = 0. If the RTCACTLL.RTCBSY bit = 1, wait until it is set to 0.
2. Write 1 to the RTCACTLL.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 RTCACTLL.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 RTCAINTE.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).
   RTCAALM1.RTCSHA[2:0]/RTCSLA[3:0] bits (second)
   RTCAALM2.RTCMIHA[2:0]/RTCMILA[3:0] bits (minute)
   RTCAALM2.RTCHHA[1:0]/RTCHLA[3:0] bits (hour)
   RTCAALM2.RTCAPA bit (AM/PM) (effective when RTCACTLL.RTC24H bit = 0)
3. Write 1 to the RTCAINTF.ALARMIF bit to clear the alarm interrupt flag.
4. Write 1 to the RTCAINTE.ALARMIE bit to enable alarm interrupts.
   When the real-time clock counter reaches the alarm time set in Step 2, an alarm interrupt occurs.

10.4.2 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 RTCAMON.RTCMOL[3:0] bits to 0x0 if the RTCAMON.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 RTCACTLL.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 RTCACTLL.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 RTCACTLL.RTCHLD bit.

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

10.4.3 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 RTCASWCTL.SWRST bit to reset the stopwatch counter.
2. Write 1 to the stopwatch interrupt flags in the RTCAINTF register to clear them.
3. Write 1 to the interrupt enable bits in the RTCAINTE register to enable stopwatch interrupts.
4. Write 1 to the RTCASWCTL.SWRUN bit to start stopwatch count up operation.

Counter read

1. Read the count value from the RTCASWCTL.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.

10.4.4 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 10.4.4.1.

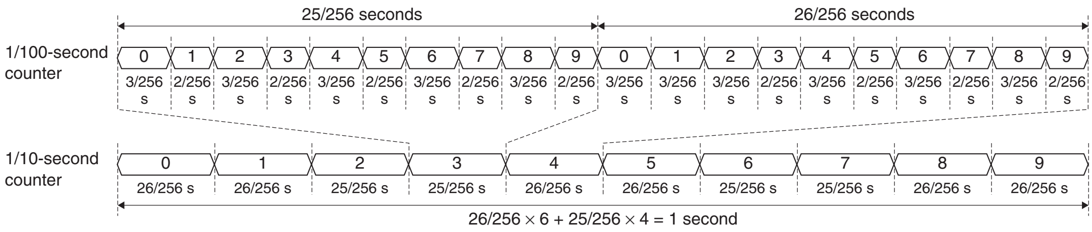


Figure 10.4.4.1 Stopwatch Count-Up Patterns

10.5 Interrupts

RTCA has a function to generate the interrupts shown in Table 10.5.1.

Table 10.5.1 RTCA Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Alarm	RTCAINTF.ALARMIF	Matching between the RTCAALM1–2 register contents and the real-time clock counter contents	Writing 1
1-day	RTCAINTF.T1DAYIF	Day counter count up	Writing 1
1-hour	RTCAINTF.T1HURIF	Hour counter count up	Writing 1
1-minute	RTCAINTF.T1MINIF	Minute counter count up	Writing 1
1-second	RTCAINTF.T1SECIF	Second counter count up	Writing 1
1/2-second	RTCAINTF.T1_2SECIF	See Figure 10.5.1.	Writing 1
1/4-second	RTCAINTF.T1_4SECIF	See Figure 10.5.1.	Writing 1
1/8-second	RTCAINTF.T1_8SECIF	See Figure 10.5.1.	Writing 1
1/32-second	RTCAINTF.T1_32SECIF	See Figure 10.5.1.	Writing 1
Stopwatch 1 Hz	RTCAINTF.SW1IF	1/10-second counter overflow	Writing 1
Stopwatch 10 Hz	RTCAINTF.SW10IF	1/10-second counter count up	Writing 1
Stopwatch 100 Hz	RTCAINTF.SW100IF	1/100-second counter count up	Writing 1
Theoretical regulation completion	RTCAINTF.RTCTRMIF	At the end of theoretical regulation operation	Writing 1

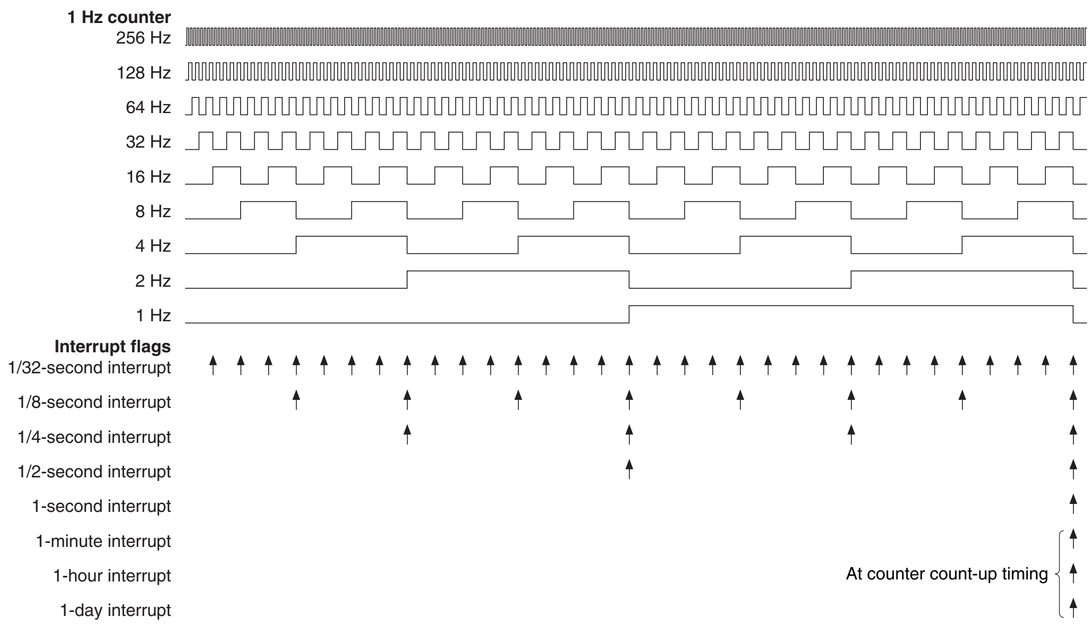


Figure 10.5.1 RTCA Interrupt Timings

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 CPU 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" chapter.

10.6 Control Registers

RTCA Control Register (Low Byte)

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCACTLL	7	–	0	–	R	–
	6	RTCBSY	0	H0	R	
	5	RTCHLD	0	H0	R/W	Cleared by setting the RTCACTLL.RTCRST bit to 1.
	4	RTC24H	0	H0	R/W	–
	3	–	0	–	R	
	2	RTCADJ	0	H0	R/W	Cleared by setting the RTCACTLL.RTCRST bit to 1.
	1	RTCRST	0	H0	R/W	–
	0	RTCRUN	0	H0	R/W	

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 RTCACTLH.RTCTRMBSY bit = 1, the RTCACTLL.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 RTCACTLL.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 RTCACTLL.RTCRUN bit = 0. The correction takes up to 2/256 seconds. The RTCACTLL.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 RTCACTLL.RTCBSY bit = 1.

• Do not write 1 to this bit again while the RTCACTLL.RTCADJ bit = 1.

Bit 1 RTCRST

This bit resets the 1 Hz counter, the RTCACTLL.RTCADJ bit, and the RTCACTLL.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.

RTCA Control Register (High Byte)

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCACTLH	7	RTCTRMBSY	0	H0	R	–
	6–0	RTCTRM[6:0]	0x00	H0	W	Read as 0x00.

Bit 7 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 RTCACTLH.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 6–0 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 RTCACTLH.RTCTRMBSY bit = 1, the RTCACTLH.RTCTRM[6:0] bits cannot be rewritten.

• Writing 0x00 to the RTCACTLH.RTCTRM[6:0] bits sets the RTCACTLH.RTCTRMBSY bit to 1 as well. However, no correcting operation is performed.

RTCA Second Alarm Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAALM1	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 RTCAALM1.RTCSHA[2:0] bits and the RTCAALM1.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 10.6.1.

Table 10.6.1 Setting Examples in BCD Code

Setting value in BCD code		Alarm (second) setting
RTCAALM1.RTCSHA[2:0] bits	RTCAALM1.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

RTCA Hour/Minute Alarm Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAALM2	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 (RTCACTLL.RTC24H bit = 0).
1 (R/W): P.M.
0 (R/W): A.M.

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

Bits 13–12 RTCHHA[1:0]
Bits 11–8 RTCHLA[3:0]

The RTCAALM2.RTCHHA[1:0] bits and the RTCAALM2.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 RTCAALM2.RTCMIHA[2:0] bits and the RTCAALM2.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.

RTCA Stopwatch Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCASWCTL	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 RTCASWCTL.BCD10[3:0] bits and the RTCASWCTL.BCD100[3:0] bits, respectively.

Note: The counter value may not be read correctly while the stopwatch counter is running. The RTCASWCTL.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 RTCASWCTL.SWRUN bit. Therefore, the counter value may be incremented (+1) from the value at writing 0.

RTCA Second/1Hz Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCASEC	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 RTCACTLL.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 RTCASEC.RTCSH[2:0] bits and the RTCASEC.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 RTCASEC.RTCSH[2:0]/RTCSL[3:0] bits while the RTCACTLL.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:
RTCASEC.RTC1HZ bit: 1 Hz
RTCASEC.RTC2HZ bit: 2 Hz
RTCASEC.RTC4HZ bit: 4 Hz
RTCASEC.RTC8HZ bit: 8 Hz
RTCASEC.RTC16HZ bit: 16 Hz
RTCASEC.RTC32HZ bit: 32 Hz
RTCASEC.RTC64HZ bit: 64 Hz
RTCASEC.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.

RTCA Hour/Minute Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAHUR	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 (RTCACTLL.RTC24H bit = 0).
1 (R/W): P.M.
0 (R/W): A.M.

In 24H mode (RTCACTLL.RTC24H bit = 1), this bit is fixed at 0 and writing 1 is ignored. However, if the RTCAHUR.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 RTCAHUR.RTCHH[1:0] bits and the RTCAHUR.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 RTCAHUR.RTCHH[1:0]/RTCHL[3:0] bits while the RTCACTLL.RTCBSY bit = 1.

Bit 7 Reserved

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

The RTCAHUR.RTCMIH[2:0] bits and the RTCAHUR.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 RTCAHUR.RTCMIH[2:0]/RTCMIL[3:0] bits while the RTCACTLL.RTCBSY bit = 1.

RTCA Month/Day Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAMON	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 RTCAMON.RTCMOH bit and the RTCAMON.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 RTCAMON.RTCMOH/RTCMOL[3:0] bits while the RTCACTLL.RTCBSY bit = 1.
- Be sure to avoid setting the RTCAMON.RTCMOH/RTCMOL[3:0] bits to 0x00.

Bits 7–6 Reserved

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

The RTCAMON.RTCDH[1:0] bits and the RTCAMON.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 RTCAMON.RTCDH[1:0]/RTCDL[3:0] bits while the RTCACTLL.RTCBSY bit = 1.

RTCA Year/Week Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAYAR	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 10.6.2 lists the correspondence between the count value and day of the week.

Table 10.6.2 Correspondence between the count value and day of the week

RTCAYAR.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 RTCAYAR.RTCWK[2:0] bits while the RTCACTLL.RTCBSY bit = 1.

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

The RTCAYAR.RTCYH[3:0] bits and the RTCAYAR.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 RTCAYAR.RTCYH[3:0]/RTCYL[3:0] bits while the RTCACTLL. RTCBSY bit = 1.

RTCA Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAINTF	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	T1DAYIF	0	H0	R/W	
	6	T1HURIF	0	H0	R/W	
	5	T1MINIF	0	H0	R/W	
	4	T1SECIF	0	H0	R/W	
	3	T1_2SECIF	0	H0	R/W	
	2	T1_4SECIF	0	H0	R/W	
	1	T1_8SECIF	0	H0	R/W	
	0	T1_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:
RTCAINTF.RTCTRMIF bit: Theoretical regulation completion interrupt
RTCAINTF.SW1IF bit: Stopwatch 1 Hz interrupt
RTCAINTF.SW10IF bit: Stopwatch 10 Hz interrupt
RTCAINTF.SW100IF bit: Stopwatch 100 Hz interrupt

Bits 11–9 Reserved

Bit 8 ALARMIF
Bit 7 T1DAYIF
Bit 6 T1HURIF
Bit 5 T1MINIF
Bit 4 T1SECIF
Bit 3 T1_2SECIF
Bit 2 T1_4SECIF
Bit 1 T1_8SECIF
Bit 0 T1_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:
RTCAINTF. ALARMIF bit: Alarm interrupt
RTCAINTF.T1DAYIF bit: 1-day interrupt
RTCAINTF.T1HURIF bit: 1-hour interrupt
RTCAINTF.T1MINIF bit: 1-minute interrupt
RTCAINTF.T1SECIF bit: 1-second interrupt
RTCAINTF.T1_2SECIF bit: 1/2-second interrupt
RTCAINTF.T1_4SECIF bit: 1/4-second interrupt
RTCAINTF.T1_8SECIF bit: 1/8-second interrupt
RTCAINTF.T1_32SECIF bit: 1/32-second interrupt

RTCA Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RTCAINTE	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	T1DAYIE	0	H0	R/W	
	6	T1HURIE	0	H0	R/W	
	5	T1MINIE	0	H0	R/W	
	4	T1SECIE	0	H0	R/W	
	3	T1_2SECIE	0	H0	R/W	
	2	T1_4SECIE	0	H0	R/W	
	1	T1_8SECIE	0	H0	R/W	
	0	T1_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:
RTCAINTE.RTCTRMIE bit: Theoretical regulation completion interrupt
RTCAINTE.SW1IE bit: Stopwatch 1 Hz interrupt
RTCAINTE.SW10IE bit: Stopwatch 10 Hz interrupt
RTCAINTE.SW100IE bit: Stopwatch 100 Hz interrupt

Bits 11–9 Reserved

Bit 8 ALARMIE
Bit 7 T1DAYIE
Bit 6 T1HURIE
Bit 5 T1MINIE
Bit 4 T1SECIE
Bit 3 T1_2SECIE
Bit 2 T1_4SECIE
Bit 1 T1_8SECIE
Bit 0 T1_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:

RTCAINTE.ALARMIE bit: Alarm interrupt
RTCAINTE.T1DAYIE bit: 1-day interrupt
RTCAINTE.T1HURIE bit: 1-hour interrupt
RTCAINTE.T1MINIE bit: 1-minute interrupt
RTCAINTE.T1SECIE bit: 1-second interrupt
RTCAINTE.T1_2SECIE bit: 1/2-second interrupt
RTCAINTE.T1_4SECIE bit: 1/4-second interrupt
RTCAINTE.T1_8SECIE bit: 1/8-second interrupt
RTCAINTE.T1_32SECIE bit: 1/32-second interrupt

11 Supply Voltage Detector (SVD3)

11.1 Overview

SVD3 is a supply voltage detector to monitor the VDD voltage, or an external voltage detection input pin. The main features are listed below.

- Power supply voltage to be detected: Selectable from VDD and external power sources (EXSVD0, EXSVD1) (Note: See the table below.)
- Detectable voltage level: Selectable from among 32 levels (max.) (Note: See the table below.)
- 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 11.1.1 shows the configuration of SVD3.

Table 11.1.1 SVD3 Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Power supply voltage to be detected	VDD and two externally input voltages (EXSVD0)	VDD and two externally input voltages (EXSVD0, EXSVD1)	
Detectable voltage level	VDD: 28 levels (1.8 to 5.0 V)/external voltage: 32 levels (1.2 to 5.0 V)		

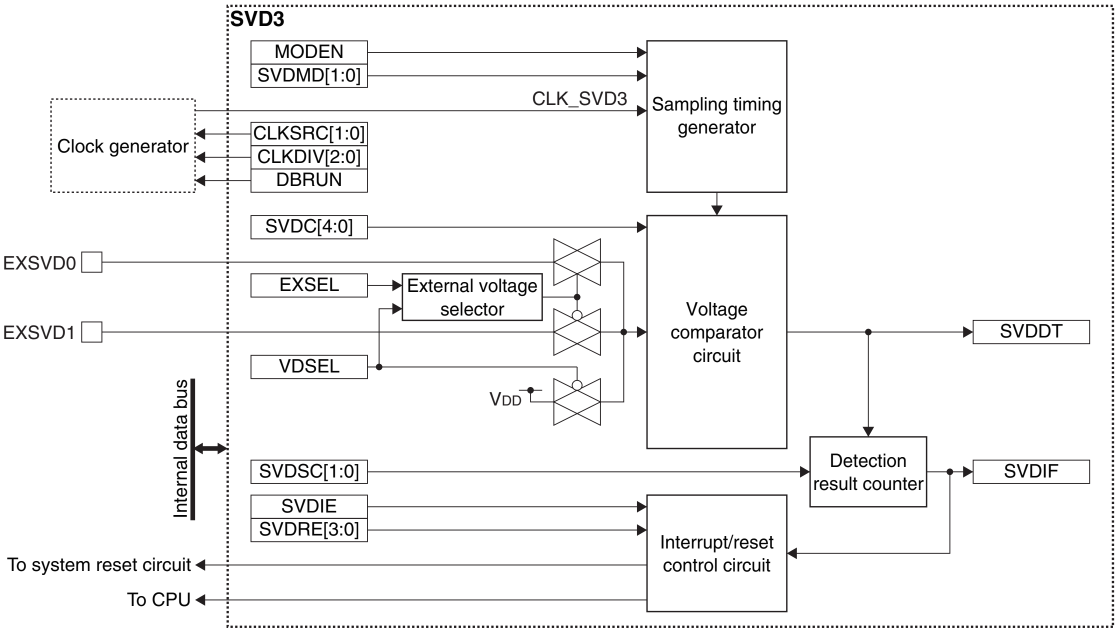


Figure 11.1.1 SVD3 Configuration

11.2 Input Pins and External Connection

11.2.1 Input Pins

Table 11.2.1.1 shows the SVD3 input pins.

Table 11.2.1.1 SVD3 Input Pins

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

* Indicates the status when the pin is configured for SVD3.

If the port is shared with the EXSVD*n* pin and other functions, the EXSVD*n* function must be assigned to the port before SVD3 can be activated. For more information, refer to the "I/O Ports" chapter.

11.2.2 External Connection

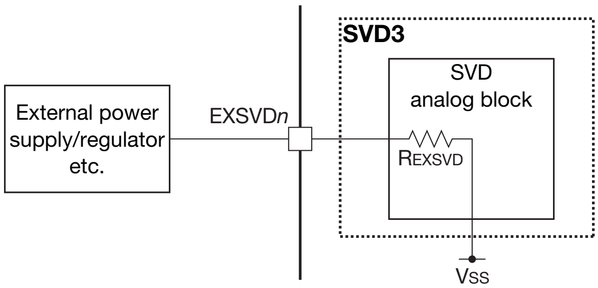


Figure 11.2.2.1 Connection between EXSVD1 Pin and External Power Supply

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

11.3 Clock Settings

11.3.1 SVD3 Operating Clock

When using SVD3, the SVD3 operating clock CLK_SVD3 must be supplied to SVD3 from the clock generator.
The CLK_SVD3 supply should be controlled as in the procedure shown below.

1. Write 0x0096 to the SYSPROT.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 SVD3CLK register bits:
   - SVD3CLK.CLKSRC[1:0] bits (Clock source selection)
   - SVD3CLK.CLKDIV[2:0] bits (Clock division ratio selection = Clock frequency setting)
4. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

The CLK_SVD3 frequency should be set to around 32 kHz.

11.3.2 Clock Supply in SLEEP Mode

When using SVD3 during SLEEP mode, the SVD3 operating clock CLK_SVD3 must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_SVD3 clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_SVD3 clock source is 1, the CLK_SVD3 clock source is deactivated during SLEEP mode and SVD3 stops with the register settings maintained at those before entering SLEEP mode. After the CPU returns to normal mode, CLK_SVD3 is supplied and the SVD3 operation resumes.

11.3.3 Clock Supply in DEBUG Mode

The CLK_SVD3 supply during DEBUG mode should be controlled using the SVD3CLK.DBRUN bit.
The CLK_SVD3 supply to SVD3 is suspended when the CPU enters DEBUG mode if the SVD3CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_SVD3 supply resumes. Although SVD3 stops operating when the CLK_SVD3 supply is suspended, the registers retain the status before DEBUG mode was entered.
If the SVD3CLK.DBRUN bit = 1, the CLK_SVD3 supply is not suspended and SVD3 will keep operating in DEBUG mode.

11.4 Operations

11.4.1 SVD3 Control

Starting detection

SVD3 should be initialized and activated with the procedure listed below.

1. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
2. Configure the operating clock using the SVD3CLK.CLKSRC[1:0] and SVD3CLK.CLKDIV[2:0] bits.
3. Set the following SVD3CTL register bits:
   - SVD3CTL.VDSEL and SVD3CTL.EXSEL bits (Select detection voltage (V_{DD}, EXSVD0, or EXSVD1))
   - SVD3CTL.SVDSC[1:0] bits (Set low power supply voltage detection counter)
   - SVD3CTL.SVDC[4:0] bits (Set SVD detection voltage V_{SVD}/EXSVD detection voltage V_{SVD_EXT})
   - SVD3CTL.SVDRE[3:0] bits (Select reset/interrupt mode)
   - SVD3CTL.SVDMD[1:0] bits (Set intermittent operation mode)
4. Set the following bits when using the interrupt:
   - Write 1 to the SVD3INTF.SVDIF bit. (Clear interrupt flag)
   - Set the SVD3INTE.SVDIE bit to 1. (Enable SVD3 interrupt)
5. Set the SVD3CTL.MODEN bit to 1. (Enable SVD3 detection)
6. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

Terminating detection

Follow the procedure shown below to stop SVD3 operation.

1. Write 0x0096 to the SYSPROT.PROT[15:0] bits. (Remove system protection)
2. Write 0 to the SVD3CTL.MODEN bit. (Disable SVD3 detection)
3. Write a value other than 0x0096 to the SYSPROT.PROT[15:0] bits. (Set system protection)

Reading detection results

The following two detection results can be obtained by reading the SVD3INTF.SVDDT bit:

- When SVD3INTF.SVDDT bit = 0
  Power supply voltage (V_{DD}, EXSVDn) ≥ SVD detection voltage V_{SVD} or EXSVD detection voltage V_{SVD_EXT}
- When SVD3INTF.SVDDT bit = 1
  Power supply voltage (V_{DD}, EXSVDn) < SVD detection voltage V_{SVD} or EXSVD detection voltage V_{SVD_EXT}

Before reading the SVD3INTF.SVDDT bit, wait for at least SVD circuit enable response time after 1 is written to the SVD3CTL.MODEN bit (refer to "Supply Voltage Detector Characteristics, SVD circuit enable response time t_{SVDEN}" in the "Electrical Characteristics" chapter).
After the SVD3CTL.SVDC[4:0] bits setting value is altered to change the SVD detection voltage V_{SVD}/EXSVD detection voltage V_{SVD_EXT} when the SVD3CTL.MODEN bit = 1, wait for at least SVD circuit response time before reading the SVD3INTF.SVDDT bit (refer to "Supply Voltage Detector Characteristics, SVD circuit response time t_{SVD}" in the "Electrical Characteristics" chapter).

11.4.2 SVD3 Operations

Continuous operation mode

SVD3 operates in continuous operation mode by default (SVD3CTL.SVDMD[1:0] bits = 0x0). In this mode, SVD3 operates continuously while the SVD3CTL.MODEN bit is set to 1 and it keeps loading the detection results to the SVD3INTF.SVDDT bit. During this period, the current detection results can be obtained by reading the SVD3INTF.SVDDT bit as necessary. Furthermore, an interrupt (if the SVD3CTL.SVDRE[3:0] bits ≠ 0xa) or a reset (if the SVD3CTL.SVDRE[3:0] bits = 0xa) can be generated when the SVD3INTF.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

SVD3 operates in intermittent operation mode when the SVD3CTL.SVDMD[1:0] bits are set to 0x1 to 0x3. In this mode, SVD3 turns on at an interval set using the SVD3CTL.SVDMD[1:0] bits to perform detection operation and then it turns off while the SVD3CTL.MODEN bit is set to 1. During this period, the latest detection results can be obtained by reading the SVD3INTF.SVDDT bit as necessary. Furthermore, an interrupt or a reset can be generated when SVD3 has successively detected low power supply voltage the number of times specified by the SVD3CTL.SVDSC[1:0] bits.

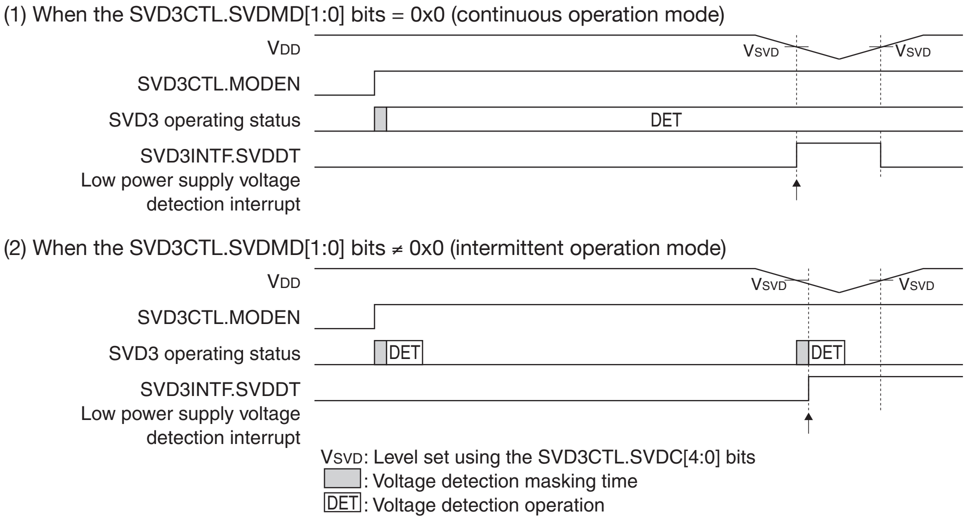


Figure 11.4.2.1 SVD3 Operations

11.5 SVD3 Interrupt and Reset

11.5.1 SVD3 Interrupt

Setting the SVD3CTL.SVDRE[3:0] bits to a value other than 0xa allows use of the low power supply voltage detection interrupt function.

Table 11.5.1.1 Low Power Supply Voltage Detection Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Low power supply voltage detection	SVD3INTF.SVDIF	In continuous operation mode When the SVD3INTF.SVDDT bit is 1 In intermittent operation mode When low power supply voltage is successively detected the specified number of times	Writing 1

SVD3 provides the interrupt enable bit (SVD3INTE.SVDIE bit) corresponding to the interrupt flag (SVD3INTF.SVDIF bit). An interrupt request is sent to the CPU only when the SVD3INTF.SVDIF bit is set while the interrupt is enabled by the SVD3INTE.SVDIE bit. For more information on interrupt control, refer to the “Interrupt” chapter.

Once the SVD3INTF.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}/EXSVD detection voltage V_{SVD_EXT}. An interrupt may occur due to a temporary power supply voltage drop, check the power supply voltage status by reading the SVD3INTF. SVDDT bit in the interrupt handler routine.

11.5.2 SVD3 Reset

Setting the SVD3CTL.SVDRE[3:0] bits to 0xa allows use of the SVD3 reset issuance function.
The reset issuing timing is the same as that of the SVD3INTF.SVDIF bit being set when a low voltage is detected.
After a reset has been issued, SVD3 enters continuous operation mode even if it was operating in intermittent operation mode, and continues operating. Issuing an SVD3 reset initializes the port assignment. However, when EXSVD*n* is being detected, the input of the port for the EXSVD*n* pin is sent to SVD3 so that SVD3 will continue the EXSVD*n* detection operation.
If the power supply voltage reverts to the normal level, the SVD3INTF.SVDDT bit goes 0 and the reset state is canceled. After that, SVD3 resumes operating in the operation mode set previously via the initialization routine.
During reset state, the SVD3 control bits are set as shown in Table 11.5.2.1.

Table 11.5.2.1 SVD3 Control Bits During Reset State

Control register	Control bit	Setting
SVD3CLK	DBRUN	Reset to the initial values.
	CLKDIV[2:0]	
	CLKSRC[1:0]	
SVD3CTL	VDSEL	The set value is retained.
	SVDSC[1:0]	Cleared to 0. (The set value becomes invalid as SVD3 enters continuous operation mode.)
	SVDC[4:0]	The set value is retained.
	SVDRE[3:0]	The set value (0xa) is retained.
	EXSEL	The set value is retained.
	SVDMD[1:0]	Cleared to 0 to set continuous operation mode.
	MODEN	The set value (1) is retained.
SVD3INTF	SVDIF	The status (1) before being reset is retained.
SVD3INTE	SVDIE	Cleared to 0.

11.6 Control Registers

SVD3 Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVD3CLK	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 SVD3 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 SVD3 operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of SVD3.

Table 11.6.1 Clock Source and Division Ratio Settings

SVD3CLK. CLKDIV[2:0] bits	SVD3CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x7, 0x6	Reserved	1/1	Reserved	1/1
0x5	1/512		1/512	
0x4	1/256		1/256	
0x3	1/128		1/128	
0x2	1/64		1/64	
0x1	1/32		1/32	
0x0	1/16		1/16	

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

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

SVD3 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVD3CTL	15	VDSEL	0	H1	R/WP	–
	14–13	SVDSC[1:0]	0x0	H0	R/WP	Writing takes effect when the SVD3CTL.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	EXSEL	0	H1	R/WP	
	2–1	SVDMD[1:0]	0x0	H0	R/WP	
	0	MODEN	0	H1	R/WP	

Bit 15 VDSEL

This bit selects the power supply voltage to be detected by SVD3.

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

0 (R/WP): V_{DD}

Bits 14–13 SVDSC[1:0]

These bits set the condition to generate an interrupt/reset (number of successive low voltage detections) in intermittent operation mode (SVD3CTL.SVDMD[1:0] bits = 0x1 to 0x3).

Table 11.6.2 Interrupt/Reset Generating Condition in Intermittent Operation Mode

SVD3CTL.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 (SVD3CTL.SVDMD[1:0] bits = 0x0).

Bits 12–8 SVDC[4:0]

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

Table 11.6.3 Setting of SVD Detection Voltage V_{SVD}/EXSVD Detection Voltage V_{SVD_EXT}

SVD3CTL.SVDC[4:0] bits	SVD detection voltage V_{SVD}/ EXSVD detection voltage V_{SVD_EXT} [V]
0x1f	High
0x1e	↑
0x1d	
:	
0x02	
0x01	↓
0x00	Low

For the configurable range and voltage values, refer to "Supply Voltage Detector Characteristics, SVD detection voltage V_{SVD}/EXSVD detection voltage V_{SVD_EXT}" 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 SVD3 reset issuance function, refer to "SVD3 Reset."

Bit 3 EXSEL

This bit selects the voltage to be detected when the SVD3CTL.VDSEL bit = 1.
1 (R/WP): EXSVD1
0 (R/WP): EXSVD0

Bits 2–1 SVDMD[1:0]

These bits select intermittent operation mode and its detection cycle.

Table 11.6.4 Intermittent Operation Mode Detection Cycle Selection

SVD3CTL.SVDMD[1:0] bits	Operation mode (detection cycle)
0x3	Intermittent operation mode (CLK_SVD3/512)
0x2	Intermittent operation mode (CLK_SVD3/256)
0x1	Intermittent operation mode (CLK_SVD3/128)
0x0	Continuous operation mode

For more information on intermittent and continuous operation modes, refer to "SVD3 Operations."

Bit 0 MODEN

This bit enables/disables for the SVD3 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 SVD3CTL.MODEN bit resets the SVD3 hardware. However, the register values set and the interrupt flag are not cleared. The SVD3CTL.MODEN bit is actually set to 0 after this processing has finished. If 1 is written to the SVD3CTL.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 SVD3 internal circuit is initialized if the SVD3CTL.SVDSC[1:0] bits, SVD3CTL.SVDRE[3:0] bits, or SVD3CTL.SVDMD[1:0] bits are altered while SVD3 is in operation after 1 is written to the SVD3CTL.MODEN bit.

SVD3 Status and Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVD3INTF	15–9	–	0x00	–	R	–
	8	SVDDT	x	–	R	
	7–1	–	0x00	–	R	
	0	SVDIF	0	H1	R/W	Cleared by writing 1.

Bits 15–9 Reserved

Bit 8 SVDDT

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

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 SVD3 internal circuit is initialized if the interrupt flag is cleared while SVD3 is in operation after 1 is written to the SVD3CTL.MODEN bit.

SVD3 Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SVD3INTE	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 SVD3CTL.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.

12 16-bit Timers (T16)

12.1 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 12.1.1 shows the configuration of a T16 channel.

Table 12.1.1 T16 Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	8 channels (Ch.0–Ch.7)		
Event counter function	Not supported (No EXCL*m* pins are provided.)		
Peripheral clock output (Outputs the counter underflow signal.)	Ch.1 → Synchronous serial interface Ch.0 master clock Ch.2 → Quad synchronous serial interface Ch.0 master clock Ch.5 → Synchronous serial interface Ch.2 master clock Ch.6 → Synchronous serial interface Ch.1 master clock Ch.7 → 12-bit A/D converter trigger signal		

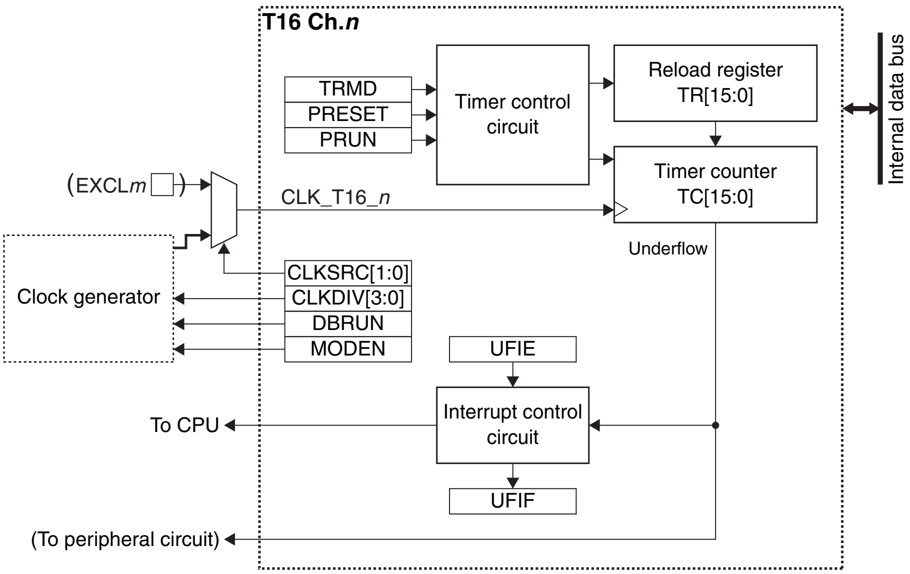


Figure 12.1.1 Configuration of a T16 Channel

12.2 Input Pin

Table 12.2.1 shows the T16 input pin.

Table 12.2.1 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. The EXCL*m* signal can be input through the chattering filter. For more information, refer to the “I/O Ports” chapter.

12.3 Clock Settings

12.3.1 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)

12.3.2 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.

12.3.3 Clock Supply During Debugging

The CLK_T16_*n* supply during debugging 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 state if the T16_*n*CLK.DBRUN bit = 0. After the CPU returns to normal operation, 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 the debug state 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 a debug state.

12.3.4 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.

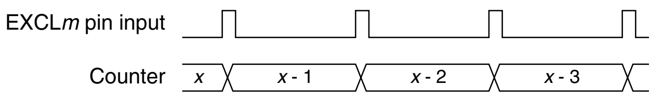


Figure 12.3.4.1 Count Down Timing

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

12.4 Operations

12.4.1 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)

12.4.2 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:

$$T = \frac{TR + 1}{f_{CLK\_T16\_n}} \qquad f_T = \frac{f_{CLK\_T16\_n}}{TR + 1} \qquad \text{(Eq. 12.1)}$$

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]

12.4.3 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.

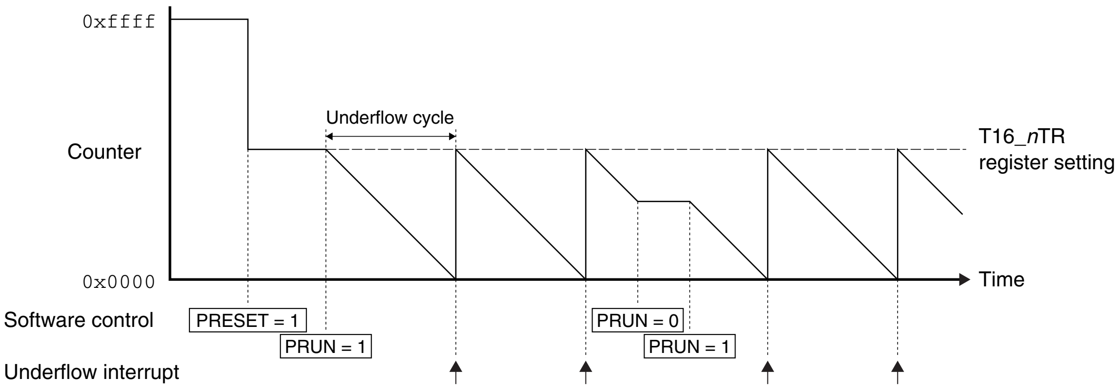


Figure 12.4.3.1 Count Operations in Repeat Mode

12.4.4 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.

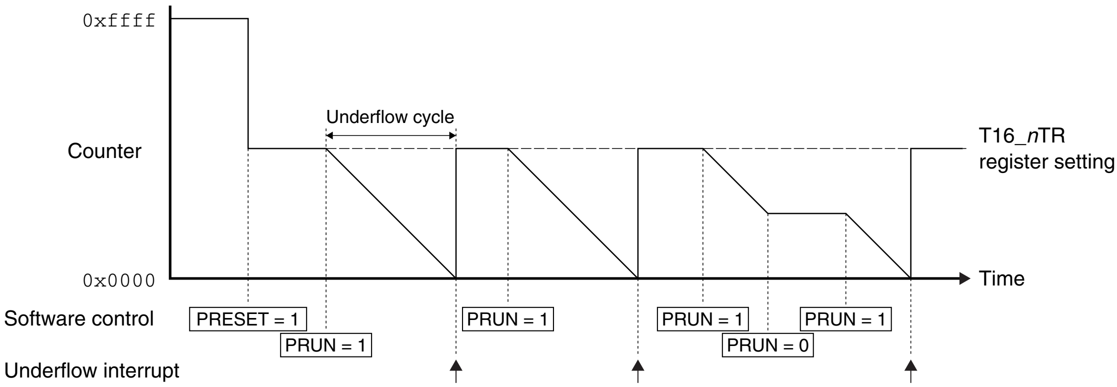


Figure 12.4.4.1 Count Operations in One-shot Mode

12.4.5 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.

12.5 Interrupt

Each T16 channel has a function to generate the interrupt shown in Table 12.5.1.

Table 12.5.1 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 CPU 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" chapter.

12.6 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 during debugging or not.
1 (R/W): Clock supplied during debugging
0 (R/W): No clock supplied during debugging

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 12.6.1 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.

13 UART (UART3)

13.1 Overview

The UART3 is an asynchronous serial interface. The features of the UART3 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.
- Can issue a DMA transfer request when a receive buffer one byte full or a transmit buffer empty occurs.
- Input pin can be pulled up with an internal resistor.
- The output pin is configurable as an open-drain output.
- Provides the carrier modulation output function.

Figure 13.1.1 shows the UART3 configuration.

Table 13.1.1 UART3 Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	3 channels (Ch.0 to Ch.2)		

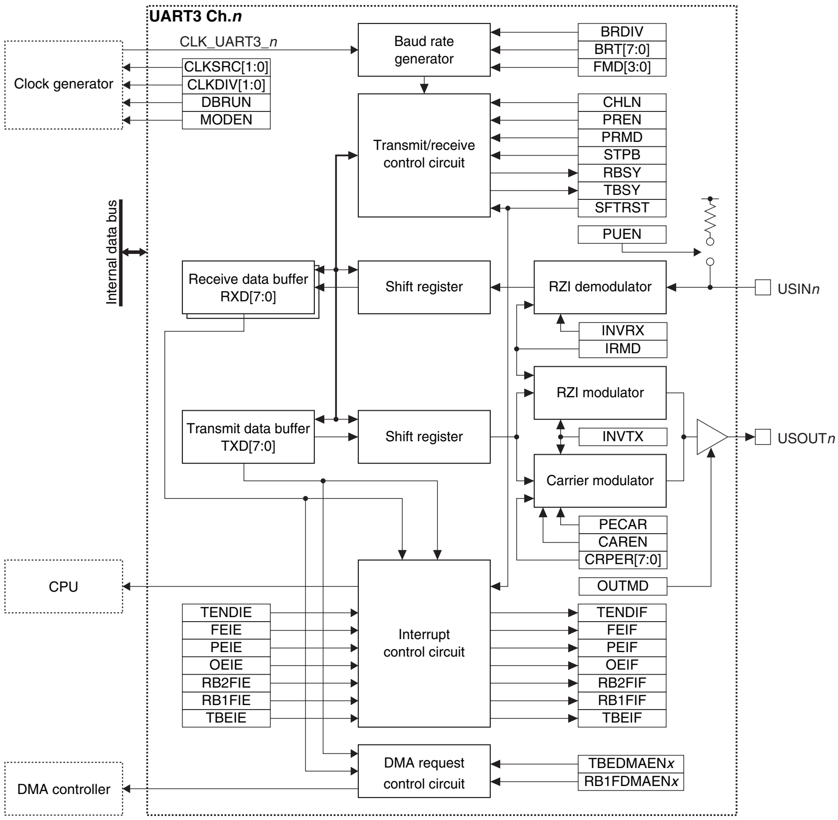


Figure 13.1.1 UART3 Configuration

13.2 Input/Output Pins and External Connections

13.2.1 List of Input/Output Pins

Table 13.2.1.1 lists the UART3 pins.

Table 13.2.1.1 List of UART3 Pins

Pin name	I/O*	Initial status*	Function
USIN*n*	I	I (Hi-Z)	UART3 Ch.*n* data input pin
USOUT*n*	O	O (High)	UART3 Ch.*n* data output pin

* Indicates the status when the pin is configured for the UART3.

If the port is shared with the UART3 pin and other functions, the UART3 input/output function must be assigned to the port before activating the UART3. For more information, refer to the “I/O Ports” chapter.

13.2.2 External Connections

Figure 13.2.2.1 shows a connection diagram between the UART3 in this IC and an external UART device.

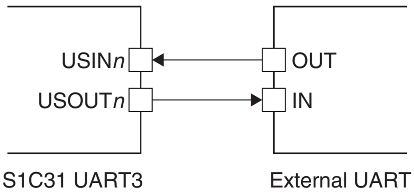


Figure 13.2.2.1 Connections between UART3 and an External UART Device

13.2.3 Input Pin Pull-Up Function

The UART3 includes a pull-up resistor for the USIN*n* pin. Setting the UART3_*n*MOD.PUEN bit to 1 enables the resistor to pull up the USIN*n* pin.

13.2.4 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 UART3_*n*MOD.OUTMD bit to 1.

13.2.5 Input/Output Signal Inverting Function

The UART3 can invert the signal polarities of the USIN*n* pin input and the USOUT*n* pin output by setting the UART3_*n*MOD.INVRX bit and the UART3_*n*MOD.INVTX bit, respectively, to 1.

Note: Unless otherwise specified, this chapter shows input/output signals with non-inverted waveforms (UART3_*n*MOD.INVRX bit = 0, UART3_*n*MOD.INVTX bit =0).

13.3 Clock Settings

13.3.1 UART3 Operating Clock

When using the UART3 Ch.*n*, the UART3 Ch.*n* operating clock CLK_UART3_*n* must be supplied to the UART3 Ch.*n* from the clock generator. The CLK_UART3_*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 UART3_*n*CLK register bits:
   - UART3_*n*CLK.CLKSRC[1:0] bits (Clock source selection)
   - UART3_*n*CLK.CLKDIV[1:0] bits (Clock division ratio selection = Clock frequency setting)

   The UART3 operating clock should be selected so that the baud rate generator will be configured easily.

13.3.2 Clock Supply in SLEEP Mode

When using the UART3 during SLEEP mode, the UART3 operating clock CLK_UART3_*n* must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_UART3_*n* clock source.

13.3.3 Clock Supply During Debugging

The CLK_UART3_*n* supply during debugging should be controlled using the UART3_*n*CLK.DBRUN bit.
The CLK_UART3_*n* supply to the UART3 Ch.*n* is suspended when the CPU enters debug state if the UART3_*n*CLK.DBRUN bit = 0. After the CPU returns to normal mode, the CLK_UART3_*n* supply resumes. Although the UART3 Ch.*n* stops operating when the CLK_UART3_*n* supply is suspended, the output pin and registers retain the status before the debug state was entered. If the UART3_*n*CLK.DBRUN bit = 1, the CLK_UART3_*n* supply is not suspended and the UART3 Ch.*n* will keep operating in a debug state.

13.3.4 Baud Rate Generator

The UART3 includes a baud rate generator to generate the transfer (sampling) clock. The transfer rate is determined by the UART3_*n*MOD.BRDIV, UART3_*n*BR.BRT[7:0], and UART3_*n*BR.FMD[3:0] bit settings. Use the following equations to calculate the setting values for obtaining the desired transfer rate.

$$\text{bps} = \frac{\text{CLK\_UART3}}{\frac{\text{BRT} + 1}{\text{BRDIV}} + \text{FMD}} \qquad \text{BRT} = \text{BRDIV} \times \left(\frac{\text{CLK\_UART3}}{\text{bps}} - \text{FMD}\right) - 1 \qquad \text{(Eq. 13.1)}$$

Where
- bps: Transfer rate [bit/s]
- CLK_UART3: UART3 operating clock frequency [Hz]
- BRDIV: Baud rate division ratio (1/16 or 1/4) * Selected by the UART3_*n*MOD.BRDIV bit
- BRT: UART3_*n*BR.BRT[7:0] setting value (0 to 255)
- FMD: UART3_*n*BR.FMD[3:0] setting value (0 to 15)

For the transfer rate range configurable in the UART3, refer to “UART Characteristics, Transfer baud rates U_{BRT1} and U_{BRT2}” in the “Electrical Characteristics” chapter.

13.4 Data Format

The UART3 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 UART3_*n*MOD.CHLN bit, the data length can be set to seven bits (UART3_*n*MOD.CHLN bit = 0) or eight bits (UART3_*n*MOD.CHLN bit = 1).

Stop bit length

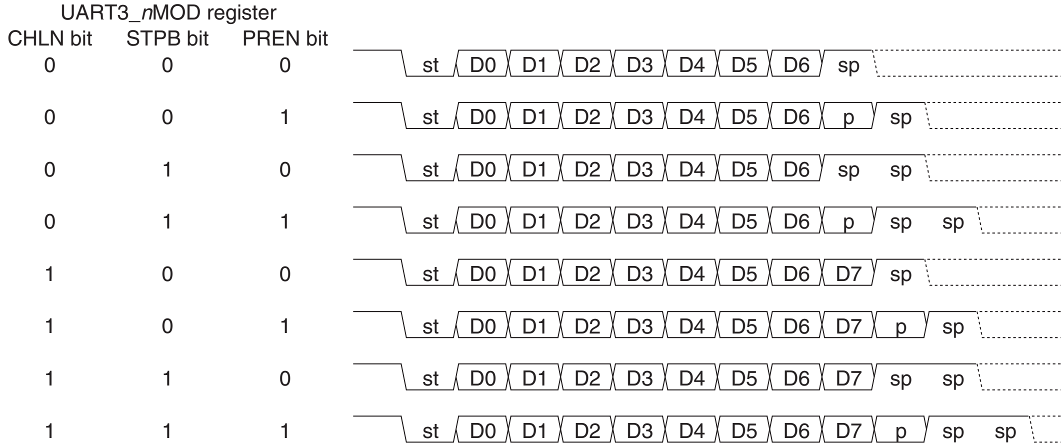
With the UART3_*n*MOD.STPB bit, the stop bit length can be set to one bit (UART3_*n*MOD.STPB bit = 0) or two bits (UART3_*n*MOD.STPB bit = 1).

Parity function

The parity function is configured using the UART3_*n*MOD.PREN and UART3_*n*MOD.PRMD bits.

Table 13.4.1 Parity Function Setting

UART3_*n*MOD.PREN bit	UART3_*n*MOD.PRMD bit	Parity function
1	1	Odd parity
1	0	Even parity
0	*	Non parity



st: start bit, sp: stop bit, p: parity bit

Figure 13.4.1 Data Format

13.5 Operations

13.5.1 Initialization

The UART3 Ch.*n* should be initialized with the procedure shown below.

1. Assign the UART3 Ch.*n* input/output function to the ports. (Refer to the "I/O Ports" chapter.)
2. Set the UART3_*n*CLK.CLKSRC[1:0] and UART3_*n*CLK.CLKDIV[1:0] bits. (Configure operating clock)
3. Configure the following UART3_*n*MOD register bits:
   - UART3_*n*MOD.BRDIV bit (Select baud rate division ratio (1/16 or 1/4))
   - UART3_nMOD.INVRX bit (Enable/disable USIN*n* input signal inversion)
   - UART3_nMOD.INVTX bit (Enable/disable USOUT*n* output signal inversion)
   - UART3_*n*MOD.PUEN bit (Enable/disable USIN*n* pin pull-up)
   - UART3_*n*MOD.OUTMD bit (Enable/disable USOUT*n* pin open-drain output)
   - UART3_*n*MOD.IRMD bit (Enable/disable IrDA interface)
   - UART3_*n*MOD.CHLN bit (Set data length (7 or 8 bits))
   - UART3_*n*MOD.PREN bit (Enable/disable parity function)
   - UART3_*n*MOD.PRMD bit (Select parity mode (even or odd))
   - UART3_*n*MOD.STPB bit (Set stop bit length (1 or 2 bits))
   - UART3_*n*MOD.CAREN bit (Enable/disable carrier modulation function)
   - UART3_nMOD.PECAR bit (Select carrier modulation period (H data period/L data period))
4. Set the UART3_*n*BR.BRT[7:0] and UART3_*n*BR.FMD[3:0] bits. (Set transfer rate)
5. Set the UART3_*n*CAWF.CRPER[7:0] bits. (Set carrier cycle)
6. Set the following UART3_*n*CTL register bits:
   - Set the UART3_*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the UART3_*n*CTL.MODEN bit to 1. (Enable UART3 Ch.*n* operations)
7. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the UART3_*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the UART3_*n*INTE register to 1. * (Enable interrupts)

   * The initial value of the UART3_*n*INTF.TBEIF bit is 1, therefore, an interrupt will occur immediately after the UART3_*n*INTE.TBEIE bit is set to 1.
8. Configure the DMA controller and set the following UART3 control bits when using DMA transfer:
   - Write 1 to the DMA transfer request enable bits in the UART3_*n*TBEDMAEN and UART3_*n*RB1FDMAEN registers. (Enable DMA transfer requests)

13.5.2 Data Transmission

A data sending procedure and the UART3 Ch.*n* operations 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. Check to see if the UART3_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
2. Write transmit data to the UART3_*n*TXD register.
3. Wait for a UART3 interrupt when using the interrupt.
4. Repeat Steps 1 to 3 (or 1 and 2) until the end of transmit data.

UART3 data sending operations

The UART3 Ch.*n* starts data sending operations when transmit data is written to the UART3_*n*TXD register.
The transmit data in the UART3_*n*TXD register is automatically transferred to the shift register and the UART3_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
The USOUT*n* pin outputs a start bit and the UART3_*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 UART3_*n*TXD register after making sure the UART3_*n*INTF.TBEIF bit is set to 1.

If no transmit data remains in the UART3_*n*TXD register after the stop bit has been output from the USOUT*n* pin, the UART3_*n*INTF.TBSY bit is cleared to 0 and the UART3_*n*INTF.TENDIF bit is set to 1 (transmission completed).

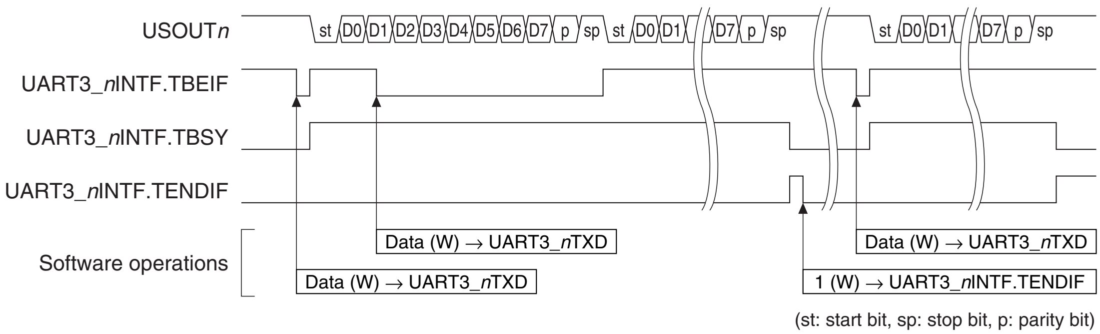


Figure 13.5.2.1 Example of Data Sending Operations

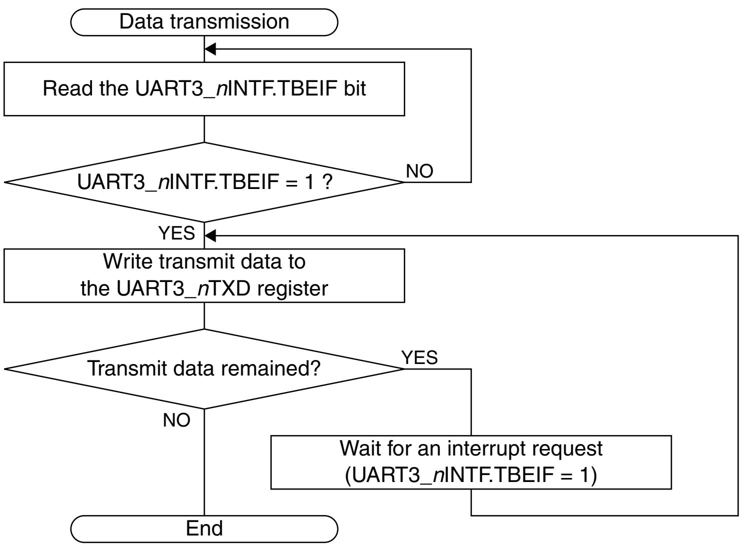


Figure 13.5.2.2 Data Transmission Flowchart

Data transmission using DMA

By setting the UART3_*n*TBEDMAEN.TBEDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and transmit data is transferred from the specified memory to the UART3_*n*TXD register via DMA Ch.*x* when the UART3_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty). This automates the data sending procedure described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance so that transmit data will be transferred to the UART3_*n*TXD register. For more information on DMA, refer to the “DMA Controller” chapter.

Table 13.5.2.1 DMA Data Structure Configuration Example (for Data Transmission)

Item		Setting example
End pointer	Transfer source	Memory address in which the last transmit data is stored
	Transfer destination	UART3_*n*TXD register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x0 (byte)
	src_inc	0x0 (+1)
	src_size	0x0 (byte)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

13.5.3 Data Reception

A data receiving procedure and the UART3 Ch.*n* operations are shown below. Figures 13.5.3.1 and 13.5.3.2 show a timing chart and flowcharts, respectively.

Data receiving procedure (read by one byte)

1. Wait for a UART3 interrupt when using the interrupt.
2. Check to see if the UART3_*n*INTF.RB1FIF bit is set to 1 (receive buffer one byte full).
3. Read the received data from the UART3_*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 UART3 interrupt when using the interrupt.
2. Check to see if the UART3_*n*INTF.RB2FIF bit is set to 1 (receive buffer two bytes full).
3. Read the received data from the UART3_*n*RXD register twice.
4. Repeat Steps 1 to 3 (or 2 and 3) until the end of data reception.

UART3 data receiving operations

The UART3 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 UART3_*n*INTF.RBSY bit is set to 1 when the start bit is detected.

The UART3_*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 UART3_*n*INTF.RB1FIF bit to 1 (receive buffer one byte full). If the second data is received without reading the first data, the UART3_*n*INTF.RB2FIF bit is set to 1 (receive buffer two bytes full).

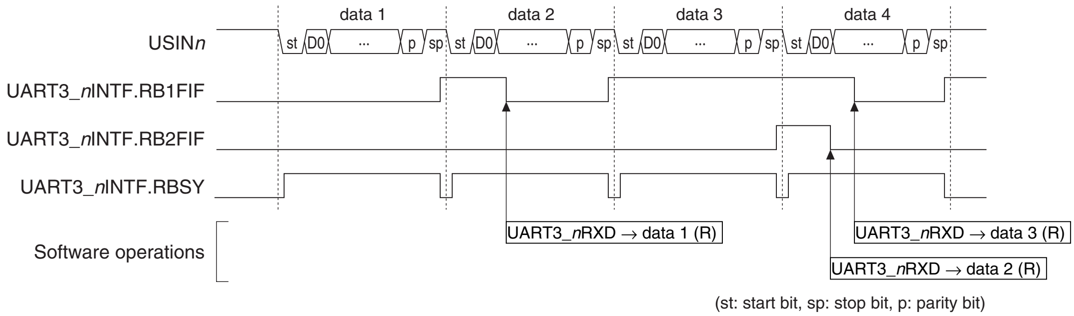


Figure 13.5.3.1 Example of Data Receiving Operations

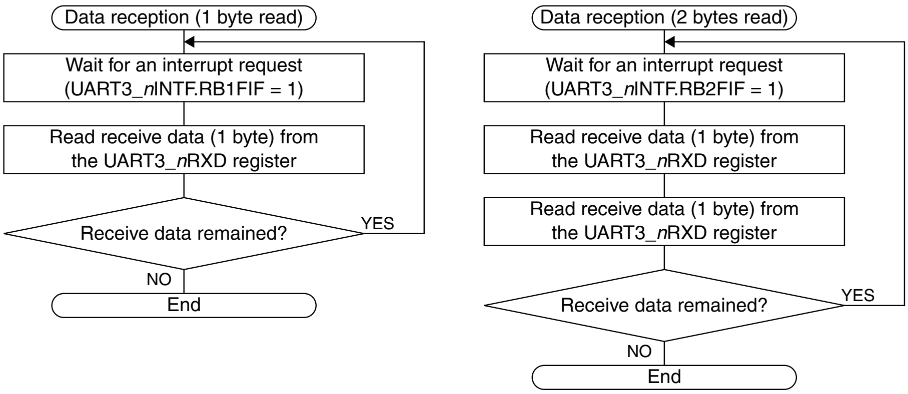


Figure 13.5.3.2 Data Reception Flowcharts

Data reception using DMA

By setting the UART3_*n*RB1FDMAEN.RB1FDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and the received data is transferred from the UART3_*n*RXD register to the specified memory via DMA Ch.*x* when the UART3_*n*INTF.RB1FIF bit is set to 1 (receive buffer one byte full).

This automates the procedure (read by one byte) described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 13.5.3.1 DMA Data Structure Configuration Example (for Data Reception)

Item		Setting example
End pointer	Transfer source	UART3_*n*RXD register address
	Transfer destination	Memory address to which the last received data is stored
Control data	dst_inc	0x0 (+1)
	dst_size	0x0 (byte)
	src_inc	0x3 (no increment)
	src_size	0x0 (byte)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

13.5.4 IrDA Interface

This UART3 includes an RZI modulator/demodulator circuit enabling implementation of IrDA 1.0-compatible infrared communication function simply by adding simple external circuits.

Set the UART3_*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.

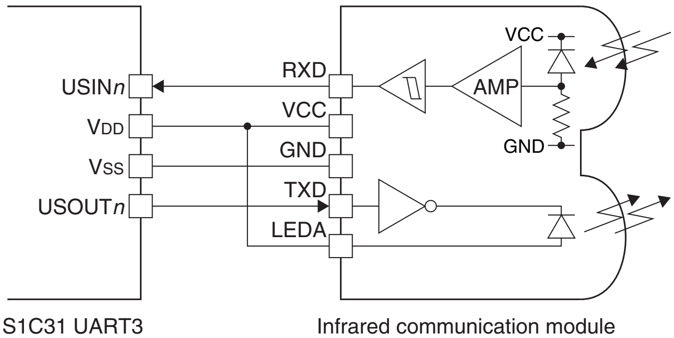


Figure 13.5.4.1 Example of Connections with an Infrared Communication Module

The transmit data output from the UART3 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.

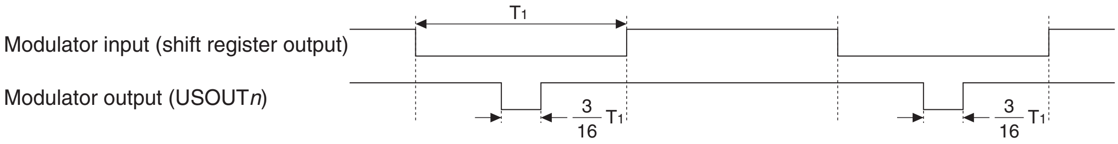


Figure 13.5.4.2 IrDA Transmission Signal Waveform

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.

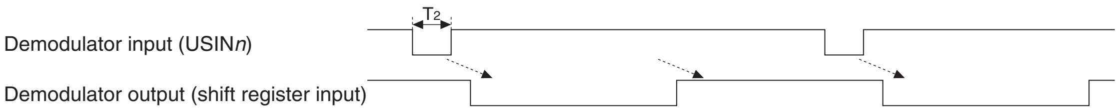


Figure 13.5.4.3 IrDA Receive Signal Waveform

Notes:
- Set the baud rate division ratio to 1/16 when using the IrDA interface function.
- The low pulse width (T_2) of the IrDA signal input must be CLK_UART3_*n* × 3 cycles or longer.

13.5.5 Carrier Modulation

The UART3 has a carrier modulation function.

Writing 1 to the UART3_*n*MOD.CAREN bit enables the carrier modulation function allowing carrier modulation waveforms to be output according to the UART3_*n*MOD.PECAR bit setting. Data transmit control is identical to that for normal interface even in this case.

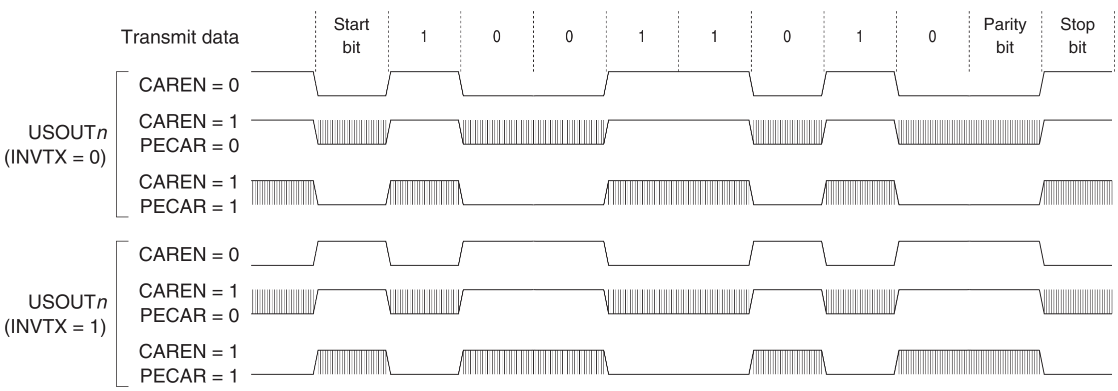


Figure 13.5.5.1 Carrier Modulation Waveform
(UART3_*n*MOD.CHLN = 1, UART3_*n*MOD.STPB = 0, UART3_*n*MOD.PREN = 1)

The carrier modulation output frequency is determined by the UART3_*n*CAWF.CRPER[7:0] bit settings. Use the following equations to calculate the setting values for obtaining the desired frequency.

$$\text{Carrier modulation output frequency} = \frac{\text{CLK\_UART3}}{(\text{CRPER} + 1) \times 2} \text{ [Hz]} \qquad \text{(Eq. 13.2)}$$

Where
CLK_UART3: UART3 operating clock frequency [Hz]
CRPER: UART3_*n*CAWF.CRPER[7:0] setting value (0 to 255)

13.6 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.

13.6.1 Framing Error

The UART3 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 UART3_*n*INTF.FEIF bit (framing error interrupt flag) is set to 1 when the data becomes ready to read from the UART3_*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.

13.6.2 Parity Error

If the parity function is enabled, a parity check is performed when data is received. The UART3 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 UART3_*n*INTF.PEIF bit (parity error interrupt flag) is set to 1 when the data becomes ready to read from the UART3_*n*RXD register (see the Note on framing error).

13.6.3 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 UART3_*n*INTF.OEIF bit (overrun error interrupt flag) is set to 1.

13.7 Interrupts

The UART3 has a function to generate the interrupts shown in Table 13.7.1.

Table 13.7.1 UART3 Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
End of transmission	UART3_*n*INTF.TENDIF	When the UART3_*n*INTF.TBEIF bit = 1 after the stop bit has been sent	Writing 1 or software reset
Framing error	UART3_*n*INTF.FEIF	Refer to the “Receive Errors.”	Writing 1, reading received data that encountered an error, or software reset
Parity error	UART3_*n*INTF.PEIF	Refer to the “Receive Errors.”	Writing 1, reading received data that encountered an error, or software reset
Overrun error	UART3_*n*INTF.OEIF	Refer to the “Receive Errors.”	Writing 1 or software reset
Receive buffer two bytes full	UART3_*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	UART3_*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	UART3_*n*INTF.TBEIF	When transmit data written to the transmit data buffer is transferred to the shift register	Writing transmit data

The UART3 provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU 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” chapter.

13.8 DMA Transfer Requests

The UART3 has a function to generate DMA transfer requests from the causes shown in Table 13.8.1.

Table 13.8.1 DMA Transfer Request Causes of UART3

Cause to request DMA transfer	DMA transfer request flag	Set condition	Clear condition
Receive buffer one byte full	Receive buffer one byte full flag (UART3_*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	Transmit buffer empty flag (UART3_*n*INTF.TBEIF)	When transmit data written to the transmit data buffer is transferred to the shift register	Writing transmit data

The UART3 provides DMA transfer request enable bits corresponding to each DMA transfer request flag shown above for the number of DMA channels. A DMA transfer request is sent to the pertinent channel of the DMA controller only when the DMA transfer request flag, of which DMA transfer has been enabled by the DMA transfer request enable bit, is set. The DMA transfer request flag also serves as an interrupt flag, therefore, both the DMA transfer request and the interrupt cannot be enabled at the same time. After a DMA transfer has completed, disable the DMA transfer to prevent unintended DMA transfer requests from being issued. For more information on the DMA control, refer to the “DMA Controller” chapter.

13.9 Control Registers

UART3 Ch.*n* Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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 UART3 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 UART3 operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of the UART3.

Table 13.9.1 Clock Source and Division Ratio Settings

UART3_*n*CLK. CLKDIV[1:0] bits	UART3_*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 UART3_*n*CLK register settings can be altered only when the UART3_*n*CTL.MODEN bit = 0.

UART3 Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*n*MOD	15–13	–	0x0	–	R	–
	12	PECAR	0	H0	R/W	
	11	CAREN	0	H0	R/W	
	10	BRDIV	0	H0	R/W	
	9	INVRX	0	H0	R/W	
	8	INVTX	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 15–13 Reserved

Bit 12 PECAR

This bit selects the carrier modulation period.

1 (R/W): Carrier modulation during H data period
0 (R/W): Carrier modulation during L data period

Bit 11 CAREN

This bit enables the carrier modulation function.

1 (R/W): Enable carrier modulation function
0 (R/W): Disable carrier modulation function

Bit 10 BRDIV

This bit sets the UART3 operating clock division ratio for generating the transfer (sampling) clock using the baud rate generator.

1 (R/W): 1/4
0 (R/W): 1/16

Bit 9 INVRX

This bit enables the USIN*n* input inverting function.

1 (R/W): Enable input inverting function
0 (R/W): Disable input inverting function

Bit 8 INVTX

This bit enables the USOUT*n* output inverting function.

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

Notes: • The UART3_*n*MOD register settings can be altered only when the UART3_*n*CTL.MODEN bit = 0.

• Do not set both the UART3_*n*MOD.IRMD and UART3_*n*MOD.CAREN bits simultaneously.

UART3 Ch.*n* Baud–Rate Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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 UART3 transfer rate. For more information, refer to “Baud Rate Generator.”

Notes: • The UART3_*n*BR register settings can be altered only when the UART3_*n*CTL.MODEN bit = 0.

• Do not set the UART3_*n*BR.FMD[3:0] bits to a value other than 0 to 3 when the UART3_*n*MOD.BRDIV bit = 1.

UART3 Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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 UART3.

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 UART3 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 UART3 operations.

1 (R/W): Enable UART3 operations (The operating clock is supplied.)

0 (R/W): Disable UART3 operations (The operating clock is stopped.)

Note: If the UART3_*n*CTL.MODEN bit is altered from 1 to 0 while sending/receiving data, the data being sent/received cannot be guaranteed. When setting the UART3_*n*CTL.MODEN bit to 1 again after that, be sure to write 1 to the UART3_*n*CTL.SFTRST bit as well.

UART3 Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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 UART3_*n*INTF.TBEIF bit is set to 1 before writing data.

UART3 Ch.*n* Receive Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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.

UART3 Ch.*n* Status and Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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 UART3_*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 UART3_*n*RXD register.
	1	RB1FIF	0	H0/S0	R	
	0	TBEIF	1	H0/S0	R	Cleared by writing to the UART3_*n*TXD register.

Bits 15–10 Reserved

Bit 9 RBSY

This bit indicates the receiving status. (See Figure 13.5.3.1.)

1 (R): During receiving

0 (R): Idle

Bit 8 TBSY

This bit indicates the sending status. (See Figure 13.5.2.1.)

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 UART3 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:
UART3_*n*INTF.TENDIF bit: End-of-transmission interrupt
UART3_*n*INTF.FEIF bit: Framing error interrupt
UART3_*n*INTF.PEIF bit: Parity error interrupt
UART3_*n*INTF.OEIF bit: Overrun error interrupt
UART3_*n*INTF.RB2FIF bit: Receive buffer two bytes full interrupt
UART3_*n*INTF.RB1FIF bit: Receive buffer one byte full interrupt
UART3_*n*INTF.TBEIF bit: Transmit buffer empty interrupt

UART3 Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*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 UART3 interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
UART3_*n*INTE.TENDIE bit: End-of-transmission interrupt
UART3_*n*INTE.FEIE bit: Framing error interrupt
UART3_*n*INTE.PEIE bit: Parity error interrupt
UART3_*n*INTE.OEIE bit: Overrun error interrupt
UART3_*n*INTE.RB2FIE bit: Receive buffer two bytes full interrupt
UART3_*n*INTE.RB1FIE bit: Receive buffer one byte full interrupt
UART3_*n*INTE.TBEIE bit: Transmit buffer empty interrupt

UART3 Ch.*n* Transmit Buffer Empty DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*n*T BEDMAEN	15–0	TBEDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 TBEDMAEN[15:0]

These bits enable the UART3 to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when a transmit buffer empty state has occurred.

1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

UART3 Ch.*n* Receive Buffer One Byte Full DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*n* RB1FDMAEN	15–0	RB1FDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 RB1FDMAEN[15:0]

These bits enable the UART3 to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when a receive buffer one byte full state has occurred.

1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

UART3 Ch.*n* Carrier Waveform Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
UART3_*n*CAWF	15–8	–	0x00	–	R	–
	7–0	CRPER[7:0]	0x00	H0	R/W	

Bits 15–8 Reserved

Bits 7–0 CRPER[7:0]

These bits set the carrier modulation output frequency. For more information, refer to “Carrier Modulation.”

14 Synchronous Serial Interface (SPIA)

14.1 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.
- Can issue a DMA transfer request when a receive buffer full or a transmit buffer empty occurs.
- 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 14.1.1 shows the SPIA configuration.

Table 14.1.1 SPIA Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	3 channels (Ch.0 to Ch.2)		
Internal clock input	Ch.0 ← 16-bit timer Ch.1 Ch.1 ← 16-bit timer Ch.6 Ch.2 ← 16-bit timer Ch.5		

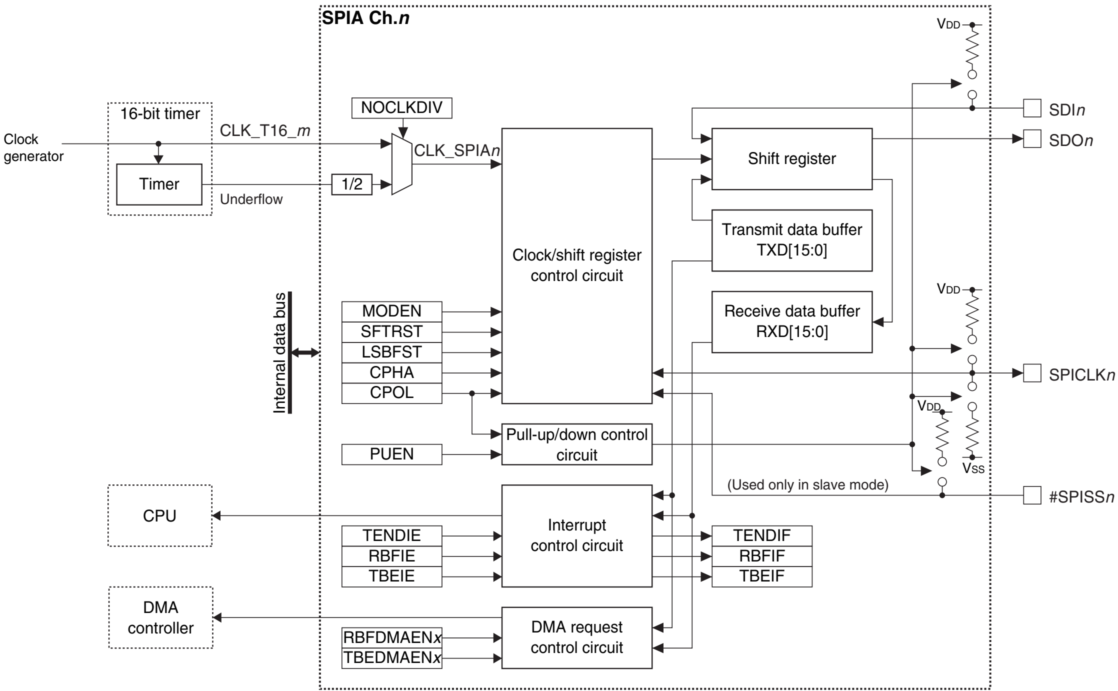


Figure 14.1.1 SPIA Configuration

14.2 Input/Output Pins and External Connections

14.2.1 List of Input/Output Pins

Table 14.2.1.1 lists the SPIA pins.

Table 14.2.1.1 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.

14.2.2 External Connections

SPIA operates in master mode or slave mode. Figures 14.2.2.1 and 14.2.2.2 show connection diagrams between SPIA in each mode and external SPI devices.

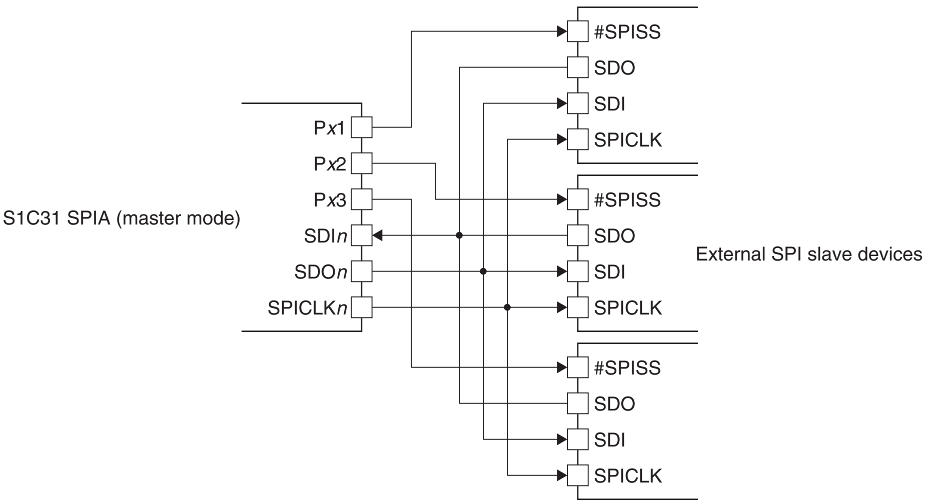


Figure 14.2.2.1 Connections between SPIA in Master Mode and External SPI Slave Devices

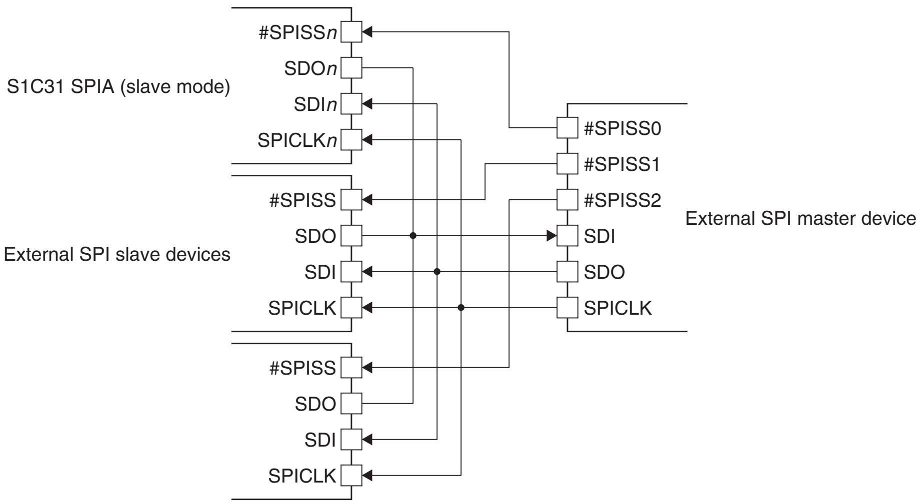


Figure 14.2.2.2 Connections between SPIA in Slave Mode and External SPI Master Device

14.2.3 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 14.2.3.1.

Table 14.2.3.1 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.

14.2.4 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 14.2.4.1. This function is enabled by setting the SPIA_*n*MOD.PUEN bit to 1.

Table 14.2.4.1 Pull-Up or Pull-Down of Input Pins

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

14.3 Clock Settings

14.3.1 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 SPIA_*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 SPIA_*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.

$$f_{SPICLK} = \frac{f_{CLK\_SPIA}}{2 \times (RLD + 1)} \qquad RLD = \frac{f_{CLK\_SPIA}}{f_{SPICLK} \times 2} - 1 \qquad \text{(Eq. 14.1)}$$

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 SPIA_*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.

14.3.2 Clock Supply During Debugging

In master mode, the operating clock supply during debugging 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 state if the T16_*m*CLK.DBRUN bit = 0. After the CPU returns to normal operation, 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 the debug state 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 a debug state.

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 state or normal operation state.

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

The SPICLK*n* phase and polarity can be configured separately using the SPIA_*n*MOD.CPHA bit and the SPIA_*n*MOD.CPOL bit, respectively. Figure 14.3.3.1 shows the clock waveform and data input/output timing in each setting.

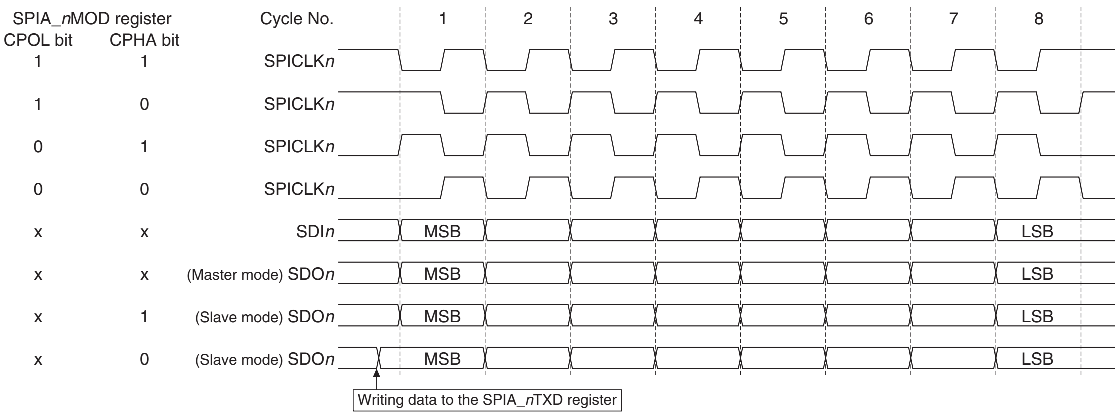


Figure 14.3.3.1 SPI Clock Phase and Polarity (SPIA_*n*MOD.LSBFST bit = 0, SPIA_*n*MOD.CHLN[3:0] bits = 0x7)

14.4 Data Format

The SPIA data length can be selected from 2 bits to 16 bits by setting the SPIA_*n*MOD.CHLN[3:0] bits. The input/output permutation is configurable to MSB first or LSB first using the SPIA_*n*MOD.LSBFST bit. Figure 14.4.1 shows a data format example when the SPIA_*n*MOD.CHLN[3:0] bits = 0x7, the SPIA_*n*MOD.CPOL bit = 0 and the SPIA_*n*MOD.CPHA bit = 0.

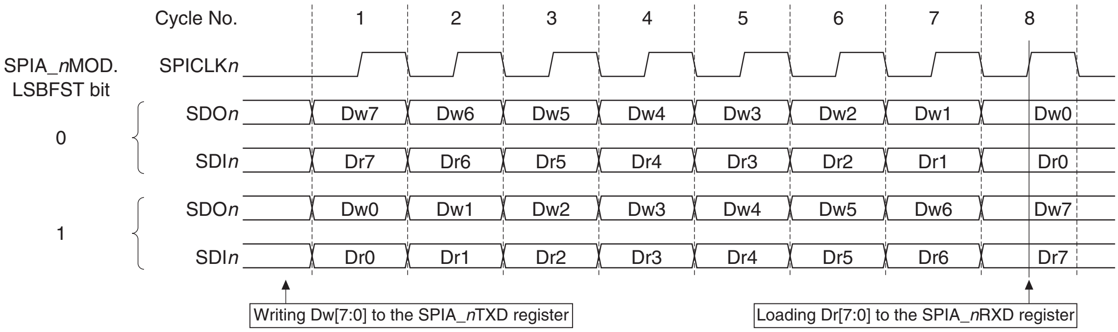


Figure 14.4.1 Data Format Selection Using the SPIA_*n*MOD.LSBFST Bit
(SPIA_*n*MOD.CHLN[3:0] bits = 0x7, SPIA_*n*MOD.CPOL bit = 0, SPIA_*n*MOD.CPHA bit = 0)

14.5 Operations

14.5.1 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 SPIA_*n*MOD register bits:
   - SPIA_*n*MOD.PUEN bit (Enable input pin pull-up/down)
   - SPIA_*n*MOD.NOCLKDIV bit (Select master mode operating clock)
   - SPIA_*n*MOD.LSBFST bit (Select MSB first/LSB first)
   - SPIA_*n*MOD.CPHA bit (Select clock phase)
   - SPIA_*n*MOD.CPOL bit (Select clock polarity)
   - SPIA_*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 SPIA_*n*CTL register bits:
   - Set the SPIA_*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the SPIA_*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 SPIA_*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the SPIA_*n*INTE register to 1. * (Enable interrupts)

   * The initial value of the SPIA_*n*INTF.TBEIF bit is 1, therefore, an interrupt will occur immediately after the SPIA_*n*INTE.TBEIE bit is set to 1.
6. Configure the DMA controller and set the following SPIA control bits when using DMA transfer:
   - Write 1 to the DMA transfer request enable bits
     in the SPIA_*n*TBEDMAEN and SPIA_*n*RBFDMAEN registers. (Enable DMA transfer requests)

14.5.2 Data Transmission in Master Mode

A data sending procedure and operations in master mode are shown below. Figures 14.5.2.1 and 14.5.2.2 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 SPIA_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
3. Write transmit data to the SPIA_*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 SPIA_*n*TXD register.
The transmit data in the SPIA_*n*TXD register is automatically transferred to the shift register and the SPIA_*n*INTF.TBEIF bit is set to 1. If the SPIA_*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 SPIA_*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 SPIA_*n*TXD register after making sure the SPIA_*n*INTF.TBEIF bit is set to 1.

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

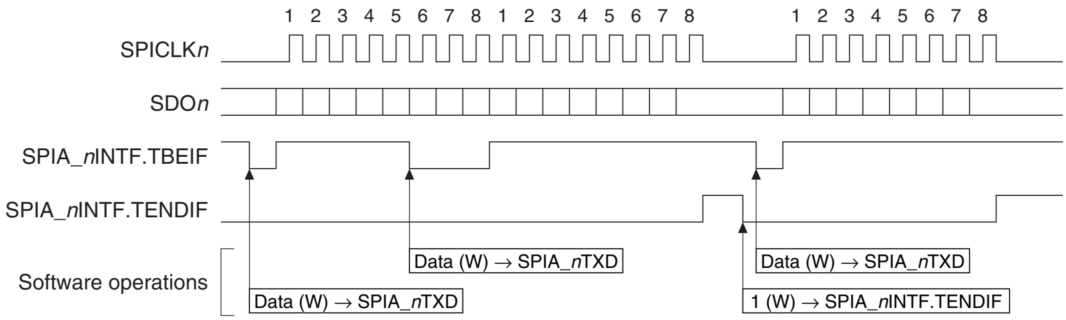


Figure 14.5.2.1 Example of Data Sending Operations in Master Mode (SPIA_*n*MOD.CHLN[3:0] bits = 0x7)

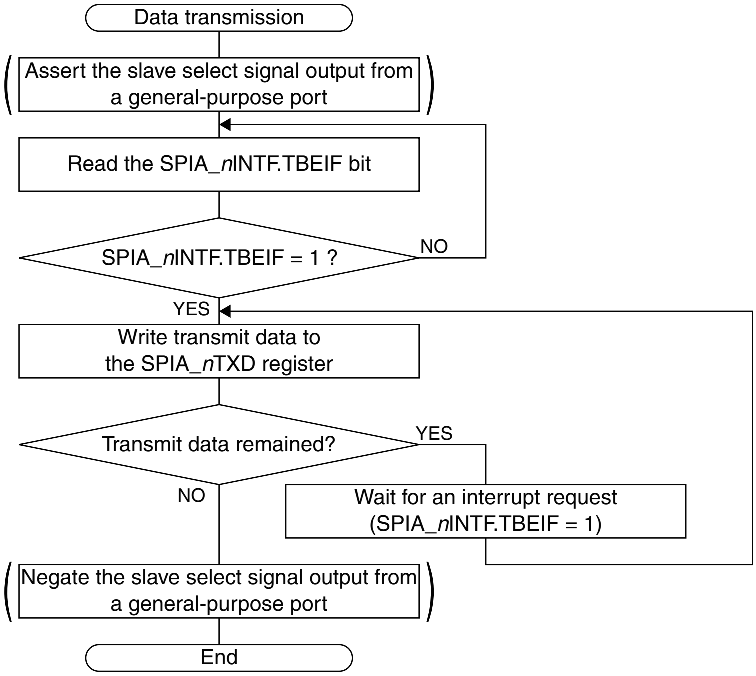


Figure 14.5.2.2 Data Transmission Flowchart in Master Mode

Data transmission using DMA

By setting the SPIA_*n*TBEDMAEN.TBEDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and transmit data is transferred from the specified memory to the SPIA_*n*TXD register via DMA Ch.*x* when the SPIA_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).

This automates the procedure from Step 2 to Step 5 described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance so that transmit data will be transferred to the SPIA_*n*TXD register. For more information on DMA, refer to the "DMA Controller" chapter.

Table 14.5.2.1 DMA Data Structure Configuration Example (for 16-bit Data Transmission)

Item		Setting example
End pointer	Transfer source	Memory address in which the last transmit data is stored
	Transfer destination	SPIA_*n*TXD register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x1 (haflword)
	src_inc	0x1 (+2)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

14.5.3 Data Reception in Master Mode

A data receiving procedure and operations in master mode are shown below. Figures 14.5.3.1 and 14.5.3.2 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 SPIA_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
3. Write dummy data (or transmit data) to the SPIA_*n*TXD register.
4. Wait for a transmit buffer empty interrupt (SPIA_*n*INTF.TBEIF bit = 1).
5. Write dummy data (or transmit data) to the SPIA_*n*TXD register.
6. Wait for a receive buffer full interrupt (SPIA_*n*INTF.RBFIF bit = 1).
7. Read the received data from the SPIA_*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 SPIA_*n*TXD register.
The SPICLK*n* pin outputs clocks of the number of the bits specified by the SPIA_*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 SPIA_*n*INTF.RBFIF bit is set to 1. At the same time SPIA issues a receive buffer full interrupt request if the SPIA_*n*INTE.RBFIE bit = 1. After that, the received data in the receive data buffer can be read through the SPIA_*n*RXD register.

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

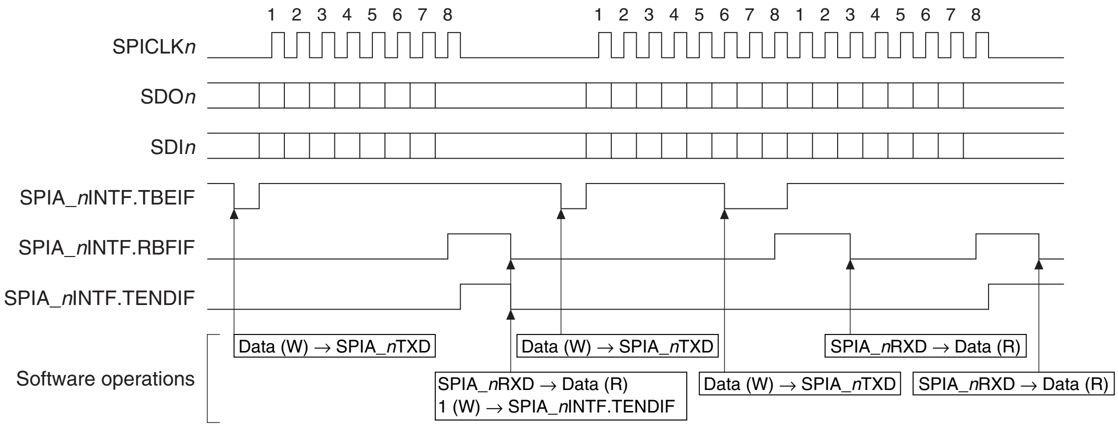


Figure 14.5.3.1 Example of Data Receiving Operations in Master Mode (SPIA_*n*MOD.CHLN[3:0] bits = 0x7)

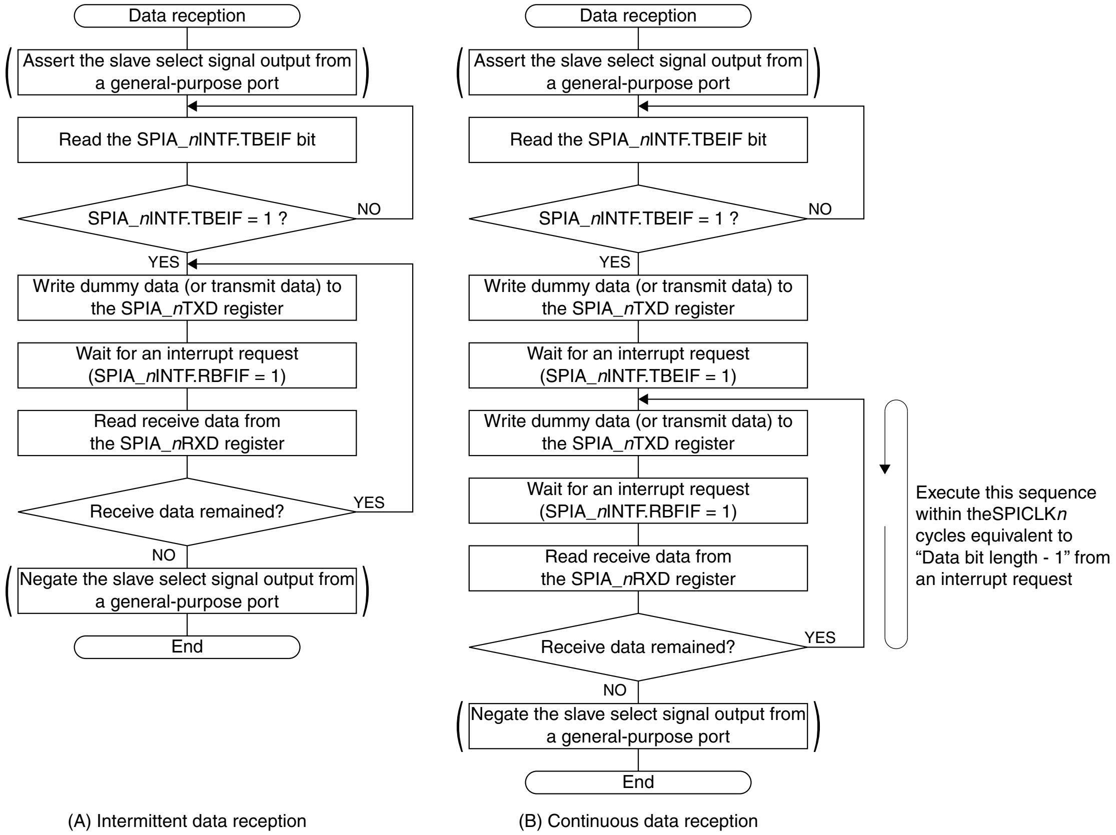


Figure 14.5.3.2 Data Reception Flowcharts in Master Mode

Data reception using DMA

For data reception, two DMA controller channels should be used to write dummy data to the SPIA_*n*TXD register as a reception start trigger and to read the received data from the SPIA_*n*RXD register.

By setting the SPIA_*n*TBEDMAEN.TBEDMAEN*x1* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and dummy data is transferred from the specified memory to the SPIA_*n*TXD register via DMA Ch.*x1* when the SPIA_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).

By setting the SPIA_*n*RBFDMAEN.RBFDMAEN*x2* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and the received data is transferred from the SPIA_*n*RXD register to the specified memory via DMA Ch.*x2* when the SPIA_*n*INTF.RBFIF bit is set to 1 (receive buffer full).

This automates the procedure from Step 2 to Step 8 described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 14.5.3.1 DMA Data Structure Configuration Example (for Writing 16-bit Dummy Transmit Data)

Item		Setting example
End pointer	Transfer source	Memory address in which dummy data is stored
	Transfer destination	SPIA_*n*TXD register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x1 (haflword)
	src_inc	0x3 (no increment)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

Table 14.5.3.2 DMA Data Structure Configuration Example (for 16-bit Data Reception)

Item		Setting example
End pointer	Transfer source	SPIA_*n*RXD register address
	Transfer destination	Memory address to which the last received data is stored
Control data	dst_inc	0x1 (+2)
	dst_size	0x1 (haflword)
	src_inc	0x3 (no increment)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

14.5.4 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 (SPIA_*n*INTF.TENDIF bit = 1).
2. Set the SPIA_*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*.

14.5.5 Data Transfer in Slave Mode

A data sending/receiving procedure and operations in slave mode are shown below. Figures 14.5.5.1 and 14.5.5.2 show a timing chart and flowcharts, respectively.

Data sending procedure

1. Check to see if the SPIA_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
2. Write transmit data to the SPIA_*n*TXD register.
3. Wait for a transmit buffer empty interrupt (SPIA_*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 SPIA_*n*TXD register after the SPIA_*n*INTF.TBEIF bit is set to 1 by the time the sending SPIA_*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 (SPIA_*n*INTF.RBFIF bit = 1).
2. Read the received data from the SPIA_*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 SPIA_*n*MOD.CHLN[3:0] bits is received when the SPIA_*n*INTF.RBFIF bit is set to 1, the SPIA_*n*RXD register is overwritten with the newly received data and the previously received data is lost. In this case, the SPIA_*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 SPIA_*n*INTF.TBEIF bit is set to 1.

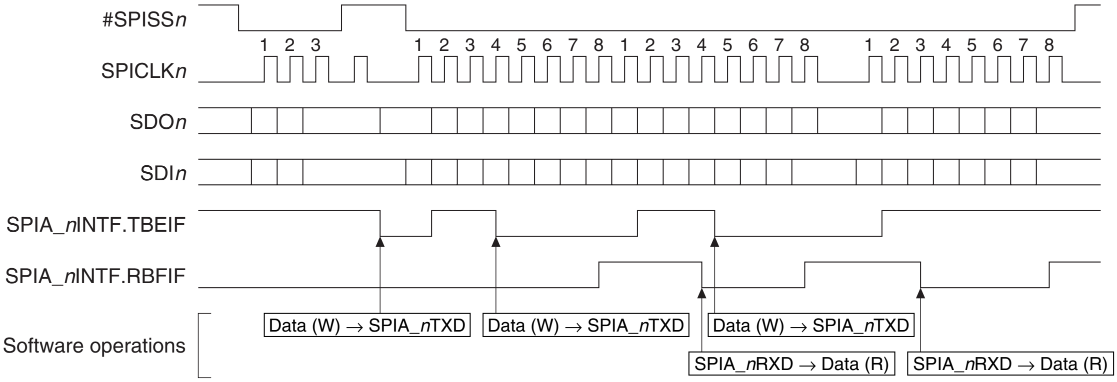


Figure 14.5.5.1 Example of Data Transfer Operations in Slave Mode (SPIA_*n*MOD.CHLN[3:0] bits = 0x7)

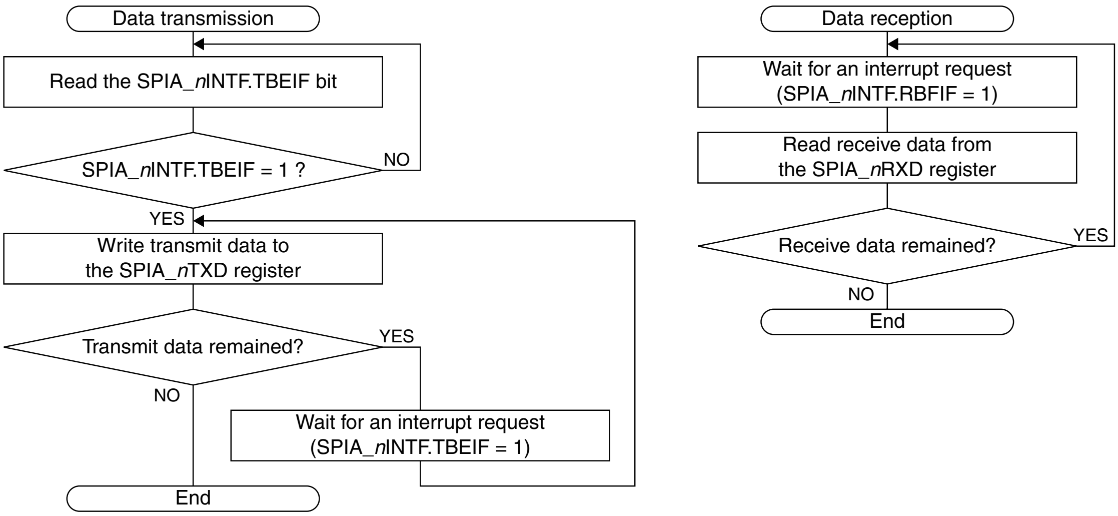


Figure 14.5.5.2 Data Transfer Flowcharts in Slave Mode

14.5.6 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 (SPIA_*n*INTF.TENDIF bit = 1). Or determine end of transfer via the received data.
2. Set the SPIA_*n*CTL.MODEN bit to 0 to disable the SPIA Ch.*n* operations.

14.6 Interrupts

SPIA has a function to generate the interrupts shown in Table 14.6.1.

Table 14.6.1 SPIA Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
End of transmission	SPIA_*n*INTF.TENDIF	When the SPIA_*n*INTF.TBEIF bit = 1 after data of the specified bit length (defined by the SPIA_*n*MOD.CHLN[3:0] bits) has been sent	Writing 1
Receive buffer full	SPIA_*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 SPIA_*n*RXD register
Transmit buffer empty	SPIA_*n*INTF.TBEIF	When transmit data written to the transmit data buffer is transferred to the shift register	Writing to the SPIA_*n*TXD register
Overrun error	SPIA_*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 CPU 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” chapter.
The SPIA_*n*INTF register also contains the BSY bit that indicates the SPIA operating status.
Figure 14.6.1 shows the SPIA_*n*INTF.BSY and SPIA_*n*INTF.TENDIF bit set timings.

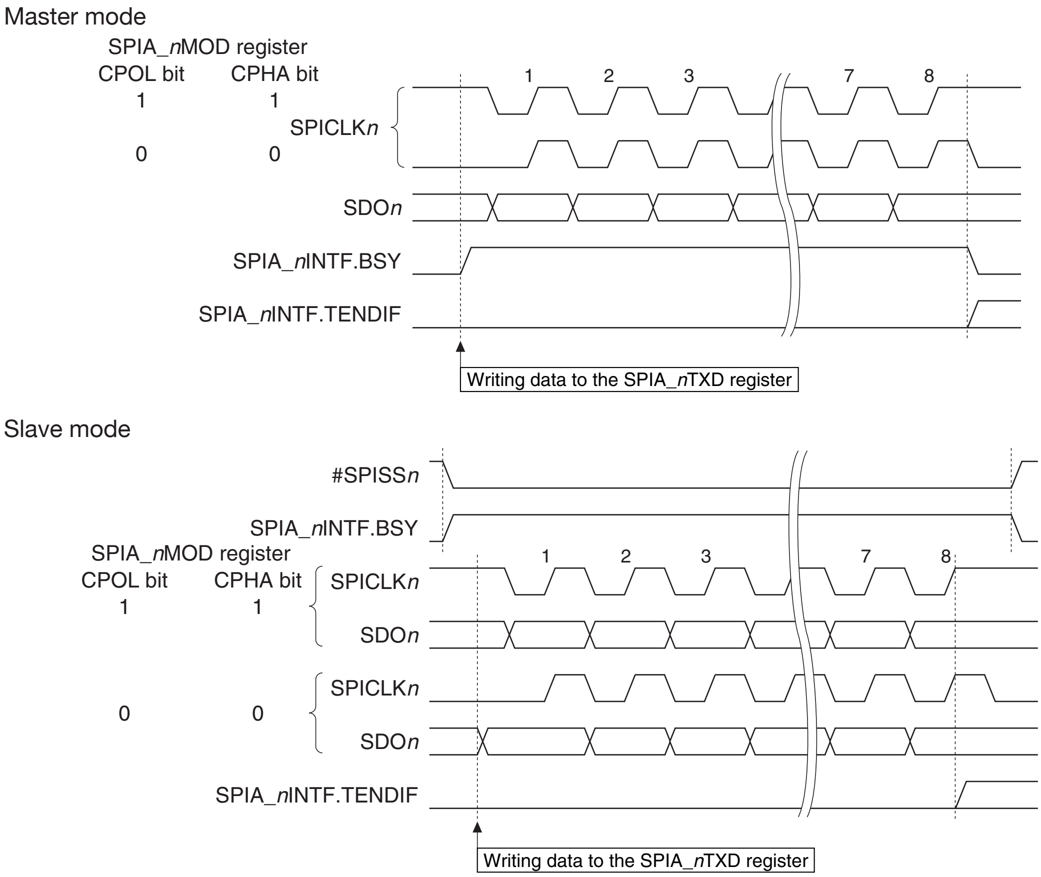


Figure 14.6.1 SPIA_*n*INTF.BSY and SPIA_*n*INTF.TENDIF Bit Set Timings (when SPIA_*n*MOD.CHLN[3:0] bits = 0x7)

14.7 DMA Transfer Requests

The SPIA has a function to generate DMA transfer requests from the causes shown in Table 14.7.1.

Table 14.7.1 DMA Transfer Request Causes of SPIA

Cause to request DMA transfer	DMA transfer request flag	Set condition	Clear condition
Receive buffer full	Receive buffer full flag (SPIA_*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 SPIA_*n*RXD register
Transmit buffer empty	Transmit buffer empty flag (SPIA_*n*INTF.TBEIF)	When transmit data written to the transmit data buffer is transferred to the shift register	Writing to the SPIA_*n*TXD register

The SPIA provides DMA transfer request enable bits corresponding to each DMA transfer request flag shown above for the number of DMA channels. A DMA transfer request is sent to the pertinent channel of the DMA controller only when the DMA transfer request flag, of which DMA transfer has been enabled by the DMA transfer request enable bit, is set. The DMA transfer request flag also serves as an interrupt flag, therefore, both the DMA transfer request and the interrupt cannot be enabled at the same time. After a DMA transfer has completed, disable the DMA transfer to prevent unintended DMA transfer requests from being issued. For more information on the DMA control, refer to the "DMA Controller" chapter.

14.8 Control Registers

SPIA Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPIA_*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 14.8.1 Data Bit Length Settings

SPIA_*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 SPIA_*n*MOD register settings can be altered only when the SPIA_*n*CTL.MODEN bit = 0.

SPIA Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPIA_*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 SPIA_*n*CTL.MODEN bit is altered from 1 to 0 while sending/receiving data, the data being sent/received cannot be guaranteed. When setting the SPIA_*n*CTL.MODEN bit to 1 again after that, be sure to write 1 to the SPIA_*n*CTL.SFTRST bit as well.

SPIA Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPIA_*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 SPIA_*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 SPIA_*n*MOD.CHLN[3:0] bits will not be output from the SDO*n* pin.

Note: Be sure to avoid writing to the SPIA_*n*TXD register when the SPIA_*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
SPIA_*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 SPIA_*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 SPIA_*n*MOD.CHLN[3:0] bits become 0.

SPIA Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPIA_*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 SPIA_*n*RXD register.
	0	TBEIF	1	H0/S0	R	Cleared by writing to the SPIA_*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:
SPIA_*n*INTF.OEIF bit: Overrun error interrupt
SPIA_*n*INTF.TENDIF bit: End-of-transmission interrupt
SPIA_*n*INTF.RBFIF bit: Receive buffer full interrupt
SPIA_*n*INTF.TBEIF bit: Transmit buffer empty interrupt

SPIA Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPIA_*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:
SPIA_*n*INTE.OEIE bit: Overrun error interrupt
SPIA_*n*INTE.TENDIE bit: End-of-transmission interrupt
SPIA_*n*INTE.RBFIE bit: Receive buffer full interrupt
SPIA_*n*INTE.TBEIE bit: Transmit buffer empty interrupt

SPIA Ch.*n* Transmit Buffer Empty DMA Request Enable Register

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

Bits 15–0 TBEDMAEN[15:0]

These bits enable the SPIA to issue a DMA transfer request to the corresponding DMA channel (Ch.0–Ch.15) when a transmit buffer empty state has occurred.
1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

SPIA Ch.*n* Receive Buffer Full DMA Request Enable Register

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

Bits 15–0 RBFDMAEN[15:0]

These bits enable the SPIA to issue a DMA transfer request to the corresponding DMA channel (Ch.0–Ch.15) when a receive buffer full state has occurred.
1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

15 Quad Synchronous Serial Interface (QSPI)

15.1 Overview

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

- Supports both master and slave modes.
- Supports single, dual, and quad transfer modes.
- Data length: 2 to 16 clocks programmable.
- Data line drive length: 1 to 16 clocks programmable (for output direction only).
- 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 QSPICLK*n* only.
- Slave mode is capable of being operated in SLEEP mode allowing wake-up by a QSPI interrupt.
- Input pins can be pulled up/down with an internal resistor.
- Low CPU overhead memory mapped access mode that can access the external Flash memory with XIP (eXecute-In-Place) mode in the same manner as the embedded system memory.
  - Memory mapped access size: 8, 16, and 32-bit access.
  - 1M-byte external Flash memory mapped access area that allows programmable re-mapping.
  - Configurable 3 or 4-byte address cycle length.
  - Single, dual, or quad transfer mode is configurable for each address, mode byte/dummy, and data cycle.
  - Programmable mode bytes for both XIP mode activation and termination.
  - Configurable mode byte/dummy output cycle length.
- Can issue a DMA transfer request when a receive buffer full, a transmit buffer empty, or a memory mapped access (32-bit read) occurs.

Figure 15.1.1 shows the QSPI configuration.

Table 15.1.1 QSPI Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	1 channels (Ch.0)		
Internal clock input	Ch.0 ← 16-bit timer Ch.2		
Memory mapped access area for external Flash memory	1M-byte area beginning with address 0x0004_0000		

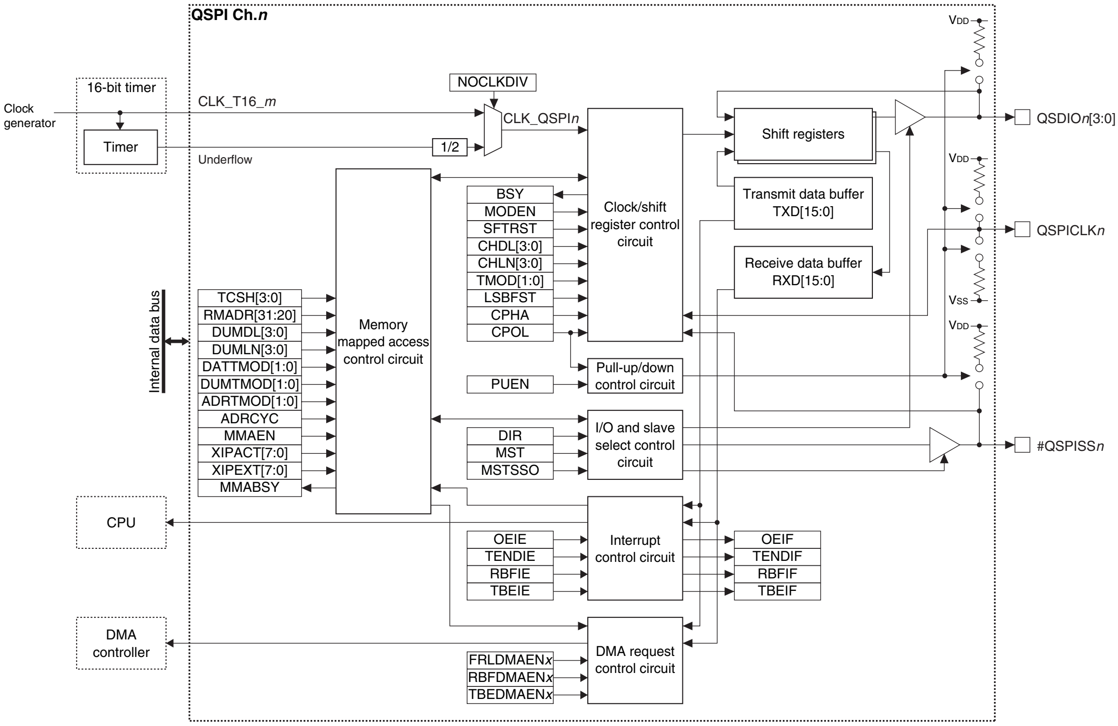


Figure 15.1.1 QSPI Configuration

15.2 Input/Output Pins and External Connections

15.2.1 List of Input/Output Pins

Table 15.2.1.1 lists the QSPI pins.

Table 15.2.1.1 List of QSPI Pins

Pin name	I/O*	Initial status*	Function
QSDIO*n*[3:0]	I or O	I (Hi-Z)	QSPI Ch.*n* data input/output pin
QSPICLK*n*	I or O	I (Hi-Z)	QSPI Ch.*n* external clock input/output pin
#QSPISS*n*	I or O	I (Hi-Z)	QSPI Ch.*n* slave select signal input/output pin

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

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

15.2.2 External Connections

The QSPI operates in master or slave mode. The memory mapped access mode is available only in master mode.
When QSPI Ch.*n* is operating in memory mapped access mode, the #QSPISS*n* output is controlled by the internal state machine. In this case, only one external QSPI device can be connected.
When QSPI Ch.*n* is operating in register access master mode, the #QSPISS*n* output is directly controlled by a register bit. In this case, GPIO pins other than #QSPISS*n* can also be used as the slave select output ports to connect the QSPI to more than one external QSPI device.
Figures 15.2.2.1 to 15.2.2.7 show connection diagrams between the QSPI in each mode and external QSPI devices.

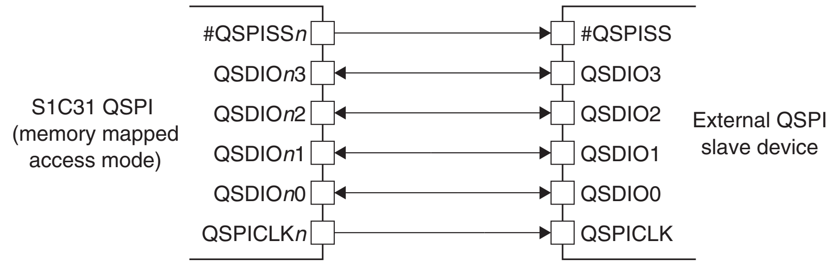


Figure 15.2.2.1 Connections between QSPI in Memory Mapped Access Mode and an External QSPI Slave Device

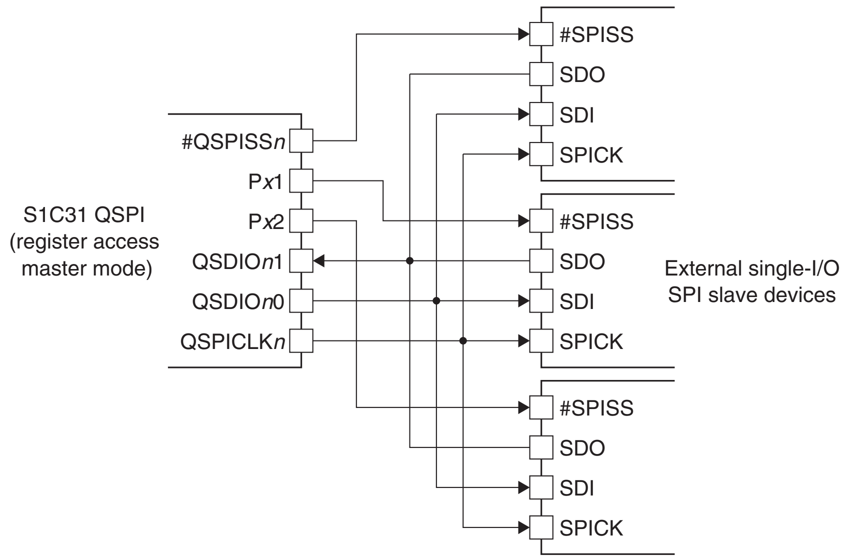


Figure 15.2.2.2 Connections between QSPI in Register Access Master Mode and External Single-I/O SPI (Legacy SPI) Slave Devices

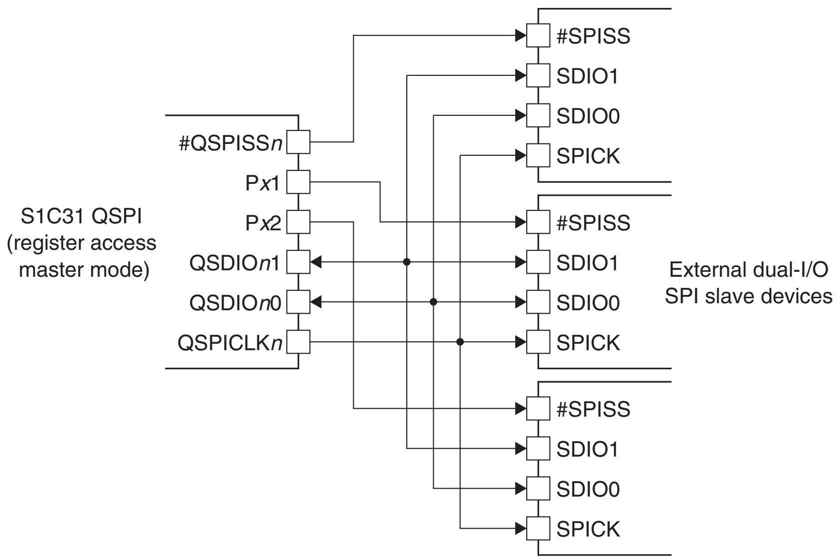


Figure 15.2.2.3 Connections between QSPI in Register Access Master Mode and External Dual-I/O SPI Slave Devices

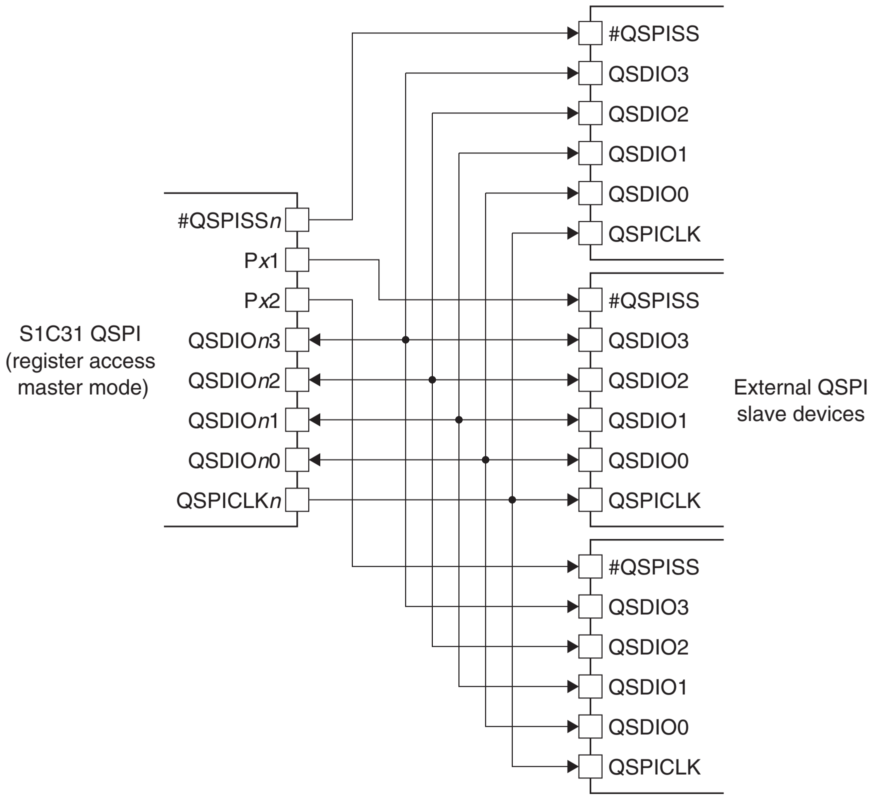


Figure 15.2.2.4 Connections between QSPI in Register Access Master Mode and External QSPI Slave Devices

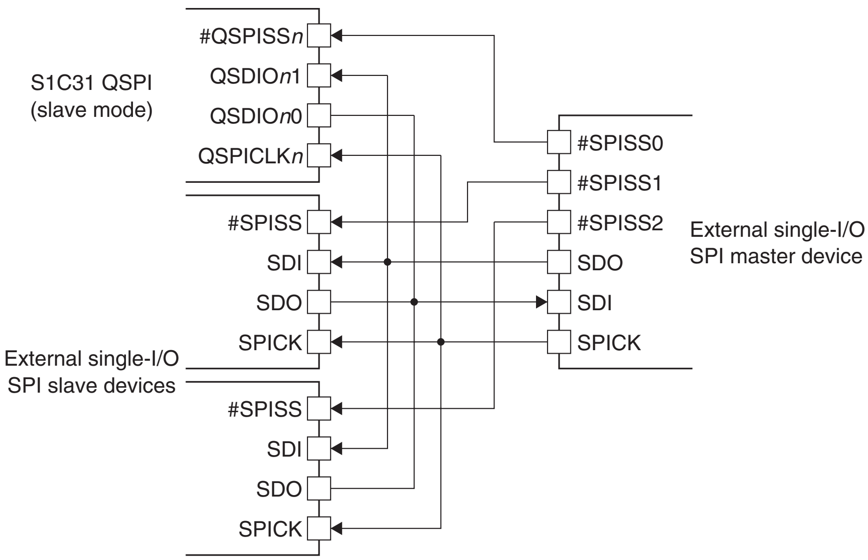


Figure 15.2.2.5 Connections between QSPI in Slave Mode and External Single-I/O SPI (Legacy SPI) Master Device

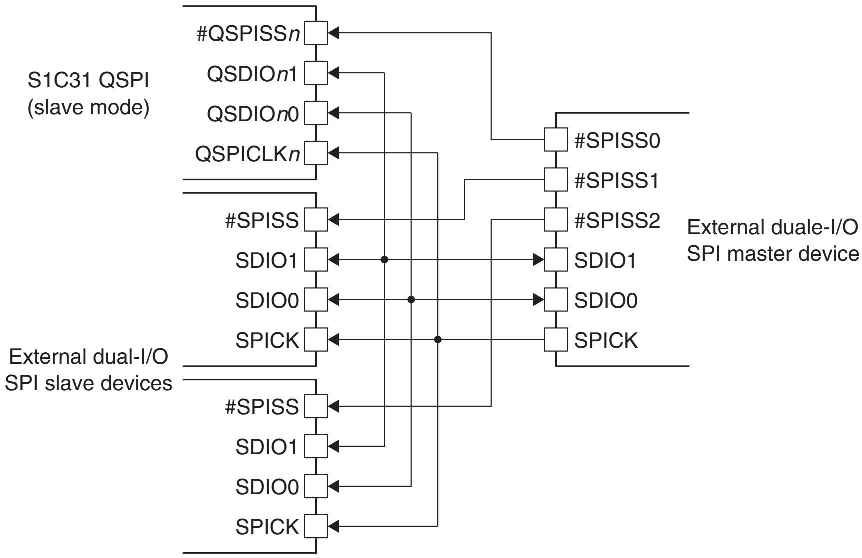


Figure 15.2.2.6 Connections between QSPI in Slave Mode and External Dual-I/O SPI Master Device

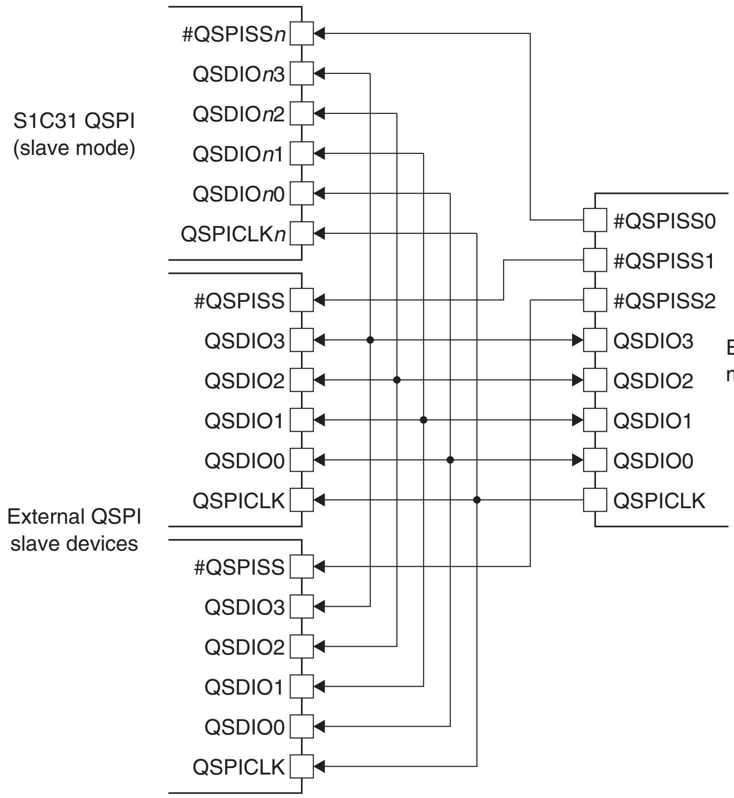


Figure 15.2.2.7 Connections between QSPI in Slave Mode and External QSPI Master Device

15.2.3 Pin Functions in Master Mode and Slave Mode

The pin functions are changed according to the transfer direction, transfer mode, and master/slave mode selections. The differences in pin functions between the modes are shown in Table 15.2.3.1.

Table 15.2.3.1 Pin Function Differences between Modes

Pin	Function in master mode			Function in slave mode		
	Single transfer mode	Dual transfer mode	Quad transfer mode	Single transfer mode	Dual transfer mode	Quad transfer mode
QSDIO*n*[3:2]	Always placed into Hi-Z state.		These pins are placed into input or output state according to the QSPI_*n*CTL.DIR bit setting.	Always placed into Hi-Z state.		These pins are placed into output state while a low level is applied to the #QSPISS*n* pin and the QSPI_*n*CTL.DIR bit is set to 0 (output), or placed into Hi-Z state while a high level is applied to the #QSPISS*n* pin or the QSPI_*n*CTL.DIR bit is set to 1 (input).
QSDIO*n*1	Always placed into input state.	These pins are placed into input or output state according to the QSPI_*n*CTL.DIR bit setting.		Always placed into input state.	These pins are placed into output state while a low level is applied to the #QSPISS*n* pin and the QSPI_*n*CTL.DIR bit is set to 0 (output), or placed into Hi-Z state while a high level is applied to the #QSPISS*n* pin or the QSPI_*n*CTL.DIR bit is set to 1 (input).	
QSDIO*n*0	Always placed into output state.			This pin is placed into output state while a low level is applied to the #QSPISS*n* pin or placed into Hi-Z state while a high level is applied to the #QSPISS*n* pin.		
QSPICLK*n*	Outputs the QSPI clock to external devices. Output clock polarity and phase can be configured if necessary.			Inputs an external QSPI clock. Clock polarity and phase can be designated according to the input clock.		
#QSPISS*n*	This pin is used to output the slave select signal in master mode. In memory mapped access mode, this pin is controlled by the internal state machine. In register access mode, this pin is controlled by a register bit. When connecting more than one external slave device, general-purpose I/O ports can be used to output the extra slave select signals.			Applying a low level to the #QSPISS*n* pin enables the QSPI to transmit/receive data. While a high level is applied to this pin, the QSPI is not selected as a slave device. Data input to the QSDIO*n* pins and the clock input to the QSPICLK*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.		

15.2.4 Input Pin Pull-Up/Pull-Down Function

The QSPI pins (QSDIO*n*[3:0] pins in master mode or QSDIO*n*[3:0] pins, QSPICLK*n*, and #QSPISS*n* pins in slave mode) have a pull-up or pull-down function as shown in Table 15.2.4.1. This function is enabled by setting the QSPI_*n*MOD.PUEN bit to 1.

Table 15.2.4.1 Pull-Up or Pull-Down of QSPI Pins

Pin	Master mode	Slave mode
QSDIO*n*[3:0]	Pull-up	Pull-up
QSPICLK*n*	–	QSPI_*n*MOD.CPOL bit = 1: Pull-up QSPI_*n*MOD.CPOL bit = 0: Pull-down
#QSPISS*n*	–	Pull-up

15.3 Clock Settings

15.3.1 QSPI Operating Clock

Operating clock in master mode

In master mode, the QSPI 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 QSPI_*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 QSPI channel is input to the QSPI as CLK_QSPI*n*. Since this clock is also used as the QSPI clock QSPICLK*n* without changing, the CLK_QSPI*n* frequency becomes the baud rate.

To supply CLK_QSPI*n* to the QSPI, 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 QSPI_*n*MOD.NOCLKDIV bit to 0, the QSPI inputs the underflow signal generated by the corresponding 16-bit timer channel and converts it to the QSPICLK*n*. The 16-bit timer must be run with an appropriate reload data set. The QSPICLK*n* frequency (baud rate) and the 16-bit timer reload data are calculated by the equations shown below.

$$f_{QSPICLK} = \frac{f_{CLK\_QSPI}}{2 \times (RLD + 1)} \qquad RLD = \frac{f_{CLK\_QSPI}}{f_{QSPICLK} \times 2} - 1 \qquad \text{(Eq. 15.1)}$$

Where

$f_{QSPICLK}$: QSPICLK*n* frequency [Hz] (= baud rate [bps])

f_{CLK_QSPI}: QSPI operating clock frequency [Hz]

RLD: 16-bit timer reload data value

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

Operating clock in slave mode

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

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

15.3.2 Clock Supply During Debugging

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

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

The QSPI in slave mode operates with the external SPI/QSPI master clock input from the QSPICLK*n* pin regardless of whether the CPU is placed into debug state or normal operation state.

15.3.3 QSPI Clock (QSPICLK*n*) Phase and Polarity

The QSPICLK*n* phase and polarity can be configured separately using the QSPI_*n*MOD.CPHA bit and the QSPI_*n*MOD.CPOL bit, respectively. Figure 15.3.3.1 shows the clock waveform and data input/output timing in each setting.

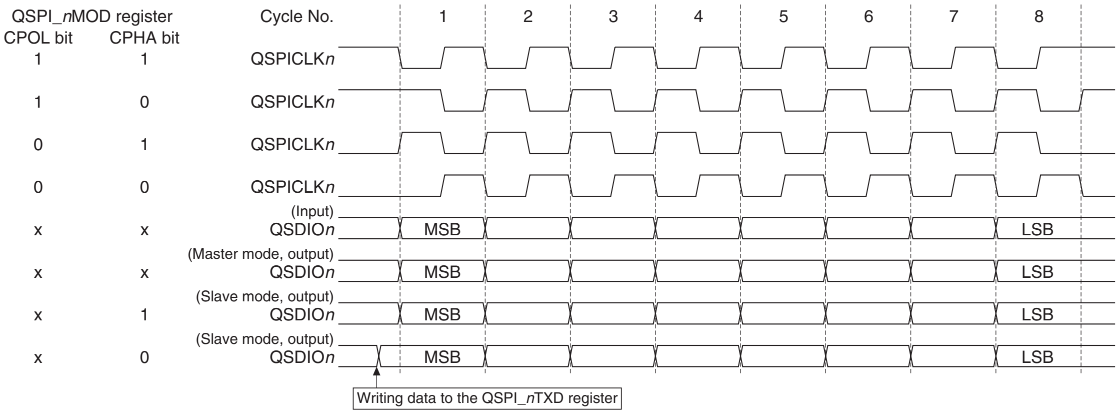


Figure 15.3.3.1 QSPI Clock Phase and Polarity (QSPI_*n*MOD.LSBFST bit = 0, QSPI_*n*MOD.CHLN[3:0] bits = 0x7)

15.4 Data Format

The QSPI data length can be selected from 2 to 16 clocks by setting the QSPI_*n*MOD.CHLN[3:0] bits. The input/output permutation is configurable to MSB first or LSB first using the QSPI_*n*MOD.LSBFST bit. Figures 15.4.1 to 15.4.3 show data format examples in different transfer modes (QSPI_*n*MOD.TMOD[1:0]) when the QSPI_*n*MOD.CPOL bit = 0 and the QSPI_*n*MOD.CPHA bit = 0.

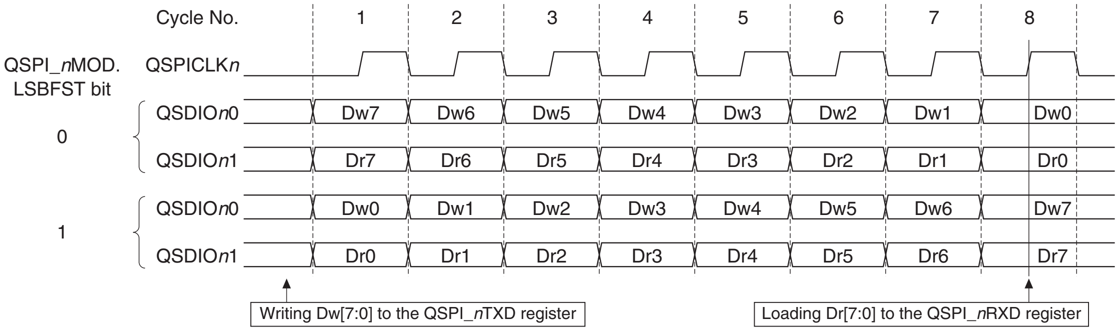


Figure 15.4.1 Data Format Selection for Single Transfer Mode Using the QSPI_*n*MOD.LSBFST Bit
(QSPI_*n*MOD.TMOD[1:0] bits = 0x0, QSPI_*n*MOD.CHDL[3:0] bits = 0x7, QSPI_*n*MOD.CHLN[3:0] bits = 0x7, QSPI_*n*MOD.CPOL bit = 0, QSPI_*n*MOD.CPHA bit = 0)

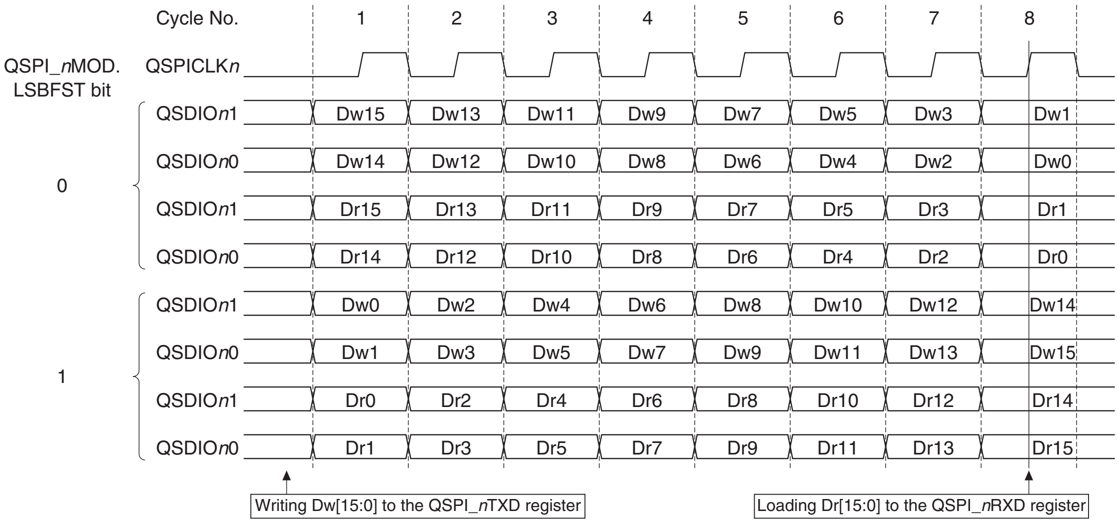


Figure 15.4.2 Data Format Selection for Dual Transfer Mode Using the QSPI_*n*MOD.LSBFST Bit
(QSPI_*n*MOD.TMOD[1:0] bits = 0x1, QSPI_*n*MOD.CHDL[3:0] bits = 0x7, QSPI_*n*MOD.CHLN[3:0] bits = 0x7, QSPI_*n*MOD.CPOL bit = 0, QSPI_*n*MOD.CPHA bit = 0)

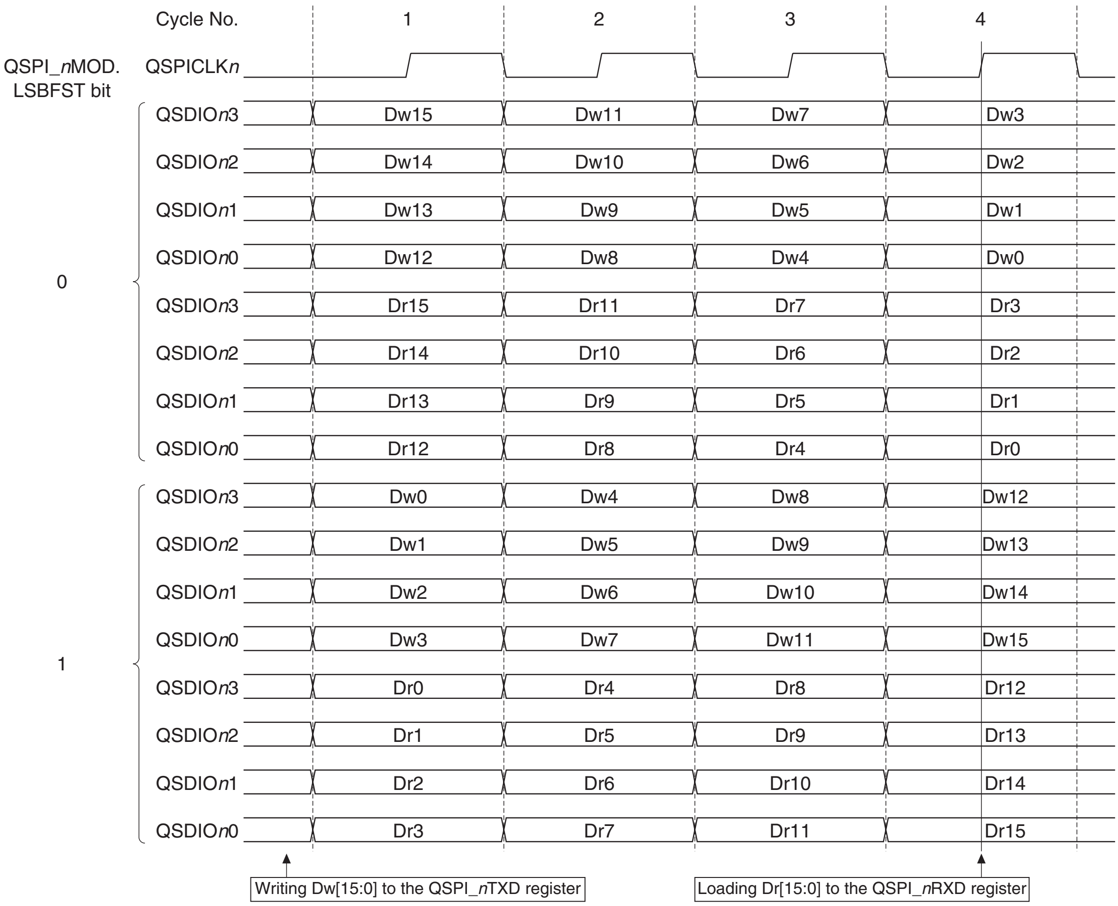


Figure 15.4.3 Data Format Selection for Quad Transfer Mode Using the QSPI_*n*MOD.LSBFST Bit
(QSPI_*n*MOD.TMOD[1:0] bits = 0x2, QSPI_*n*MOD.CHDL[3:0] bits = 0x3, QSPI_*n*MOD.CHLN[3:0] bits = 0x3, QSPI_*n*MOD.CPOL bit = 0, QSPI_*n*MOD.CPHA bit = 0)

15.5 Operations

15.5.1 Register Access Mode

Data can be read from or written to the external SPI/QSPI device by accessing the registers in both master and slave modes.

In single transfer mode, transmit data are always output from the QSDIO*n*0 pin and receive data are always input to the QSDIO*n*1 pin (the QSDIO*n*[3:2] pins are not used). The operations are backward compatible with legacy SPI (e.g., synchronous serial interface of this MCU).

In dual transfer mode, transmit data are output from the QSDIO*n*[1:0] pins when the transfer direction is set to output (QSPI_*n*CTL.DIR bit = 0). Receive data are input from the QSDIO*n*[1:0] pins when the transfer direction is set to input (QSPI_*n*CTL.DIR bit = 1). The QSDIO*n*[3:2] pins are not used. The number of data transfer clocks is configured using the QSPI_*n*MOD.CHLN[3:0] bits. Since two data lines are used for data transfer, the data bit length (number of clocks) is obtained by dividing the number of transfer data bits by two.

In quad transfer mode, transmit data are output from the QSDIO*n*[3:0] pins when the transfer direction is set to output (QSPI_*n*CTL.DIR bit = 0). Receive data are input from the QSDIO*n*[3:0] pins when the transfer direction is set to input (QSPI_*n*CTL.DIR bit = 1). The number of data transfer clocks is configured with the QSPI_*n*MOD.CHLN[3:0] bits. Since four data lines are used for data transfer, the data bit length (number of clocks) is obtained by dividing the number of transfer data bits by four.

$$\text{LENGTH} = \frac{\text{BIT}}{\text{N}} \text{ [clocks]} \qquad \text{(Eq. 15.2)}$$

Where

LENGTH: Data bit length [clocks]
BIT: Number of transfer data bits
N: 1 (single transfer mode), 2 (dual transfer mode), or 4 (quad transfer mode)

15.5.2 Memory Mapped Access Mode

Memory mapped access mode is a low CPU overhead operation mode used with master mode to read data from an external Flash memory, which supports XIP (eXecute-In-Place) mode. Once the external Flash memory enters XIP mode and a read command is executed, the same read command operation can be performed by controlling the slave select signal (inactive to active) and sending a new address to be accessed without the command being resent. This may reduce command re-execution overhead and random access time.

An XIP session consists of a command cycle, an address cycle, a dummy cycle, and consecutive data cycles, and it begins with an XIP specific read command similar to a general read command. Unlike a general read command, one or more data lines must be driven to send XIP activation or termination confirmation bit(s) at the beginning of the dummy cycle of an XIP session

In an XIP session, to start reading from a non-sequential Flash memory address, which is not continuous to the previous read address, assert the slave signal again after negating it once. After that, just send an address cycle to specify the new read start address and a dummy cycle including an XIP activation (continuation) confirmation bit(s), as the command cycle is not needed in this XIP session. The Flash memory performs read operations the same as the read command previously executed to execute a data cycle that includes a given number of data stored from the newly specified address.

To terminate an XIP session, first assert the slave signal again after negating it once. Then, send an address cycle with the address bits set to all high (suggested by most Flash memory manufacturers) and a dummy cycle including an XIP termination confirmation bit(s) at the beginning of the cycle on one or more data lines. After that, negate the slave select signal.

Figures 15.5.2.1 and 15.5.2.2 show Spansion S25FL128S Quad I/O Read command sequences as XIP operation examples.

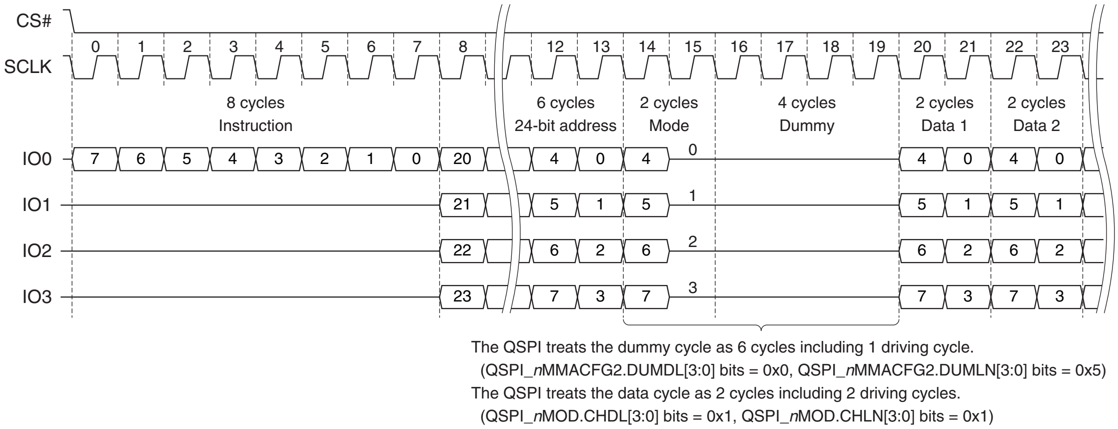
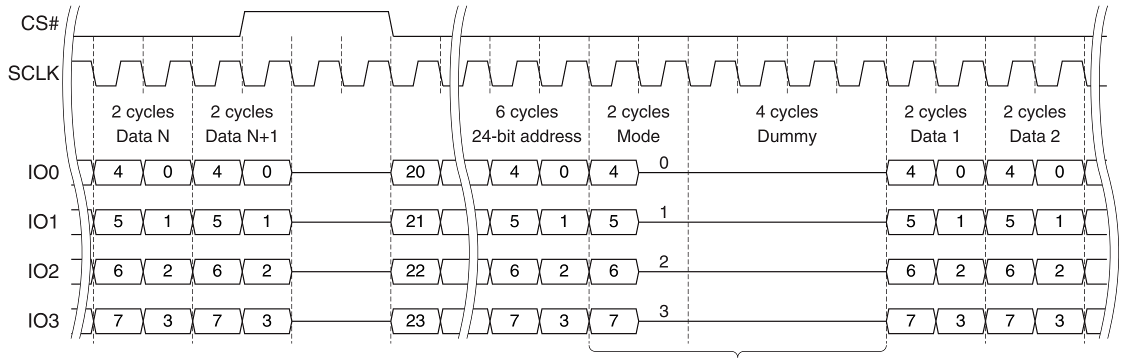


Figure 15.5.2.1 XIP Example - Spansion S25FL128S Quad I/O Read Command Sequence
(3-byte address, 0xeb [ExtAdd = 0], LC = 0b00)



The QSPI treats the dummy cycle as 6 cycles including 1 driving cycle.
(QSPI_*n*MMACFG2.DUMDL[3:0] bits = 0x0, QSPI_*n*MMACFG2.DUMLN[3:0] bits = 0x5)
The QSPI treats the data cycle as 2 cycles including 2 driving cycles.
(QSPI_*n*MOD.CHDL[3:0] bits = 0x1, QSPI_*n*MOD.CHLN[3:0] bits = 0x1)

Figure 15.5.2.2 XIP Example - Spansion S25FL128S Continuous Quad I/O Read Command Sequence (3-byte address, LC = 0b00)

In memory mapped access mode, the QSPI automates toggling of the slave select signal and executing address, dummy, and data cycles so that the CPU will be able to read the external Flash memory mapped to the system memory area. This further reduces CPU overhead.

The transfer mode can be configured for address, dummy, and data cycles individually. The address cycle supports 24 and 32-bit addresses. The QSPI considers that the mode cycle (or XIP activation/termination confirmation) is a part of the dummy cycle, so a mode cycle is sent out on the I/O data line in a dummy cycle.

The memory mapped access area for external Flash memory in the system memory area is used to map the external Flash memory and to access from the CPU. Up to 4G-byte Flash memory can be accessed from this area using a remapping register. Once the external Flash memory is set into XIP mode and a read command is sent in register access mode, the CPU can directly read external Flash memory data through this area. When a read access to a non-sequential address occurs in memory mapped access mode, the QSPI automatically executes a new address and dummy cycles. When memory mapped access mode is disabled by setting a register, the QSPI executes an address cycle and a dummy cycle including a mode byte that specifies to terminate XIP mode.

Memory mapped access mode supports 8, 16, and 32-bit read accesses.

The 32-bit access is mainly used to read data in a large memory block sequentially. In this access, up to two 32-bit data are prefetched into the internal FIFO. Therefore, zero-wait read access is possible if the desired data has already been fetched in the FIFO.

The 8 and 16-bit accesses are mainly used to read data in a small memory block or to read data from non-sequential addresses. Prefetching is not performed as it is unnecessary in non-sequential read. Therefore, overhead of a couple of clocks occurs between accesses.

The QSPI allows incorporating 8 and 16-bit accesses into 32- bit accesses. Prefetching data into FIFO is only performed immediately after a 32-bit read. An 8 or 16-bit read at the sequential address after a 32-bit read allows zero-wait read if the desired data has already been fetched in the FIFO.

15.5.3 Initialization

QSPI 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 QSPI Ch.*n*.
2. Configure the following QSPI_*n*MOD register bits:
   - QSPI_*n*MOD.PUEN bit (Enable input pin pull-up/down)
   - QSPI_*n*MOD.NOCLKDIV bit (Select master mode operating clock)
   - QSPI_*n*MOD.LSBFST bit (Select MSB first/LSB first)
   - QSPI_*n*MOD.CPHA bit (Select clock phase)
   - QSPI_*n*MOD.CPOL bit (Select clock polarity)
   - QSPI_*n*MOD.MST bit (Select master/slave mode)

3. Configure the following register bits when using memory mapped access mode:
   - QSPI_*n*MMACFG1.TCSH[3:0] bits (Set slave select signal negation period)
   - QSPI_*n*RMADRH.RMADR[31:20] bits (Set remapping address)
   - QSPI_*n*MMACFG2.DUMDL[3:0] bits (Select dummy cycle drive length)
   - QSPI_*n*MMACFG2.DUMLN[3:0] bits (Select dummy cycle length)
   - QSPI_*n*MMACFG2.DATTMOD[1:0] bits (Select data cycle transfer mode)
   - QSPI_*n*MMACFG2.DUMTMOD[1:0] bits (Select dummy cycle transfer mode)
   - QSPI_*n*MMACFG2.ADRTMOD[1:0] bits (Select address cycle transfer mode)
   - QSPI_*n*MMACFG2.ADRCYC bit (Select 24 or 32-bit address cycle)
   - QSPI_*n*MB.XIPACT[7:0] bits (Set XIP activation mode byte)
   - QSPI_*n*MB.XIPEXT[7:0] bits (Set XIP termination mode byte)
4. Assign the QSPI Ch.*n* input/output function to the ports. (Refer to the “I/O Ports” chapter.)
5. Set the following QSPI_*n*CTL register bits:
   - Set the QSPI_*n*CTL.SFTRST bit to 1. (Execute software reset)
   - Set the QSPI_*n*CTL.MODEN bit to 1. (Enable QSPI Ch.*n* operations)
6. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the QSPI_*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the QSPI_*n*INTE register to 1. * (Enable interrupts)

   * The initial value of the QSPI_*n*INTF.TBEIF bit is 1, therefore, an interrupt will occur immediately after the QSPI_*n*INTE.TBEIE bit is set to 1.
7. Configure the DMA controller and set the following QSPI control bits when using DMA transfer:
   - Write 1 to the DMA transfer request enable bits in the QSPI_*n*TBEDMAEN, QSPI_*n*RBFDMAEN, and QSPI_*n*FRLDMAEN registers. (Enable DMA transfer requests)

15.5.4 Data Transmission in Master Mode

A data sending procedure and operations in master mode are shown below. Figures 15.5.4.1 and 15.5.4.2 show a timing chart and a flowchart, respectively.

Data sending procedure

1. Set the QSPI_*n*CTL.DIR bit to 0 when QSPI Ch.*n* is set to dual or quad transfer mode. (This setting is not necessary in single transfer mode.)
2. Assert the slave select signal for the external slave device to be accessed by controlling the QSPI_*n*CTL.MSTSSO bit or the general-purpose output port used for an extra slave select signal output (if necessary).
3. Check to see if the QSPI_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
4. Write transmit data to the QSPI_*n*TXD register.
5. Wait for a QSPI interrupt when using interrupt.
6. Repeat Steps 3 to 5 (or 3 and 4) until the end of transmit data.
7. Negate the slave select signal that has been asserted in Step 2 by controlling the QSPI_*n*CTL.MSTSSO bit or the general-purpose output port (if necessary).

Data sending operations

QSPI Ch.*n* starts data sending operations when transmit data is written into the QSPI_*n*TXD register.

The transmit data in the QSPI_*n*TXD register is automatically transferred to the shift register and the QSPI_*n*INTF.TBEIF bit is set to 1. If the QSPI_*n*INTE.TBEIE bit = 1 (transmit buffer empty interrupt enabled), a transmit buffer empty interrupt occurs at the same time.

The QSPICLK*n* pin outputs clocks for the number of cycles specified by the QSPI_*n*MOD.CHLN[3:0] bits and the transmit data bits are output in sequence from the QSDIO*n* pins, according to the transfer mode specified by the QSPI_*n*MOD.TMOD[1:0] bits, in sync with these clocks.

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

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

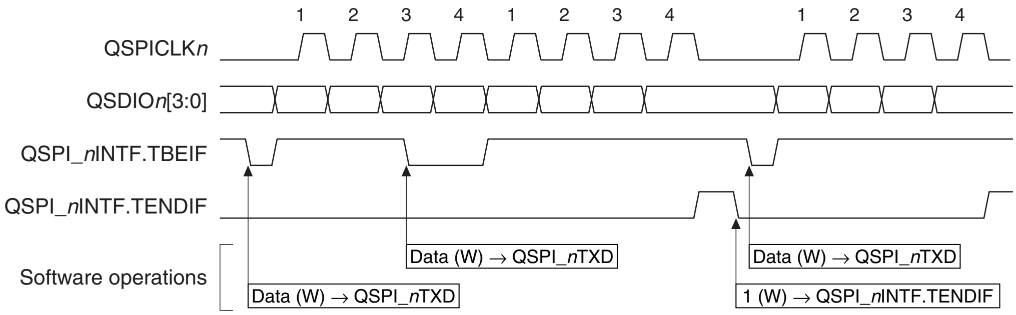


Figure 15.5.4.1 Example of Data Sending Operations in Master Mode
(QSPI_*n*MOD.CHDL[3:0] bits = QSPI_*n*MOD.CHLN[3:0] bits = 0x3)

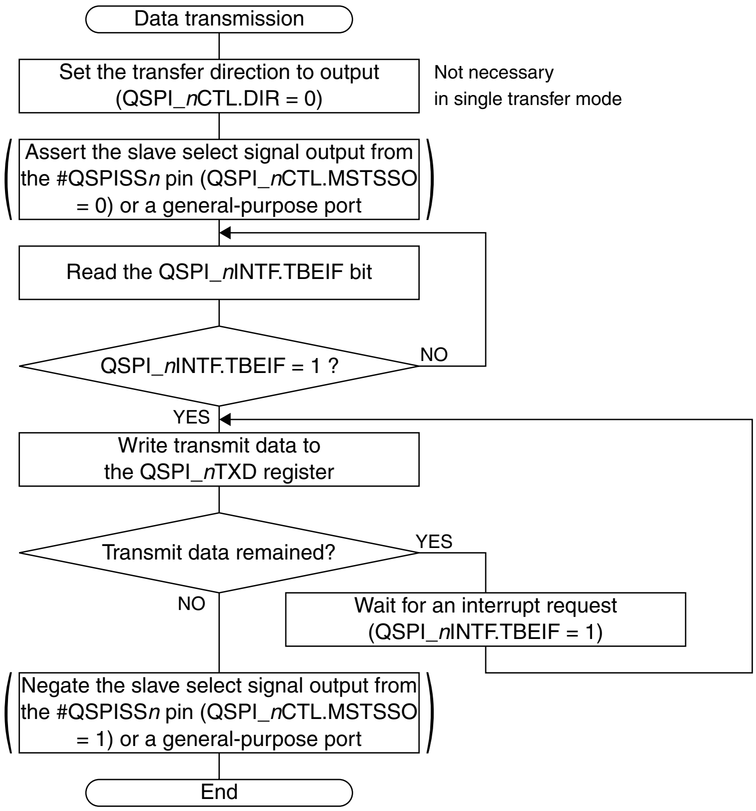


Figure 15.5.4.2 Data Transmission Flowchart in Master Mode

Data transmission using DMA

By setting the QSPI_*n*TBEDMAEN.TBEDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and transmit data is transferred from the specified memory to the QSPI_*n*TXD register via DMA Ch.*x* when the QSPI_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).

This automates the procedure from Step 3 to Step 6 described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance so that transmit data will be transferred to the QSPI_*n*TXD register. For more information on DMA, refer to the "DMA Controller" chapter.

Table 15.5.4.1 DMA Data Structure Configuration Example (for 16-bit Data Transmission)

Item		Setting example
End pointer	Transfer source	Memory address in which the last transmit data is stored
	Transfer destination	QSPI_*n*TXD register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x1 (haflword)
	src_inc	0x1 (+2)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

15.5.5 Data Reception in Register Access Master Mode

A data receiving procedure and operations in register access master mode are shown below. Figures 15.5.5.1 and 15.5.5.2 show a timing chart and flowcharts, respectively.

Data receiving procedure

1. Set the QSPI_*n*CTL.DIR bit to 1 when QSPI Ch.*n* is set to dual or quad transfer mode. (This setting is not necessary in single transfer mode.)
2. Assert the slave select signal for the external slave device to be accessed by controlling the QSPI_*n*CTL. MSTSSO bit or the general-purpose output port used for an extra slave select signal output (if necessary).
3. Check to see if the QSPI_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
4. Write dummy data (or transmit data) to the QSPI_*n*TXD register.
5. Wait for a transmit buffer empty interrupt (QSPI_*n*INTF.TBEIF bit = 1).
6. Write dummy data (or transmit data) to the QSPI_*n*TXD register.
7. Wait for a receive buffer full interrupt (QSPI_*n*INTF.RBFIF bit = 1).
8. Read the received data from the QSPI_*n*RXD register.
9. Repeat Steps 6 to 8 until the end of data reception.
10. Negate the slave select signal that has been asserted in Step 2 by controlling the QSPI_*n*CTL.MSTSSO bit or the general-purpose output port (if necessary).

Note: To perform continuous data reception without stopping QSPICLK*n*, Steps 8 and 6 operations must be completed within the QSPICLK*n* cycles equivalent to “Data bit length - 1” after Step 7.

Data receiving operations

In single transfer mode (QSPI_*n*MOD.TMOD[1:0] bits = 0), QSPI Ch.*n* operates similar to legacy SPI devices. The data receiving operation starts simultaneously with a data sending operation when transmit data (may be dummy data if data transmission is not required) is written to the QSPI_*n*TXD register. Transmit data are output from the QSDIO*n*0 pin and receive data are input from the QSDIO*n*1 pin.

In dual or quad transfer mode (QSPI_*n*MOD.TMOD[1:0] bits = 1 or 2), transmit data are not sent at data reception. Writing dummy data to the QSPI_*n*TXD register triggers the QSPI Ch.*n* to start supplying the data transfer clock from the QSPICLK*n* pin to the slave device.

The QSPICLK*n* pin outputs the number of clocks specified by the QSPI_*n*MOD.CHLN[3:0] bits. The receive data bits input from the QSDIO*n* pins, according to the transfer mode specified by the QSPI_*n*MOD. TMOD[1:0] bits, are shifted into the shift register in sync with these clocks.

When the last clock is output from the QSPICLK*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 QSPI_*n*INTF.RBFIF bit is set to 1. At the same time QSPI Ch.*n* issues a receive buffer full interrupt request if the QSPI_*n*INTE.RBFIE bit = 1. After that, the received data in the receive data buffer can be read through the QSPI_*n*RXD register.

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

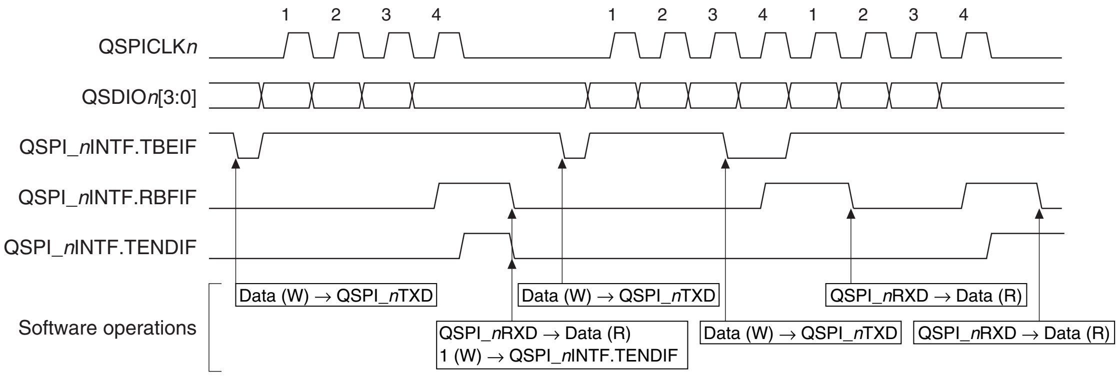


Figure 15.5.5.1 Example of Data Receiving Operations in Register Access Master Mode
(QSPI_nMOD.CHDL[3:0] bits = QSPI_nMOD.CHLN[3:0] bits = 0x3)

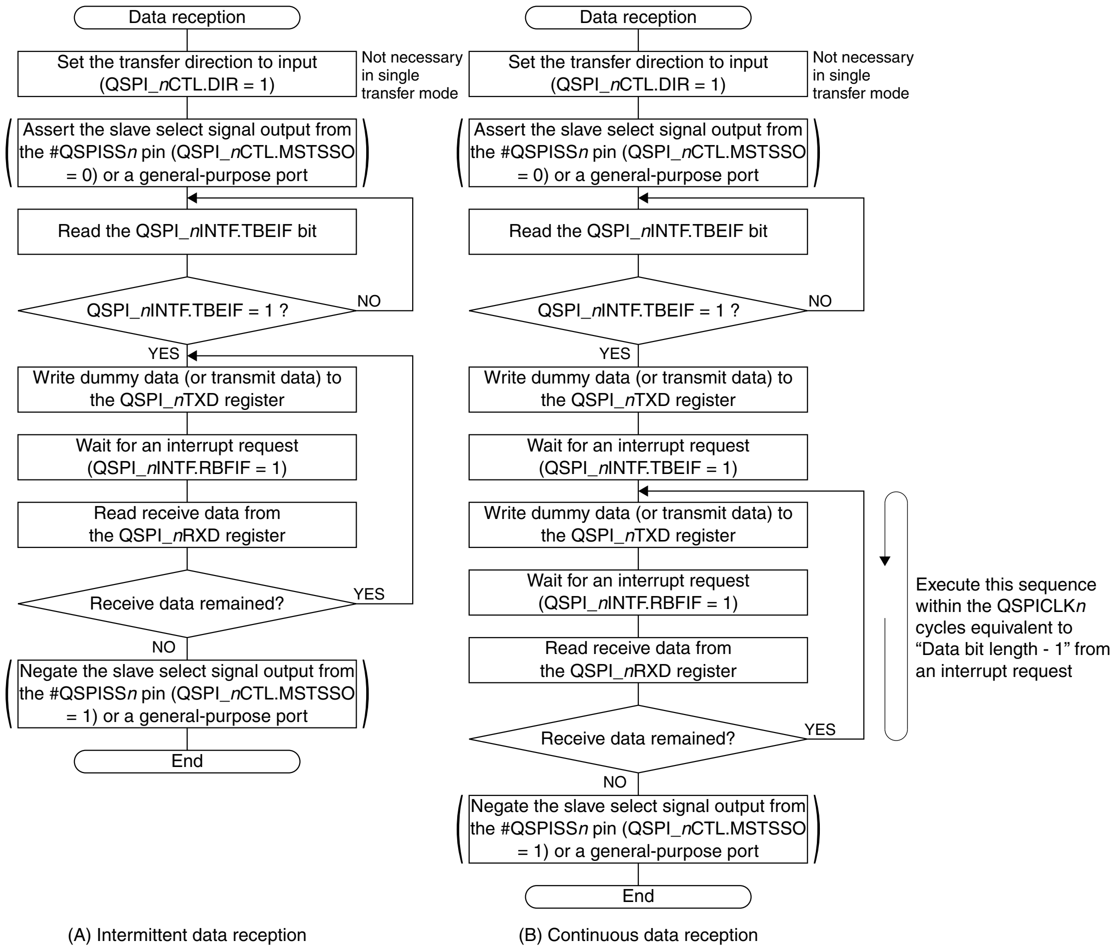


Figure 15.5.5.2 Data Reception Flowcharts in Register Access Master Mode

Data reception using DMA

For data reception, two DMA controller channels should be used to write dummy data to the QSPI_nTXD register as a reception start trigger and to read the received data from the QSPI_nRXD register.

By setting the QSPI_nTBEDMAEN.TBEDMAENx1 bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and dummy data is transferred from the specified memory to the QSPI_nTXD register via DMA Ch.x1 when the QSPI_nINTF.TBEIF bit is set to 1 (transmit buffer empty).

By setting the QSPI_*n*RBFDMAEN.RBFDMAEN*x2* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and the received data is transferred from the QSPI_*n*RXD register to the specified memory via DMA Ch.*x2* when the QSPI_*n*INTF.RBFIF bit is set to 1 (receive buffer full).
This automates the procedure from Step 3 to Step 9 described above.
The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 15.5.5.1 DMA Data Structure Configuration Example (for Writing 16-bit Dummy Transmit Data)

Item		Setting example
End pointer	Transfer source	Memory address in which dummy data is stored
	Transfer destination	QSPI_*n*TXD register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x1 (haflword)
	src_inc	0x3 (no increment)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

Table 15.5.5.2 DMA Data Structure Configuration Example (for 16-bit Data Reception)

Item		Setting example
End pointer	Transfer source	QSPI_*n*RXD register address
	Transfer destination	Memory address to which the last received data is stored
Control data	dst_inc	0x1 (+2)
	dst_size	0x1 (haflword)
	src_inc	0x3 (no increment)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

The following shows an example of the control procedure including the DMA controller operations:

1. Configure the primary data structure for the DMA channel (Ch.*x*) used for writing dummy bytes to the QSPI_*n*TXD register as shown in Table 15.5.5.1.
2. Configure the primary data structure for the DMA channel (Ch.*y*) used for reading data from the QSPI_*n*RXD register as shown in Table 15.5.5.2.
3. Enable both the DMA channels using the DMA controller register.
4. Increase the priority of the DMA channel used for reading data using the DMA controller register.
5. Clear the channel request masks for both the DMA channels using the DMA controller register.
6. Clear the DMA transfer completion interrupt flags using the DMA controller register.
7. Enable only the DMA transfer completion interrupt of the DMA channel used for reading using the DMA controller register.
8. Clear pending DMA interrupts in the CPU.
9. Enable pending DMA interrupts in the CPU.
10. Enable the QSPI to issue DMA transfer requests to both the DMA channels using the QSPI_*n*TBEDMAEN.TBEDMAEN*x* and QSPI_*n*RBFDMAEN.RBFDMAEN*y* bits.
11. Assert the slave select signal by controlling the QSPI_*n*CTL.MSTSSO bit, or the general-purpose output port used for an extra slave select signal output (if necessary).
12. Issue a software DMA transfer request to the DMA channel used for writing dummy bytes by setting the DMA controller register. This operation is required to read the first data and to set the receive buffer full status flag. Once the receive buffer full status flag is set, a hardware DMA request is generated, and the DMA controller transfers data from the QSPI_*n*RXD register and then writes another dummy byte to the QSPI_*n*TXD register, allowing the QSPI to read the next data.
13. Wait for a DMA interrupt.

14. Disable the DMA requests to be sent to both the DMA channels using the QSPI_*n*TBEDMAEN.TBEDMAEN*x* and QSPI_*n*RBFDMAEN.RBFDMAEN*y* bits.
15. Set the channel request masks for both the DMA channels using the DMA controller register.
16. Disable both the DMA channels using the DMA controller register.
17. Negate the slave select signal by controlling the QSPI_*n*CTL.MSTSSO bit or the general-purpose output port (if necessary).

15.5.6 Data Reception in Memory Mapped Access Mode

A data receiving procedure, and 32-bit and 8/16-bit received data read operations in memory mapped access mode are shown below. Figures 15.5.6.1 to 15.5.6.7 show their timing charts and a flowchart.

Data receiving procedure

QSPI Flash memories of different manufacturers have a different XIP operation mode setup procedure. The procedure described below assumes that the external Flash memory has already been placed into XIP operation mode.

1. Send a read command that supports XIP mode to the external Flash memory.
   For the sending procedure, see Steps 1 to 5 of the data sending procedure described in Section 15.5.4, "Data Transmission in Master Mode." The slave select signal that has been asserted should be left unchanged.
2. Set the QSPI_*n*MADRH.RMADR[31:20] bits. (Remap external Flash memory)
3. Write 1 to the QSPI_*n*MMACFG2.MMAEN bit. (Enable memory mapped access mode)
4. Read the memory mapped access area for external Flash memory with an 8, 16, or 32-bit memory read instruction.
   This operation directly reads data within the 1M-byte external Flash memory area remapped to the memory mapped access area for external Flash memory at Step 2.
5. Repeat Step 4 as needed.
   When reading an address outside the remapped area, start from Step 2 again after setting the QSPI_*n*MMACFG2.MMAEN bit to 0 once.

Data receiving operations (32-bit read)

In memory mapped access mode, the internal state machine detects the address in the memory mapped access area from which data is read. If it is the first read operation after the QSPI has entered memory mapped access mode, the state machine generates an address cycle and a dummy cycle (including the XIP activation confirmation bit(s)). At the same time, it pulls the HREADY signal on the internal system bus down to low.

The address cycle can be configured for 24 or 32-bit addresses and it consists of two transfer cycles. The state machine determines actual Flash memory address from the memory mapped access area start address, the read address in that area, and the external Flash memory remapping start address set using the QSPI_*n*RMADRH[31:20] bits. The first transfer cycle is an 8-bit transfer that sends the high-order 8 bits of the address (when 24-bit address cycle is configured) or a 16-bit transfer that sends the high-order 16 bits of the address (when 32-bit address cycle is configured). The second cycle is fixed at 16-bit transfer that sends the low-order 16 bits of the address.

A dummy cycle follows. The XIP activation confirmation byte set in the QSPI_*n*MB.XIPACT[7:0] bits is sent at the beginning of the cycle.

Then, the state machine starts fetching data from the external Flash memory. Once 32-bit data has been fetched into the internal FIFO, the FIFO read level is incremented (FIFO data ready). At this time, the HREADY signal reverts to high and the data fetched into the FIFO is sent to the internal system bus. The state machine prefetches two more 32-bit data from the continuous address and stores it into the FIFO.

If the address in the memory mapped access area that is continuous to the previous read address is read when the FIFO contains the prefetched data (FIFO data ready status), the prefetched data is sent to the internal system bus with the HREADY signal held high (zero-wait read).

If an address in the memory mapped access area that is not continuous to the previous read address is read, the HREADY signal is pulled down to low immediately and the FIFO read level is cleared to 0 (empty status). The #QSPISS*n* signal is negated once for the period set in the QSPI_*n*MMACFG1.TCSH[3:0] bits and then asserted again. After that a new address cycle, dummy cycle, and data cycle are executed.
The beginning and the end of each address, dummy, or data cycle take a couple of HCLK clocks for handshaking.

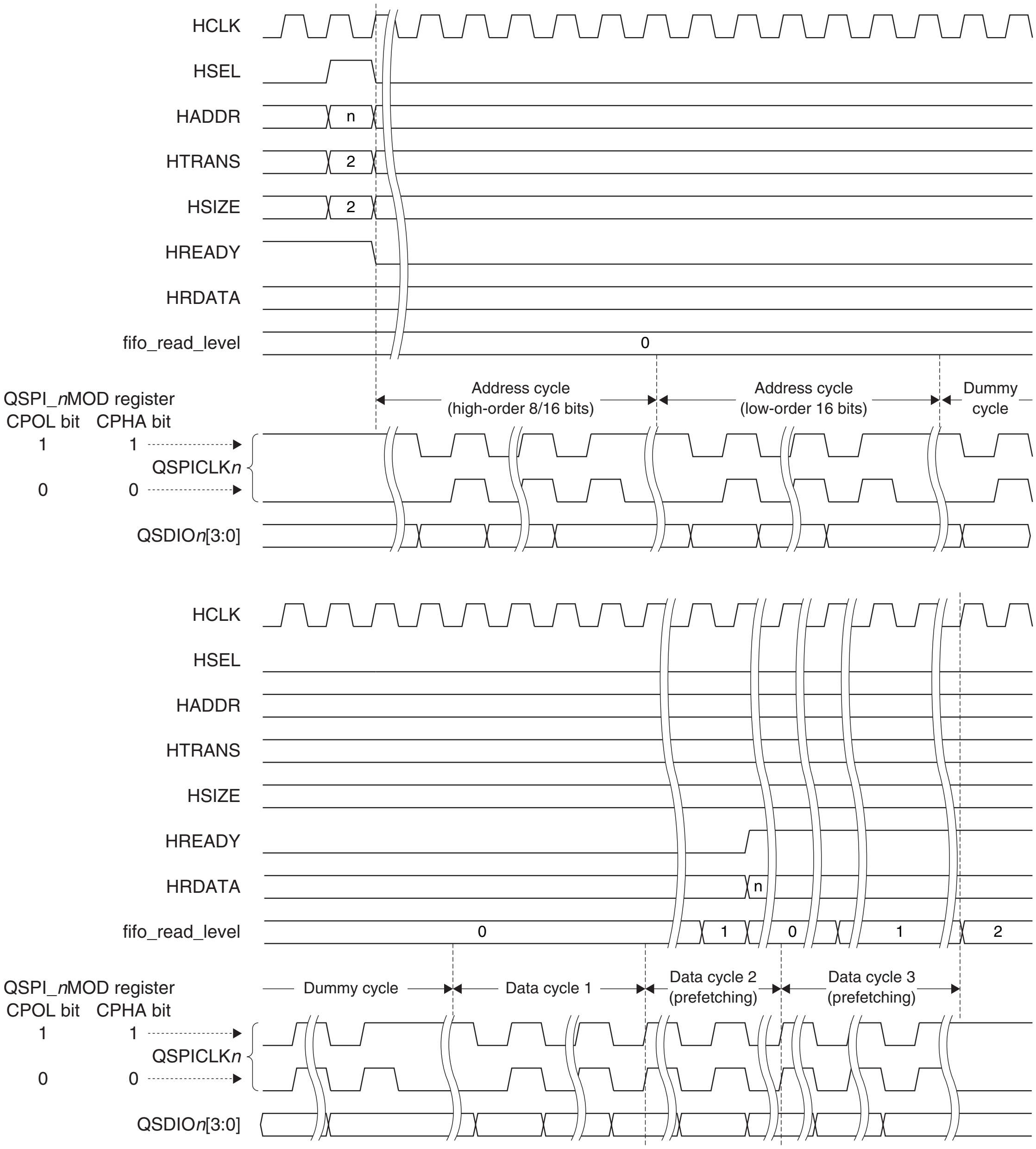


Figure 15.5.6.1 Data Receiving Operation in Memory Mapped Access Mode - First 32-bit Read

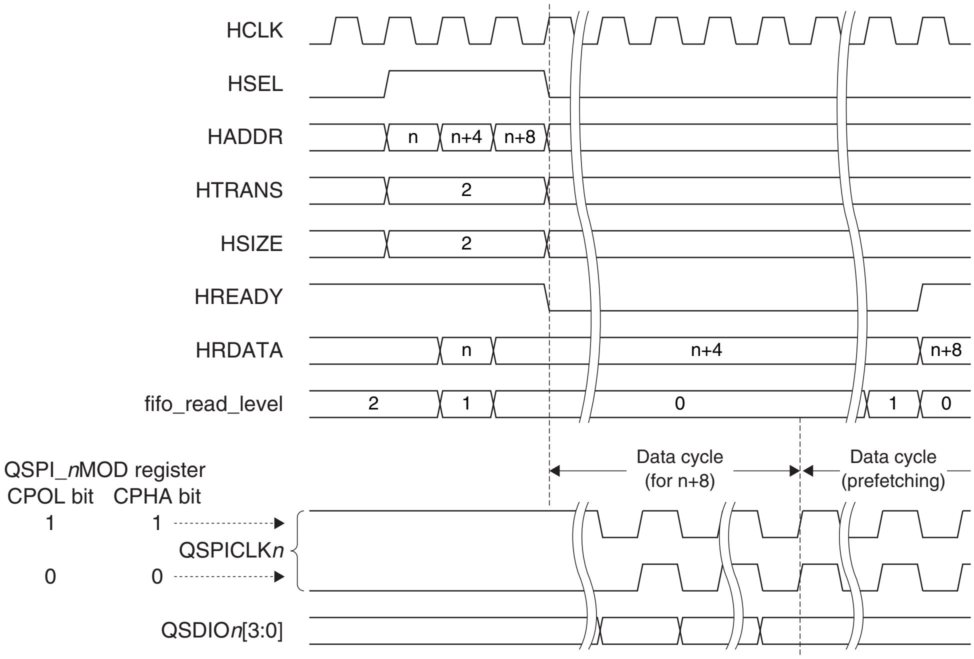


Figure 15.5.6.2 Data Receiving Operation in Memory Mapped Access Mode - 32-bit Sequential Read

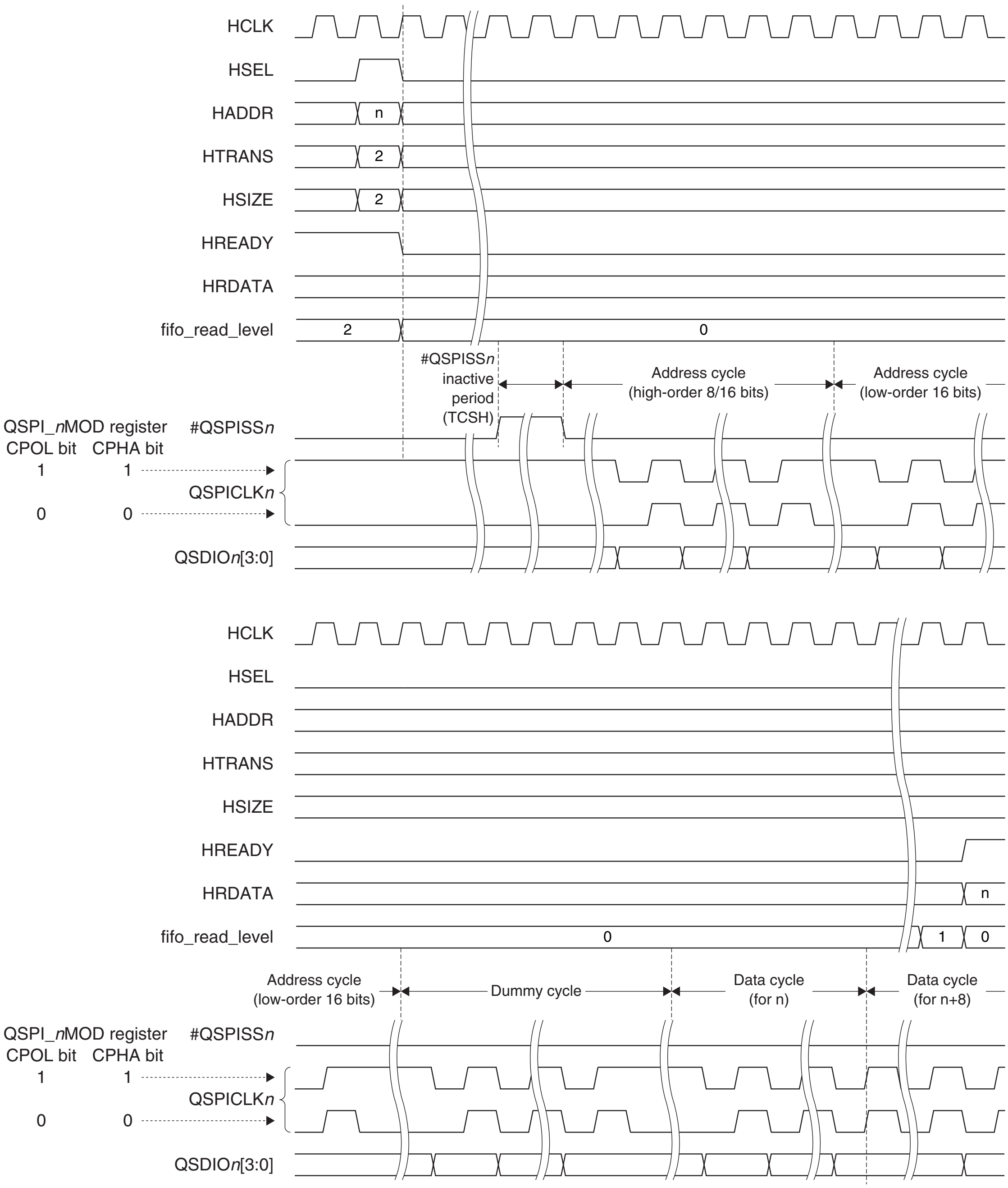


Figure 15.5.6.3 Data Receiving Operation in Memory Mapped Access Mode - 32-bit Non-Sequential Read

Data receiving operations (8/16-bit read)

The 8 and 16-bit read operations are the same as the 32-bit read operation except that data are not prefetched into the FIFO.

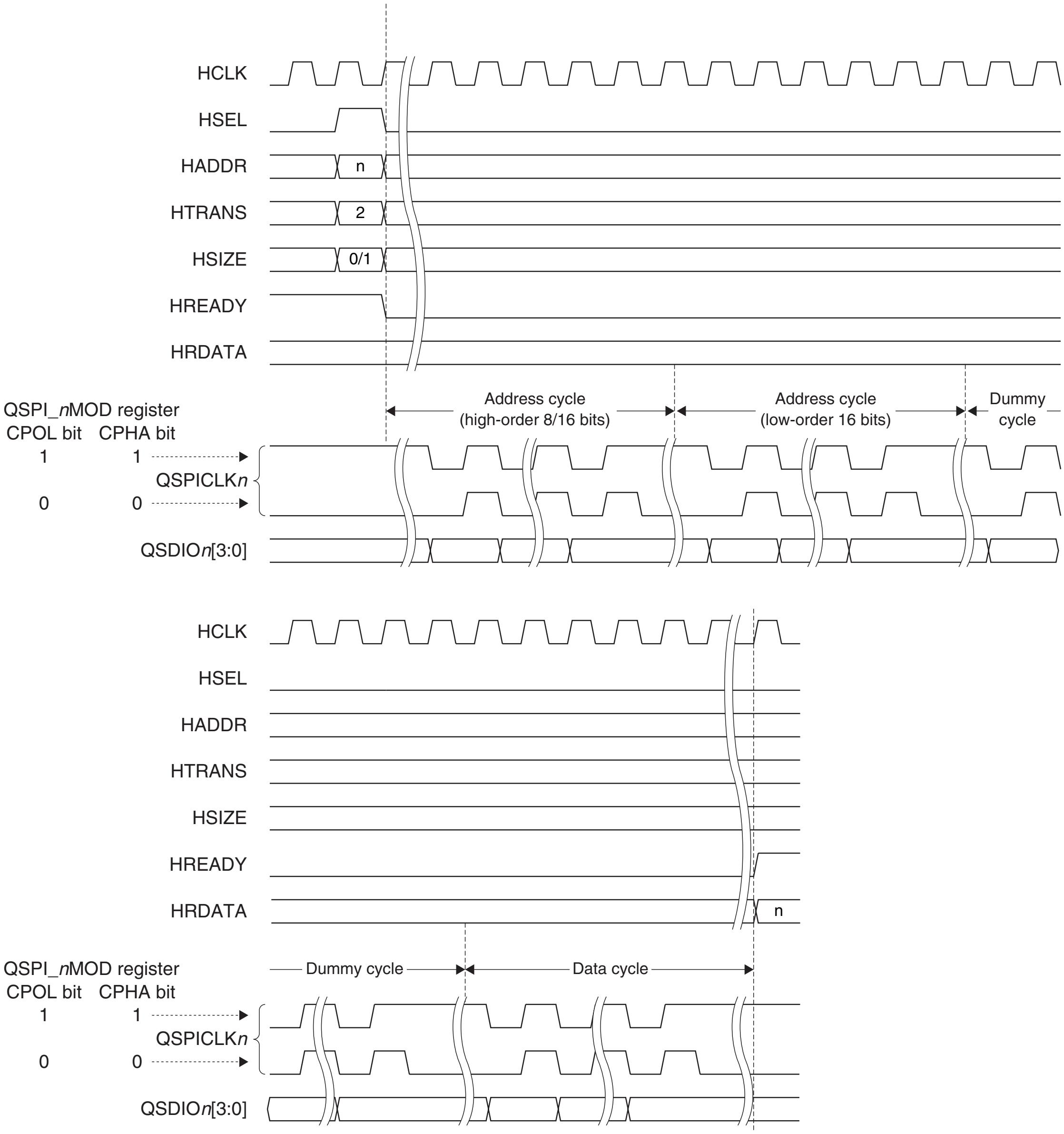


Figure 15.5.6.4 Data Receiving Operation in Memory Mapped Access Mode - First 8/16-bit Read

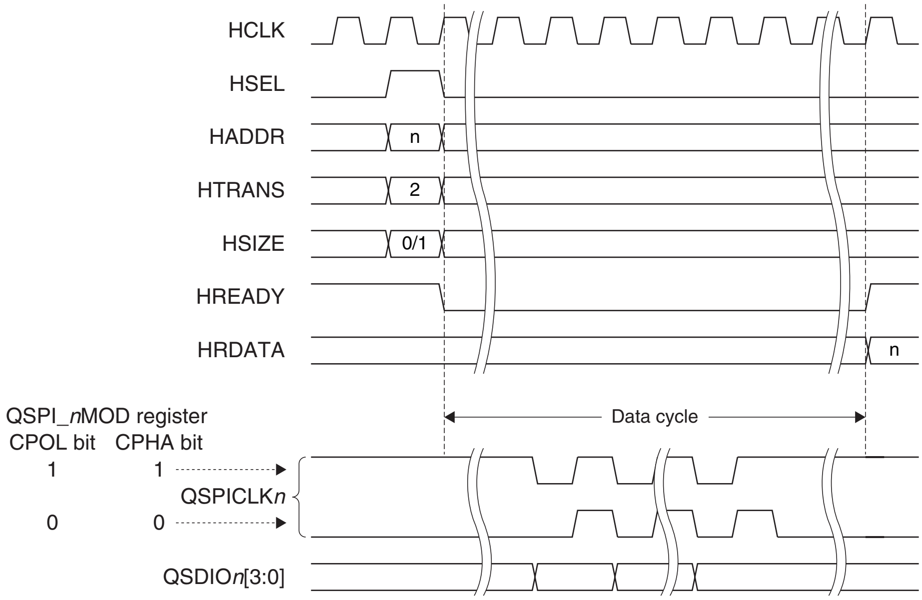


Figure 15.5.6.5 Data Receiving Operation in Memory Mapped Access Mode - 8/16-bit Sequential Read

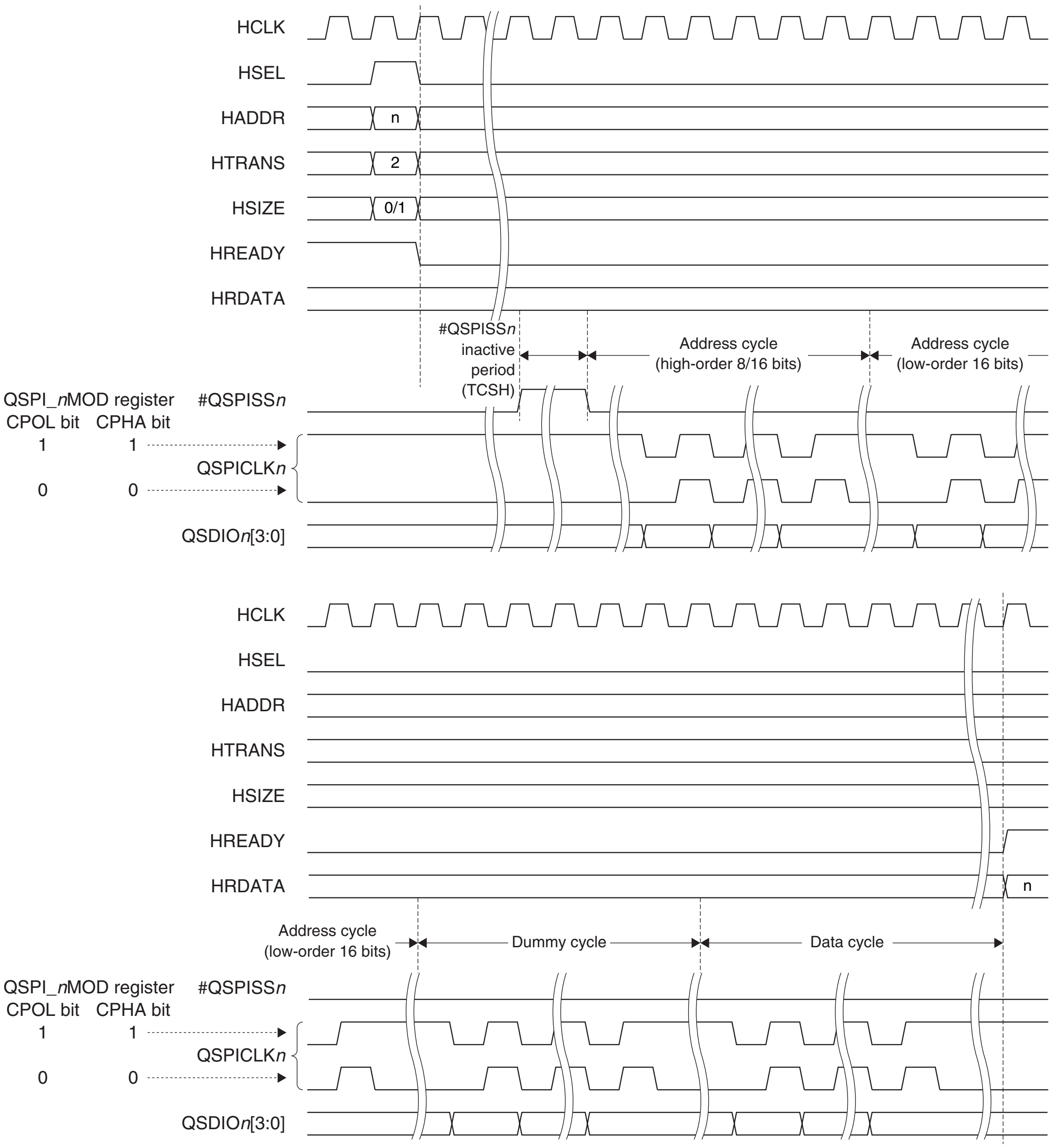


Figure 15.5.6.6 Data Receiving Operation in Memory Mapped Access Mode - 8/16-bit Non-Sequential Read

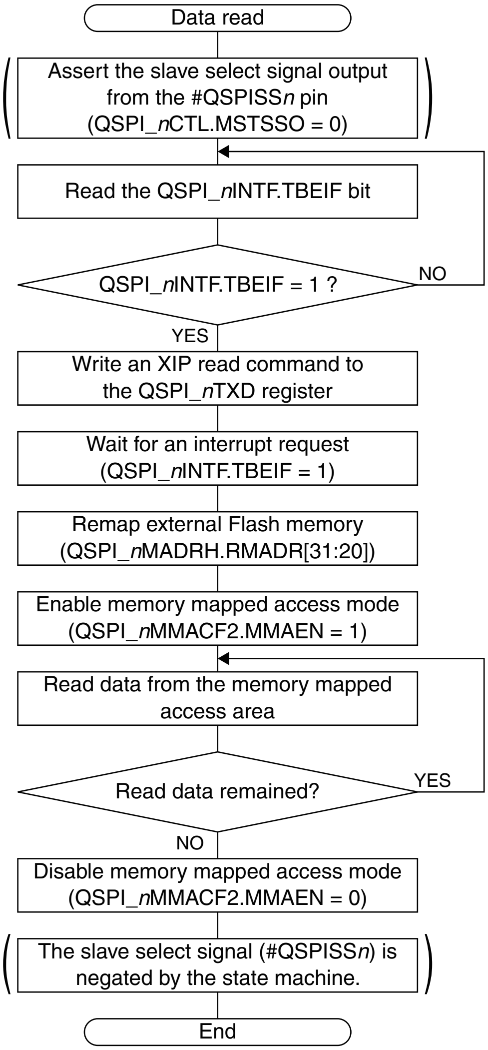


Figure 15.5.6.7 Data Reception Flowchart in Memory Mapped Access Mode

Data reception using DMA

In memory mapped access mode, DMA transfer from the external Flash memory to the internal memory is allowed only for the 32-bit sequential read using the internal FIFO. A non-sequential read and 8/16-bit reads cannot issue a DMA transfer request as they cannot use the FIFO.

By setting the QSPI_*n*FRLDMAEN.FRLDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and the external Flash memory data is transferred to the specified internal memory via DMA Ch.*x* when the FIFO read level is incremented (FIFO data ready flag is set). This function allows high-speed data block transfer as it does not need to execute read commands and uses the data pre-fetched into the FIFO.

Note, however, that the first data read must be performed via software or a software triggered DMA.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 15.5.6.1 DMA Data Structure Configuration Example
(for 32-bit Sequential Read in Memory Mapped Access Mode)

Item		Setting example
End pointer	Transfer source	External Flash memory transfer start address
	Transfer destination	Memory area start address from which the read data are stored
Control data	dst_inc	0x2 (+4)
	dst_size	0x2 (word)
	src_inc	0x2 (+4)
	src_size	0x2 (word)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of receive data
	cycle_ctrl	0x1 (basic transfer)

The following shows an example of the control procedure including the DMA controller operations:

1. Configure the primary data structure for the DMA channel (Ch.*x*) as shown in Table 15.5.6.1.
2. Enable the DMA channel using the DMA controller register.
3. Clear the channel request mask for the DMA channel using the DMA controller register.
4. Clear the DMA transfer completion interrupt flag using the DMA controller register.
5. Enable the DMA transfer completion interrupt of the DMA channel using the DMA controller register.
6. Clear pending DMA interrupts in the CPU.
7. Enable pending DMA interrupts in the CPU.
8. Enable the QSPI to issue DMA transfer requests to the DMA channel using the QSPI_*n*FRLDMAEN.FRLDMAEN*x* bit.
9. Issue a software DMA transfer request to the DMA channel by setting the DMA controller register. This operation is required to kickstart the first data fetching.
10. Wait for a DMA interrupt.
11. Disable DMA requests to be sent to the DMA channel using the QSPI_*n*FRLDMAEN.FRLDMAEN*x* bit.
12. Set the channel request masks for the DMA channel using the DMA controller register.
13. Disable the DMA channels using the DMA controller register.

15.5.7 Terminating Memory Mapped Access Operations

A procedure to terminate memory mapped access operations is shown below.

1. Write 0 to the QSPI_*n*MMACFG2.MMAEN bit. (Disable memory mapped access mode)
   The slave select signal is negated. Note that the slave signal control via software is disabled by the state machine in memory mapped access mode.
2. Wait until the QSPI_*n*INTF.MMABSY bit is set to 0 (memory mapped access operation not busy).

15.5.8 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 (QSPI_*n*INTF.TENDIF bit = 1).
2. Set the QSPI_*n*CTL.MODEN bit to 0 to disable the QSPI Ch.*n* operations.
3. Stop the 16-bit timer to disable the clock supply to QSPI Ch.*n*.

15.5.9 Data Transfer in Slave Mode

A data sending/receiving procedure and operations in slave mode are shown below. Figures 15.5.9.1 and 15.5.9.2 show a timing chart and flowcharts, respectively.

Data sending procedure

1. Check to see if the QSPI_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).
2. Write transmit data to the QSPI_*n*TXD register.
3. Wait for a transmit buffer empty interrupt (QSPI_*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 QSPI_*n*TXD register after the QSPI_*n*INTF.TBEIF bit is set to 1 by the time the sending QSPI_*n*TXD register data written is completed. If no transmit data is written during this period, the data bits input from the QSDIO*n* pins are shifted and output from the QSDIO*n* pins without being modified.

Data receiving procedure

1. Wait for a receive buffer full interrupt (QSPI_*n*INTF.RBFIF bit = 1).
2. Read the received data from the QSPI_*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 QSPI clock supplied from the external QSPI master to the QSPICLK*n* pin. The data transfer rate is determined by the QSPICLK*n* frequency. It is not necessary to control the 16-bit timer.
- QSPI can operate as a slave device only when the slave select signal input from the external QSPI master to the #QSPISS*n* pin is set to the active (low) level. If #QSPISS*n* = high, the software transfer control, the QSPICLK*n* pin input, and the QSDIO*n* pins input are all ineffective. If the #QSPISS*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 QSPICLK*n* is input from the external QSPI master after the #QSPISS*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 a QSPI interrupt.

Other operations are the same as master mode.

Notes:
- If data of the number of cycles specified by the QSPI_*n*MOD.CHLN[3:0] bits is received when the QSPI_*n*INTF.RBFIF bit is set to 1, the QSPI_*n*RXD register is overwritten with the newly received data and the previously received data is lost. In this case, the QSPI_*n*INTF.OEIF bit is set.
- When the clock for the first bit is input from the QSPICLK*n* pin, QSPI starts sending the data currently stored in the shift register even if the QSPI_*n*INTF.TBEIF bit is set to 1.

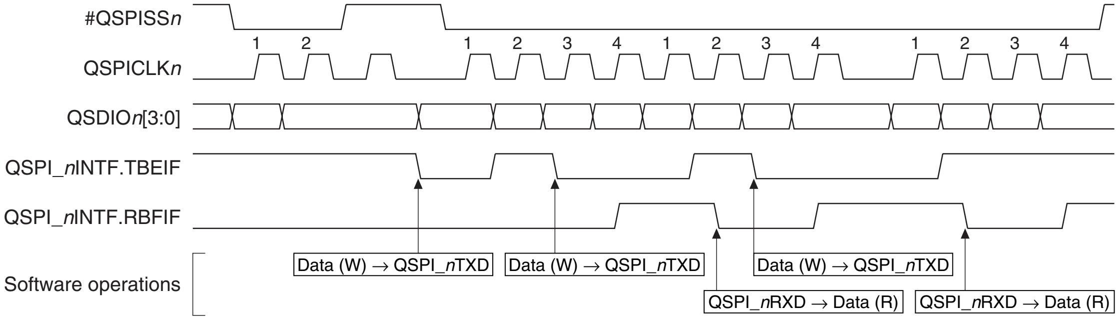


Figure 15.5.9.1 Example of Data Transfer Operations in Slave Mode
(QSPI_*n*MOD.CHDL[3:0] bits = QSPI_*n*MOD.CHLN[3:0] bits = 0x3)

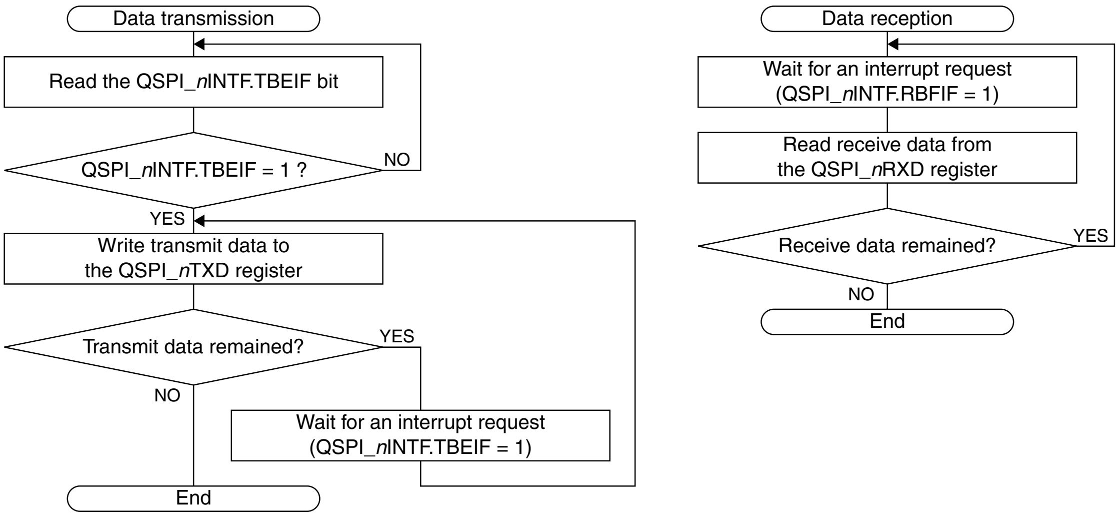


Figure 15.5.9.2 Data Transfer Flowcharts in Slave Mode

15.5.10 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 (QSPI_*n*INTF.TENDIF bit = 1). Or determine end of transfer via the received data.
2. Set the QSPI_*n*CTL.MODEN bit to 0 to disable the QSPI Ch.*n* operations.

15.6 Interrupts

The QSPI has a function to generate the interrupts shown in Table 15.6.1.

Table 15.6.1 QSPI Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
End of transmission	QSPI_*n*INTF.TENDIF	When the QSPI_*n*INTF.TBEIF bit = 1 after data of the specified bit length (defined by the QSPI_*n*MOD.CHLN[3:0] bits) has been sent	Writing 1
Receive buffer full	QSPI_*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 of the QSPI_*n*RXD register
Transmit buffer empty	QSPI_*n*INTF.TBEIF	When transmit data written to the transmit data buffer is transferred to the shift register	Writing to the QSPI_*n*TXD register
Overrun error	QSPI_*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

The QSPI provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU 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” chapter.

The QSPI_*n*INTF register also contains the BSY and MMABSY bits that indicate the QSPI operating status in register access and memory mapped access modes, respectively. Figure 15.6.1 shows the QSPI_*n*INTF.BSY, QSPI_*n*INTF.MMABSY and QSPI_*n*INTF.TENDIF bit set timings.

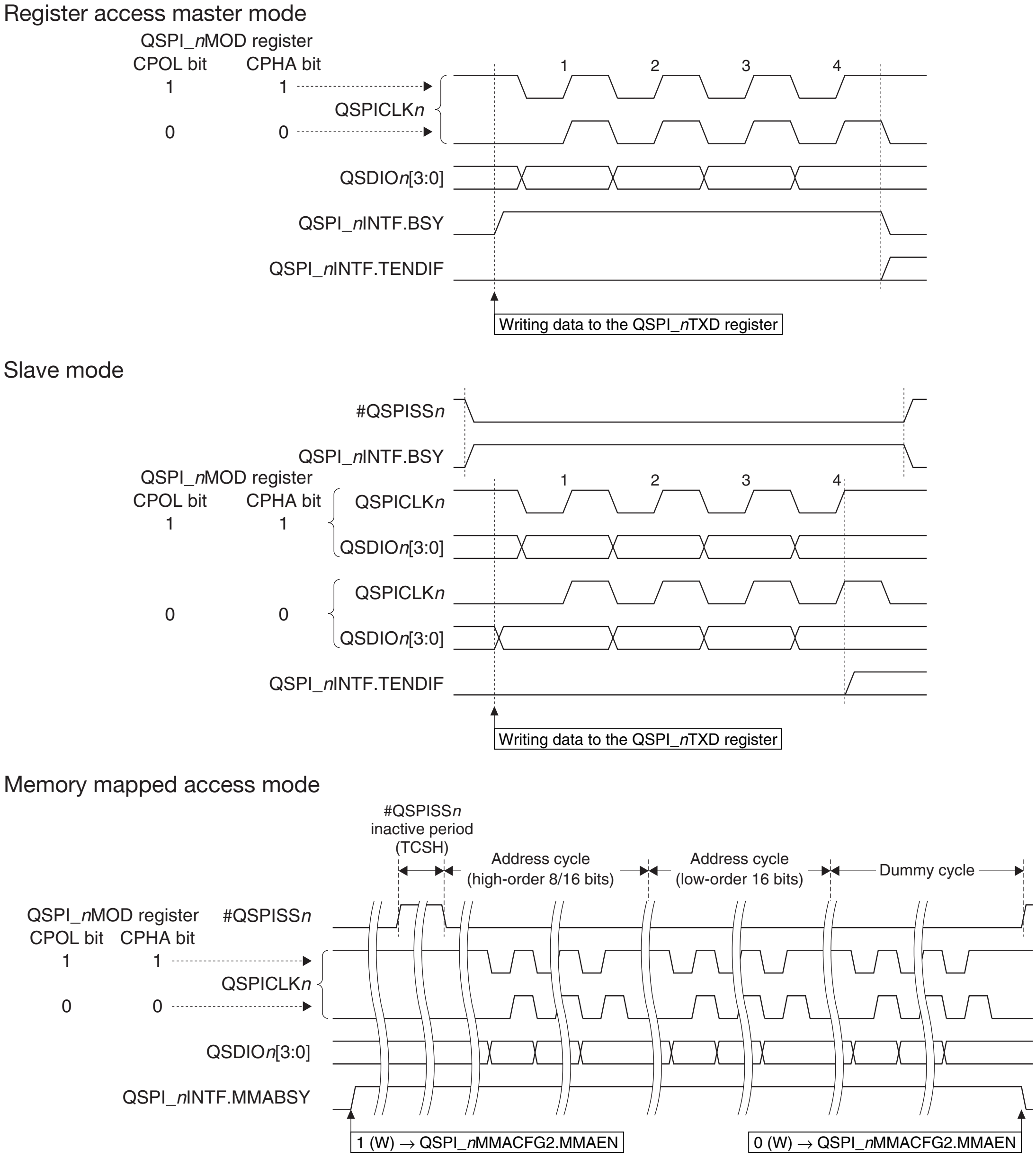


Figure 15.6.1 QSPI_nINTF.BSY, QSPI_nINTF.MMABSY, and QSPI_nINTF.TENDIF Bit Set Timings
(when QSPI_nMOD.CHDL[3:0] bits = QSPI_nMOD.CHLN[3:0] bits = 0x3)

15.7 DMA Transfer Requests

The QSPI has a function to generate DMA transfer requests from the causes shown in Table 15.7.1.

Table 15.7.1 DMA Transfer Request Causes of QSPI

Cause to request DMA transfer	DMA transfer request flag	Set condition	Clear condition
Receive buffer full	Receive buffer full flag (QSPI_nINTF.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 of the QSPI_nRXD register
Transmit buffer empty	Transmit buffer empty flag (QSPI_nINTF.TBEIF)	When transmit data written to the transmit data buffer is transferred to the shift register	Writing to the QSPI_nTXD register
Memory mapped access FIFO data ready	Memory mapped access FIFO data ready flag (internal signal)	When a 32-bit data is prefetched into the FIFO in memory mapped access mode	When the FIFO read level is cleared to 0

The QSPI provides DMA transfer request enable bits corresponding to each DMA transfer request flag shown above for the number of DMA channels. A DMA transfer request is sent to the pertinent channel of the DMA controller only when the DMA transfer request flag, of which DMA transfer has been enabled by the DMA transfer request enable bit, is set. The receive buffer full and transmit buffer empty DMA transfer request flags also serve as interrupt flags, therefore, both the DMA transfer request and the interrupt cannot be enabled at the same time. After a DMA transfer has completed, disable the DMA transfer to prevent unintended DMA transfer requests from being issued. For more information on the DMA control, refer to the "DMA Controller" chapter.

15.8 Control Registers

QSPI Ch.*n* Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*MOD	15–12	CHDL[3:0]	0x7	H0	R/W	–
	11–8	CHLN[3:0]	0x7	H0	R/W	
	7–6	TMOD[1:0]	0x0	H0	R/W	
	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 CHDL[3:0]

These bits set the number of clocks to drive the serial output data lines.

Table 15.8.1 Data Line Drive Length Settings

QSPI_*n*MOD.CHDL[3:0] bits	Data line drive length
0xf	16 clocks
0xe	15 clocks
0xd	14 clocks
0xc	13 clocks
0xb	12 clocks
0xa	11 clocks
0x9	10 clocks
0x8	9 clocks
0x7	8 clocks
0x6	7 clocks
0x5	6 clocks
0x4	5 clocks
0x3	4 clocks
0x2	3 clocks
0x1	2 clocks
0x0	1 clock

These bits must be set to a value smaller than or equal to the QSPI_*n*MOD.CHLN[3:0] bit setting.

Note: When using the QSPI in slave mode, the QSPI_*n*MOD.CHDL[3:0] bits should be set to the same value as the QSPI_*n*MOD.CHLN[3:0] bits.

Bits 11–8 CHLN[3:0]

These bits set the number of clocks for data transfer.

Table 15.8.2 Setting of Number of Data Transfer Clocks

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

Bits 7–6 TMOD[1:0]

These bits select a transfer mode.

Table 15.8.3 Transfer Mode

QSPI_*n*MOD. TMOD[1:0] bits	Transfer mode
0x3	Reserved
0x2	Quad transfer mode The QSDIO*n*[3:0] pins are configured as input or output pins according to the QSPI_*n*MOD.DIR bit setting.
0x1	Dual transfer mode The QSDIO*n*[1:0] pins are configured as input or output pins according to the QSPI_*n*MOD.DIR bit setting. The QSDIO*n*[3:2] pins are not used.
0x0	Single transfer mode The QSDIO*n*0 and QSDIO*n*1 pins are configured as an output pin and an input pin, respectively. The QSDIO*n*[3:2] pins are not used.

Bit 5 PUEN

This bit enables pull-up/down of the pins that are configured as an input or are not used.

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 QSPICLK*n* in master mode. This setting is ineffective in slave mode.

1 (R/W): QSPICLK*n* frequency = CLK_QSPI*n* frequency (= 16-bit timer operating clock frequency)

0 (R/W): QSPICLK*n* frequency = 16-bit timer output frequency / 2

For more information, refer to “QSPI 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 QSPI clock phase and polarity. For more information, refer to “QSPI Clock (QSPI-CLK*n*) Phase and Polarity.”

Bit 0 MST

This bit sets the QSPI operating mode (master mode or slave mode).

1 (R/W): Master mode

0 (R/W): Slave mode

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

QSPI Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*CTL	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3	DIR	0	H0	R/W	
	2	MSTSSO	1	H0	R/W	
	1	SFTRST	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bits 15–4 Reserved

Bit 3 DIR

This bit sets the data transfer direction on the QSDIO*n*[3:0] lines when the QSPI_*n*MOD.TMOD[1:0] bits are set to 1 or 2.

1 (R/W): Input

0 (R/W): Output

Bit 2 MSTSSO

This bit controls and indicates the #QSPISS*n* pin status.

1 (R/W): #QSPISS*n* = high (The device is deselected.)

0 (R/W): #QSPISS*n* = low (The device is selected.)

In memory mapped access mode, the #QSPISS*n* pin is automatically controlled by the internal state machine. Reading this bit allows monitoring of the current #QSPISS*n* pin output status at any time.

Bit 1 SFTRST

This bit issues software reset to QSPI.

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

Bit 0 MODEN

This bit enables the QSPI operations.

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

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

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

QSPI Ch.*n* Transmit Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*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. Writing to these bits starts data transfer. Transmit data can be written when the QSPI_*n*INTF.TBEIF bit = 1 regardless of whether data is being output from the QSDIO*n* pins or not.

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

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

QSPI Ch.*n* Receive Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*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 QSPI_*n*INTF.RBFIF bit = 1 regardless of whether data is being input from the QSDIO*n* pin or not.

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

QSPI Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*INTF	15–8	–	0x00	–	R	–
	7	BSY	0	H0	R	
	6	MMABSY	0	H0	R	
	5–4	–	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 QSPI_*n*RXD register.
	0	TBEIF	1	H0/S0	R	Cleared by writing to the QSPI_*n*TXD register.

Bits 15–8 Reserved

Bit 7 BSY

This bit indicates the QSPI operating status.

1 (R): Transmit/receive busy

0 (R): Idle

Bit 6 MMABSY

This bit indicates the QSPI memory mapped access operating status.

1 (R): Memory mapped access state machine busy

0 (R): Idle

Bits 5–4 Reserved

Bit 3 OEIF
Bit 2 TENDIF
Bit 1 RBFIF
Bit 0 TBEIF

These bits indicate the QSPI 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:
QSPI_*n*INTF.OEIF bit: Overrun error interrupt
QSPI_*n*INTF.TENDIF bit: End-of-transmission interrupt
QSPI_*n*INTF.RBFIF bit: Receive buffer full interrupt
QSPI_*n*INTF.TBEIF bit: Transmit buffer empty interrupt

QSPI Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*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 QSPI interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
QSPI_*n*INTE.OEIE bit: Overrun error interrupt
QSPI_*n*INTE.TENDIE bit: End-of-transmission interrupt
QSPI_*n*INTE.RBFIE bit: Receive buffer full interrupt
QSPI_*n*INTE.TBEIE bit: Transmit buffer empty interrupt

QSPI Ch.*n* Transmit Buffer Empty DMA Request Enable Register

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

Bits 15–0 TBEDMAEN[15:0]

These bits enable the QSPI to issue a DMA transfer request to the corresponding DMA channel (Ch.0–Ch.15) when a transmit buffer empty state has occurred.
1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

QSPI Ch.*n* Receive Buffer Full DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*RBFDMAEN	15–0	RBFDMAEN[15:0]	0x0000	–	R/W	–

Bits 15–0 RBFDMAEN[15:0]

These bits enable the QSPI to issue a DMA transfer request to the corresponding DMA channel (Ch.0–Ch.15) when a receive buffer full state has occurred.

1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

QSPI Ch.n FIFO Data Ready DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*FRLDMAEN	15–8	FRLDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 FRLDMAEN[15:0]

These bits enable the QSPI to issue a DMA transfer request to the corresponding DMA channel (Ch.0–Ch.15) when data is prefetched into the FIFO (FIFO data ready).

1 (R/W): Enable DMA transfer request
0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

QSPI Ch.*n* Memory Mapped Access Configuration Register 1

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*MMACFG1	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3–0	TCSH[3:0]	0x0	H0	R/W	

Bits 15–4 Reserved

Bits 3–0 TCSH[3:0]

When non-sequential reading from a Flash memory address, which is not continuous to the previous read address, occurs in memory mapped access mode, the #QSPISS*n* signal is reasserted after negated once. Then the new address is sent to the Flash memory before reading data.

The QSPI_*n*MMACFG1.TCSH[3:0] bits specify the period to negate the #QSPISS*n* signal at this time in a number of clocks.

Table 15.8.4 #QSPISS*n* Inactive Period between Non-Sequential Readings

QSPI_*n*MMACFG1.TCSH[3:0] bits	#QSPISS*n* Inactive Period
0xf	16 clocks
0xe	15 clocks
0xd	14 clocks
0xc	13 clocks
0xb	12 clocks
0xa	11 clocks
0x9	10 clocks
0x8	9 clocks
0x7	8 clocks
0x6	7 clocks
0x5	6 clocks
0x4	5 clocks
0x3	4 clocks
0x2	3 clocks
0x1	2 clocks
0x0	1 clock

Note: These bits specify a number of system clocks.

QSPI Ch.*n* Remapping Start Address High Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*RMADRH	15–4	RMADR[31:20]	0x000	H0	R/W	–
	3-0	–	0x0	–	R	

Bits 15–4 RMADR[31:20]

These bits specify the high-order 12 bits of the external Flash memory area start address (assumed as 32 bits) to be remapped to the system memory area allocated for memory mapped access mode. When the external Flash memory is read using the memory mapped access function, the value specified here is added, as an offset, to the relative address in the memory mapped access area to generate the external Flash memory address to actually be accessed.

Note: Make sure the QSPI_*n*MMACFG2.MMAEN = 0 when altering the QSPI_*n*RMADRH. RMADR[31:20] bits.

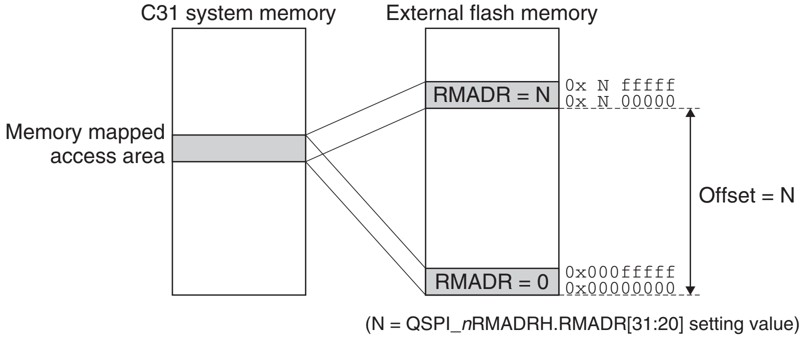


Figure 15.8.1 External Flash Memory Remapping

Bits 3–0 Reserved

QSPI Ch.*n* Memory Mapped Access Configuration Register 2

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*MMACFG2	15–12	DUMDL[3:0]	0x7	H0	R/W	–
	11–8	DUMLN[3:0]	0x7	H0	R/W	
	7–6	DATTMOD[1:0]	0x0	H0	R/W	
	5–4	DUMTMOD[1:0]	0x0	H0	R/W	
	3–2	ADRTMOD[1:0]	0x0	H0	R/W	
	1	ADRCYC	0	H0	R/W	
	0	MMAEN	0	H0	R/W	

Bits 15–12 DUMDL[3:0]

These bits set the number of clocks for driving the serial data lines during the dummy cycle output when accessing the external Flash memory in the memory mapped access mode. This setting is required to output the XIP confirmation bit to Micron Flash memories or to output the mode byte to Spansion Flash memories.

Table 15.8.5 Settings of Data Line Drive Length during Dummy Cycle

QSPI_*n*MMACFG2.DUMDL[3:0] bits	Data line drive length
0xf	16 clocks
0xe	15 clocks
0xd	14 clocks
0xc	13 clocks
0xb	12 clocks
0xa	11 clocks
0x9	10 clocks
0x8	9 clocks
0x7	8 clocks
0x6	7 clocks
0x5	6 clocks
0x4	5 clocks
0x3	4 clocks
0x2	3 clocks
0x1	2 clocks
0x0	1 clock

These bits must be set to a value smaller than or equal to the QSPI_*n*MMACFG2.DUMLN[3:0] bit setting.

Bits 11–8 DUMLN[3:0]

These bits set the dummy cycle length in a number of clocks when accessing the external Flash memory in the memory mapped access mode.

Table 15.8.6 Dummy Cycle Length Settings

QSPI_*n*MMACFG2.DUMLN[3:0] bits	Dummy cycle length
0xf	16 clocks
0xe	15 clocks
0xd	14 clocks
0xc	13 clocks
0xb	12 clocks
0xa	11 clocks
0x9	10 clocks
0x8	9 clocks
0x7	8 clocks
0x6	7 clocks
0x5	6 clocks
0x4	5 clocks
0x3	4 clocks
0x2	3 clocks
0x1	2 clocks
0x0	Setting prohibited

Bits 7–6 DATTMOD[1:0]

These bits select the transfer mode for the data cycle when accessing the external Flash memory in the memory mapped access mode.

Table 15.8.7 Transfer Mode for Data, Dummy, and Address Cycles

QSPI_*n*MMACFG2.DATTMOD[1:0] bits QSPI_*n*MMACFG2.DUMTMOD[1:0] bits QSPI_*n*MMACFG2.ADRTMOD[1:0] bits	Transfer mode
0x3	Reserved
0x2	Quad transfer mode The QSDIO*n*[3:0] pins are used.
0x1	Dual transfer mode The QSDIO*n*[1:0] pins are used. The QSDIO*n*[3:2] pins are not used.
0x0	Single transfer mode The QSDIO*n*[1:0] pins are used. The QSDIO*n*[3:2] pins are not used.

Bits 5–4 DUMTMOD[1:0]

These bits select the transfer mode for the dummy cycle when accessing the external Flash memory in the memory mapped access mode.

Bits 3–2 ADRTMOD[1:0]

These bits select the transfer mode for the address cycle when accessing the external Flash memory in the memory mapped access mode.

Bit 1 ADRCYC

This bit selects the address mode from 24 and 32 bits when accessing the external Flash memory in the memory mapped access mode.

1 (R/W): 32-bit address mode (4-byte address cycle)
0 (R/W): 24-bit address mode (3-byte address cycle)

Bit 0 MMAEN

This bit enables memory mapped access mode for accessing the external Flash memory.

1 (R/W): Enable memory mapped access mode
0 (R/W): Disable memory mapped access mode (register access mode)

When this bit is altered from 1 to 0, the QSPI sends extra address and dummy cycles to the external Flash memory. The address cycle outputs either a three or four-byte address according to the QSPI_*n*MMACFG2.ADRCYC bit setting, with all address bits set to 1. The dummy cycle is output according to the QSPI_*n*MMACFG2.DUMLN[3:0] and QSPI_*n*MMACFG2.DUMDL[3:0] bit settings, with a mode byte for terminating the XIP session of the external Flash memory that has been configured using the QSPI_*n*MB.XIPEXT[7:0] bits.

Note: Slave mode does not support memory mapped access mode, therefore, setting the QSPI_*n*MMACFG2.MMAEN bit to 1 does not take effect when the QSPI_*n*MOD.MST bit = 0.

QSPI Ch.*n* Mode Byte Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
QSPI_*n*MB	15–8	XIPACT[7:0]	0x00	H0	R/W	–
	7–0	XIPEXT[7:0]	0x00	H0	R/W	

Bits 15–8 XIPACT[7:0]

These bits configure the mode byte for activating an XIP session of the external Flash memory to be accessed in memory mapped access mode.

Bits 7–0 XIPEXT[7:0]

These bits configure the mode byte for terminating the XIP session of the external Flash memory being accessed in memory mapped access mode.
However, set these bits as follows when the HW processor (HWP) is used:

- Before enabling the HWP, set to the same value as the QSPI_*n*MB.XIPACT[7:0] bits.
- Before disabling the HWP, set to the mode byte for terminating the XIP session.

Note: In memory mapped access mode, the mode byte is always output from the LSB first. When using a Flash memory that expects the mode byte to be output from the MSB first, write the mode byte to this register in reverse bit order.

16 I²C (I2C)

16.1 Overview

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

- Functions as an I²C bus master (single master) or a slave device.
- Supports standard mode (up to 100 kbit/s) and fast mode (up to 400 kbit/s).
- Supports 7-bit and 10-bit address modes.
- Supports clock stretching.
- Includes a baud rate generator for generating the clock in master mode.
- No clock source is required to run the I2C in slave mode, as it can run with the I²C bus signals only.
- Slave mode is capable of being operated in SLEEP mode allowing wake-up by an interrupt when an address match is detected.
- Master mode supports automatic bus clear sending function.
- Can generate receive buffer full, transmit buffer empty, and other interrupts.
- Can issue a DMA transfer request when a receive buffer full or a transmit buffer empty occurs.
- The input filter for the SDA and SCL inputs does not comply with the standard for removing noise spikes less than 50 ns.

Figure 16.1.1 shows the I2C configuration.

Table 16.1.1 I2C Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	3 channels (Ch.0 to Ch.2)		

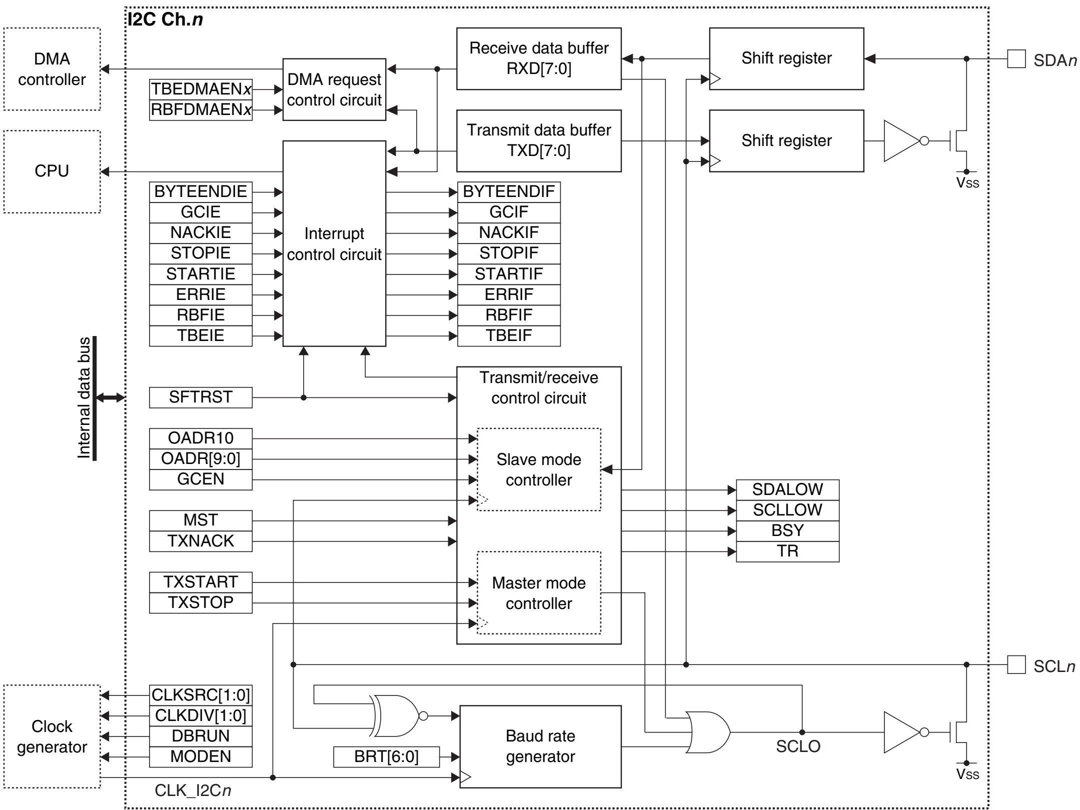


Figure 16.1.1 I2C Configuration

16.2 Input/Output Pins and External Connections

16.2.1 List of Input/Output Pins

Table 16.2.1.1 lists the I2C pins.

Table 16.2.1.1 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.

16.2.2 External Connections

Figure 16.2.2.1 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.

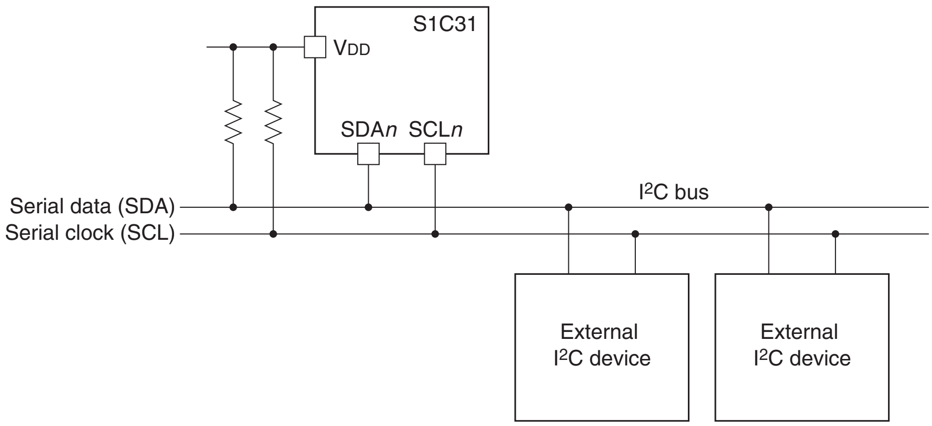


Figure 16.2.2.1 Connections between I2C and External I²C Devices

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.

16.3 Clock Settings

16.3.1 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 I^2C 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 I^2C master even if all the internal clocks halt during SLEEP mode, so the I2C can receive data and can generate receive buffer full interrupts.

16.3.2 Clock Supply During Debugging

In master mode, the CLK_I2C*n* supply during debugging 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 state if the I2C_*n*CLK.DBRUN bit = 0. After the CPU returns to normal operation, 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 state 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 state.
In slave mode, the I2C Ch.*n* operates with the external I^2C master clock input from the SCL*n* pin regardless of whether the CPU is placed into debug state or normal operation state.

16.3.3 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.

$$bps = \frac{f_{CLK\_I2Cn}}{(BRT + 3) \times 2} \qquad BRT = \frac{f_{CLK\_I2Cn}}{bps \times 2} - 3 \qquad \text{(Eq. 16.1)}$$

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 16.3.3.1).

Note: The I^2C 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 16.3.3.1 shows the clock generated by the baud rate generator and the clock waveform on the I²C bus.

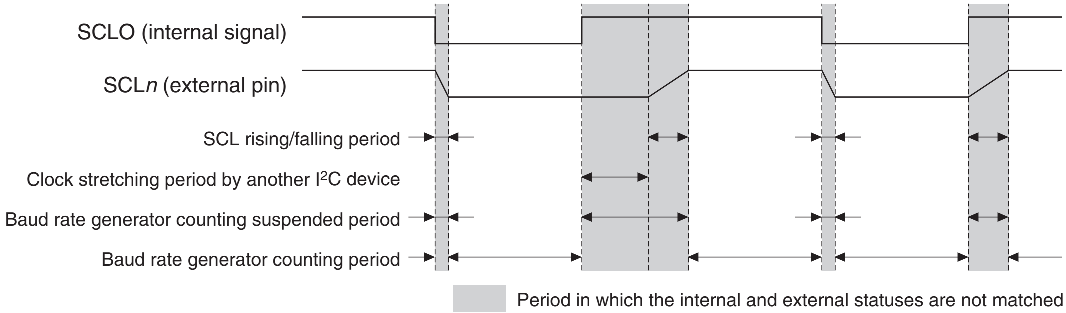


Figure 16.3.3.1 Baud Rate Generator Output Clock and SCL*n* Output Waveform

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.

16.4 Operations

16.4.1 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)

16.4.2 Data Transmission in Master Mode

A data sending procedure in master mode and the I2C Ch.*n* operations are shown below. Figures 16.4.2.1 and 16.4.2.2 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. (When DMA is used) Configure the DMA controller and set a DMA transfer request enable bit in the I2C_*n*TBEDMAEN register to 1 (DMA transfer request enabled). (This automates the data sending procedure Steps 5, 6, and 8.)
5. (When DMA is not used) Wait for a transmit buffer empty interrupt (I2C_*n*INTF.TBEIF bit = 1) generated when an ACK is received.
6. (When DMA is not used) Write transmit data to the I2C_*n*TXD register.
7. If a NACK reception interrupt (I2C_*n*INTF.NACKIF bit = 1) has occurred, go to Step 9 or 1 after clearing the I2C_*n*INTF.NACKIF bit.
8. (When DMA is not used) Repeat Steps 5 and 6 until the end of transmit data.
9. Issue a STOP condition by setting the I2C_*n*CTL.TXSTOP bit to 1.
10. 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 I^2C 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 I^2C 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.

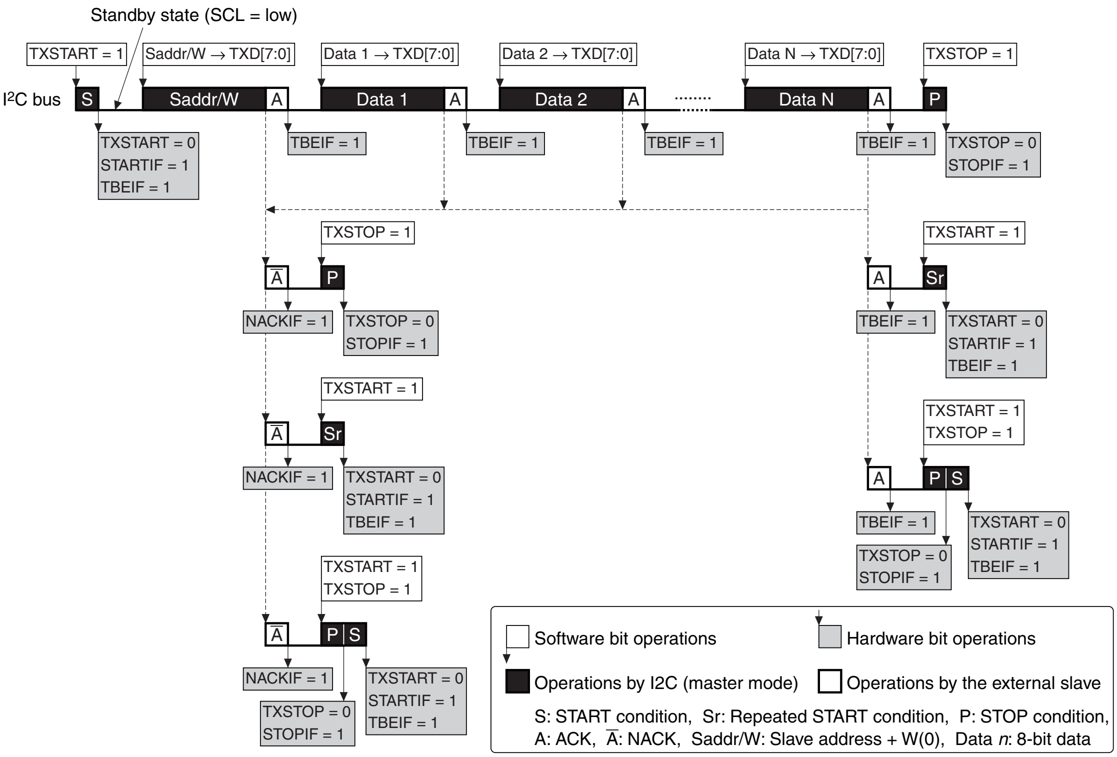


Figure 16.4.2.1 Example of Data Sending Operations in Master Mode

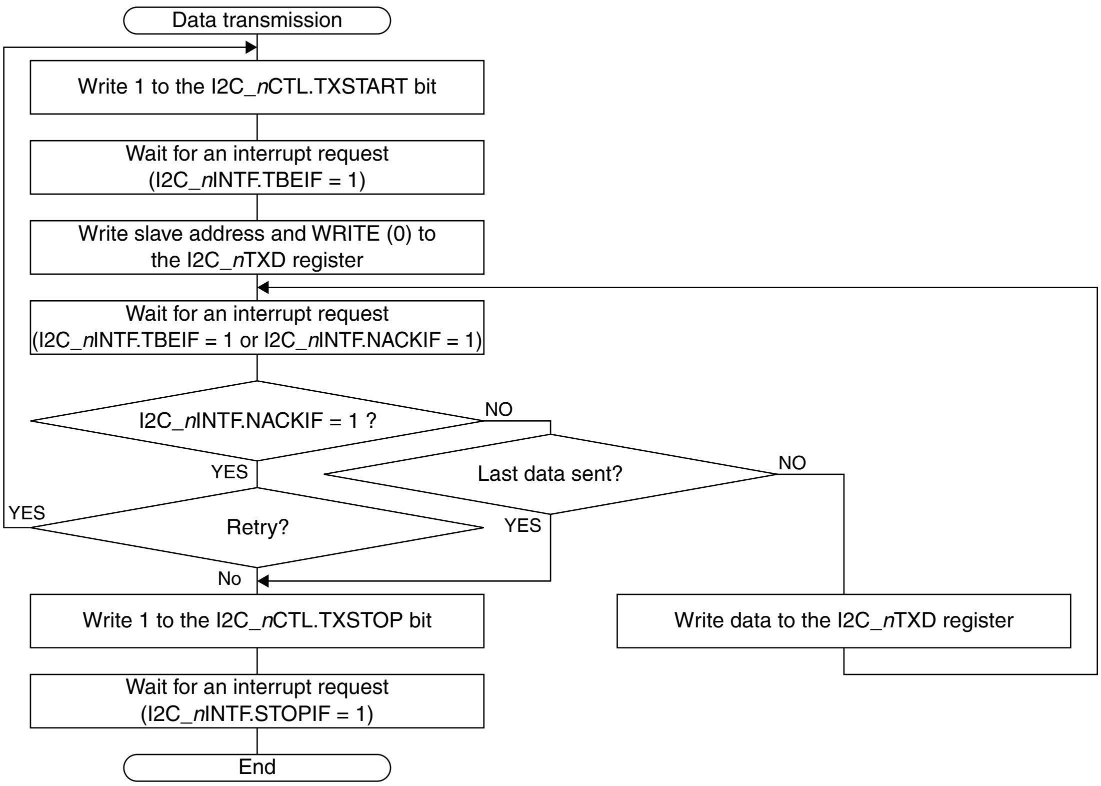


Figure 16.4.2.2 Master Mode Data Transmission Flowchart

Data transmission using DMA

By setting the I2C_*n*TBEDMAEN.TBEDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and transmit data is transferred from the specified memory to the I2C_*n*TXD register via DMA Ch.*x* when the I2C_*n*INTF.TBEIF bit is set to 1 (transmit buffer empty).

This automates the data sending procedure from Steps 5, 6, and 8 described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance so that transmit data will be transferred to the I2C_*n*TXD register. For more information on DMA, refer to the "DMA Controller" chapter.

Table 16.4.2.1 DMA Data Structure Configuration Example (for Data Transmission)

Item		Setting example
End pointer	Transfer source	Memory address in which the last transmit data is stored
	Transfer destination	I2C_*n*TXD register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x0 (byte)
	src_inc	0x0 (+1)
	src_size	0x0 (byte)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

16.4.3 Data Reception in Master Mode

A data receiving procedure in master mode and the I2C Ch.*n* operations are shown below. Figures 16.4.3.1 and 16.4.3.2 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. (When DMA is used) Configure the DMA controller and set a DMA transfer request enable bit in the I2C_*n*RBFDMAEN register to 1 (DMA transfer request enabled). (This automates the data receiving procedure Steps 6, 8, and 10.)
6. (When DMA is not used) Wait for a receive buffer full interrupt (I2C_*n*INTF.RBFIF bit = 1) generated when a one-byte reception has completed.
7. 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 8.
   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 11.
8. (When DMA is not used) Read the received data from the I2C_*n*RXD register.
9. If a NACK reception interrupt (I2C_*n*INTF.NACKIF bit = 1) has occurred, clear the I2C_*n*INTF.NACKIF bit and issue a STOP condition by setting the I2C_*n*CTL.TXSTOP bit to 1. Then go to Step 11 or Step 2 if making a retry.
10. (When DMA is not used) Repeat Steps 6 to 8 until the end of data reception.
11. 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.

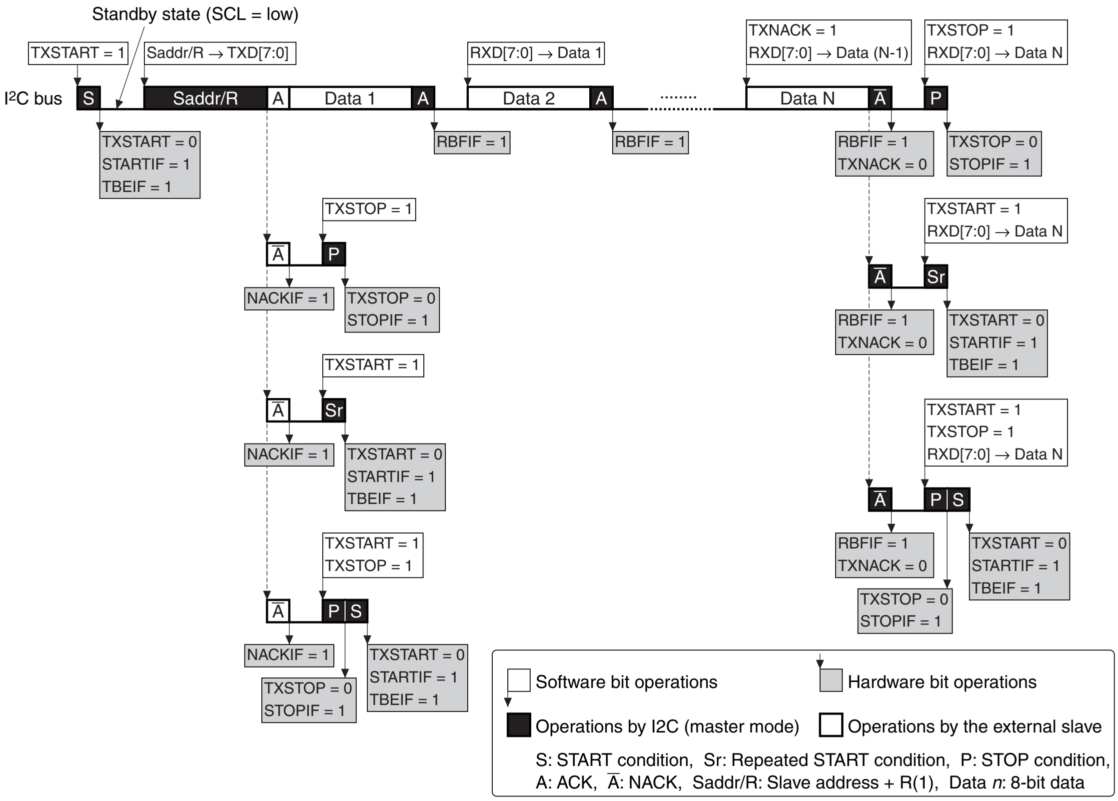


Figure 16.4.3.1 Example of Data Receiving Operations in Master Mode

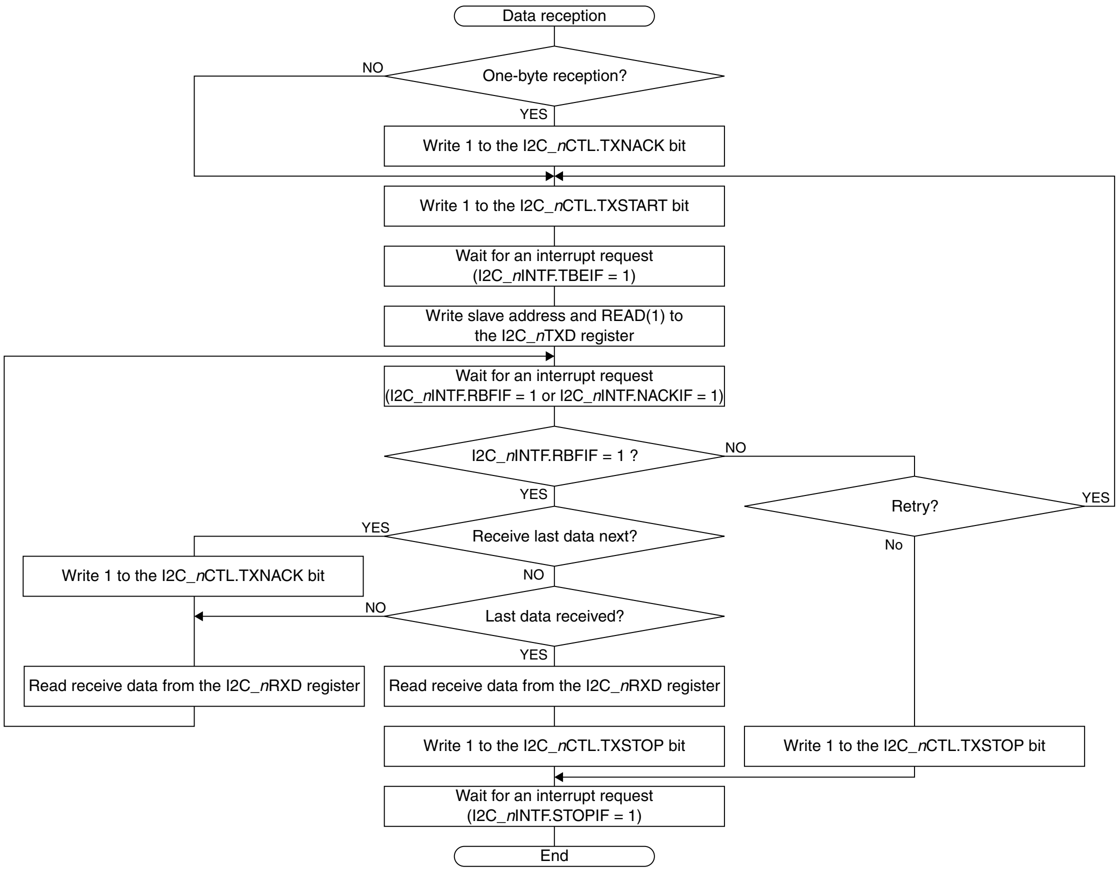


Figure 16.4.3.2 Master Mode Data Reception Flowchart

Data reception using DMA

By setting the I2C_*n*RBFDMAEN.RBFDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and the received data is transferred from the I2C_*n*RXD register to the specified memory via DMA Ch.*x* when the I2C_*n*INTF.RBFIF bit is set to 1 (receive buffer full).

This automates the data receiving procedure Steps 6, 8, and 10 described above.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 16.4.3.1 DMA Data Structure Configuration Example (for Data Reception)

Item		Setting example
End pointer	Transfer source	I2C_*n*RXD register address
	Transfer destination	Memory address to which the last received data is stored
Control data	dst_inc	0x0 (+1)
	dst_size	0x0 (byte)
	src_inc	0x3 (no increment)
	src_size	0x0 (byte)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of receive data
	cycle_ctrl	0x1 (basic transfer)

16.4.4 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.

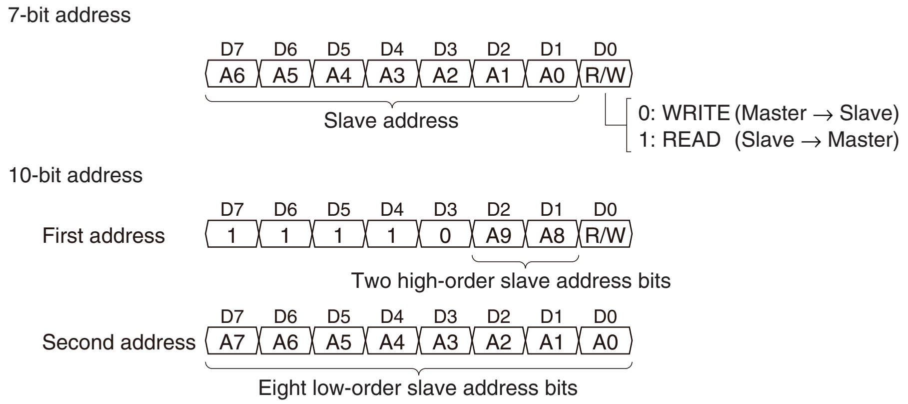


Figure 16.4.4.1 10-bit Address Configuration

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 16.4.4.2 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.

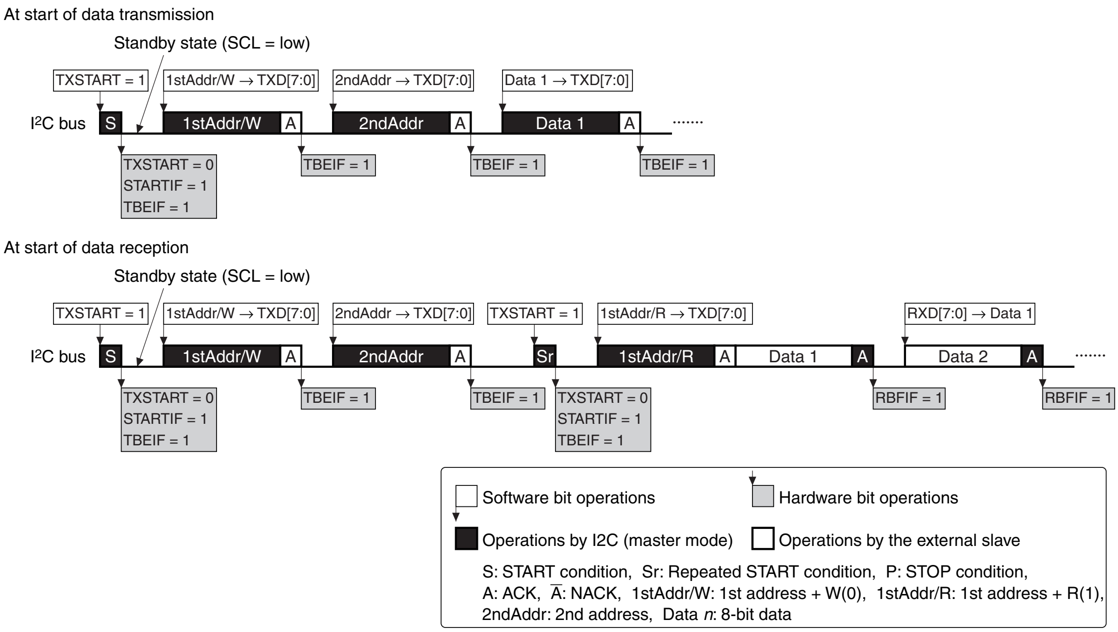


Figure 16.4.4.2 Example of Data Transfer Starting Operations in 10-bit Address Mode (Master Mode)

16.4.5 Data Transmission in Slave Mode

A data sending procedure in slave mode and the I2C Ch.*n* operations are shown below. Figures 16.4.5.1 and 16.4.5.2 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.

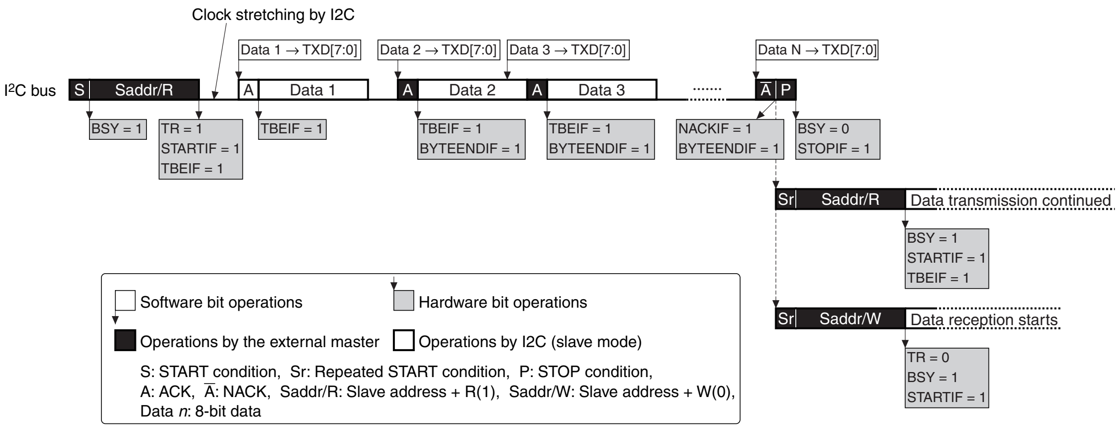


Figure 16.4.5.1 Example of Data Sending Operations in Slave Mode

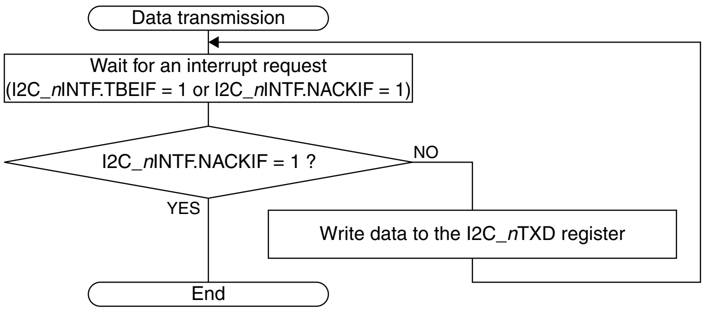


Figure 16.4.5.2 Slave Mode Data Transmission Flowchart

16.4.6 Data Reception in Slave Mode

A data receiving procedure in slave mode and the I2C Ch.*n* operations are shown below. Figures 16.4.6.1 and 16.4.6.2 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.

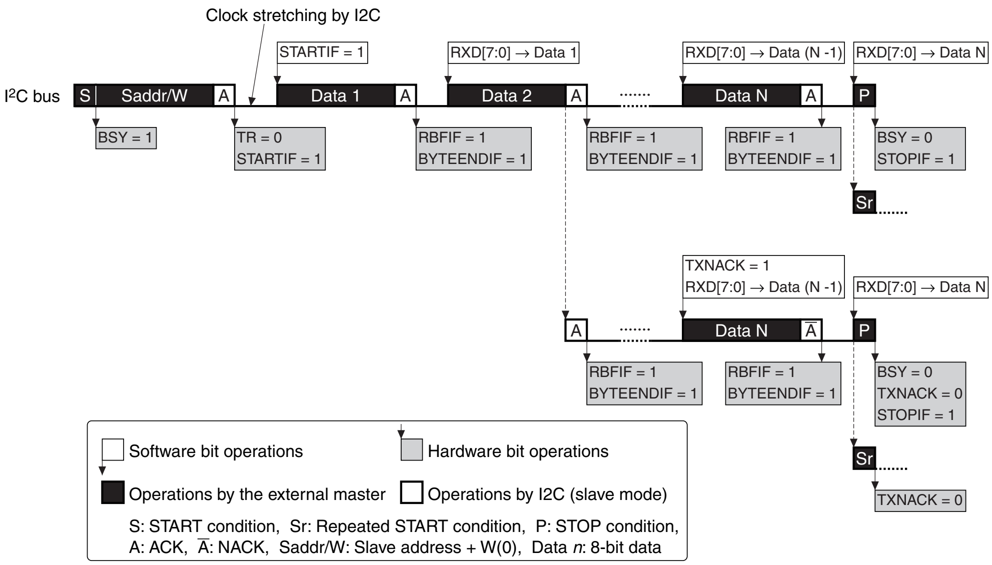


Figure 16.4.6.1 Example of Data Receiving Operations in Slave Mode

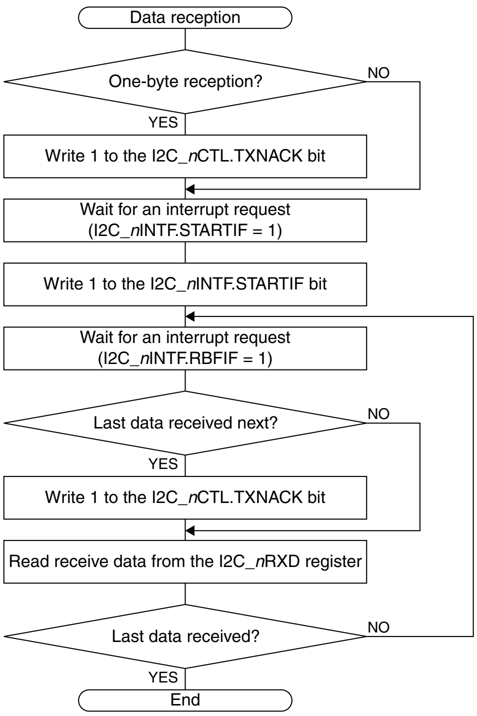


Figure 16.4.6.2 Slave Mode Data Reception Flowchart

16.4.7 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 16.4.7.1 shows an operation example. See Figure 16.4.4.1 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.

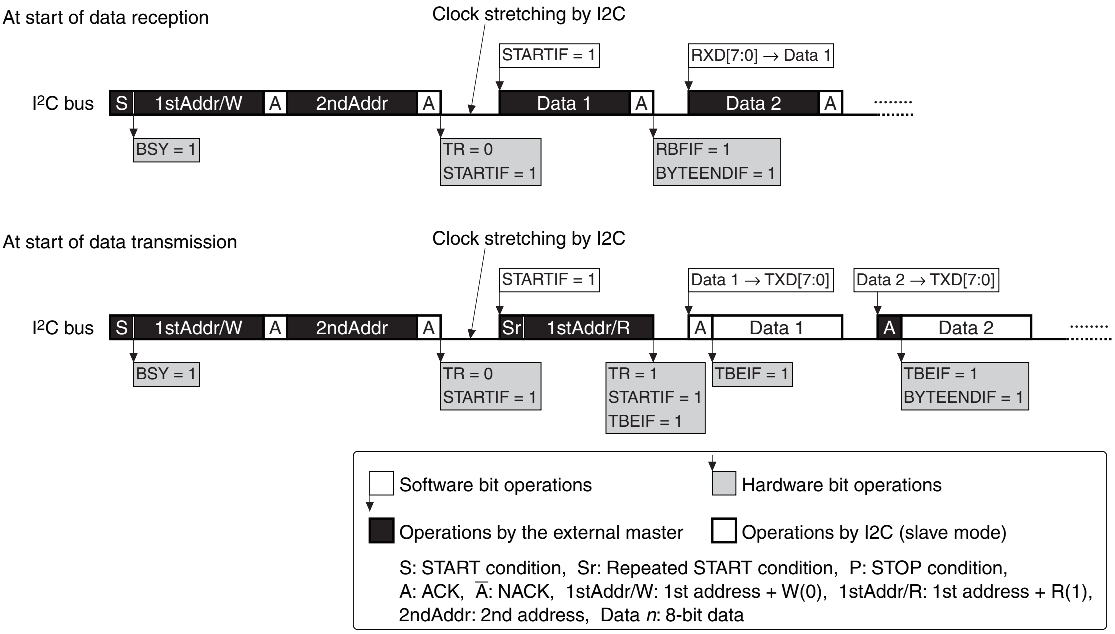


Figure 16.4.7.1 Example of Data Transfer Starting Operations in 10-bit Address Mode (Slave Mode)

16.4.8 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.

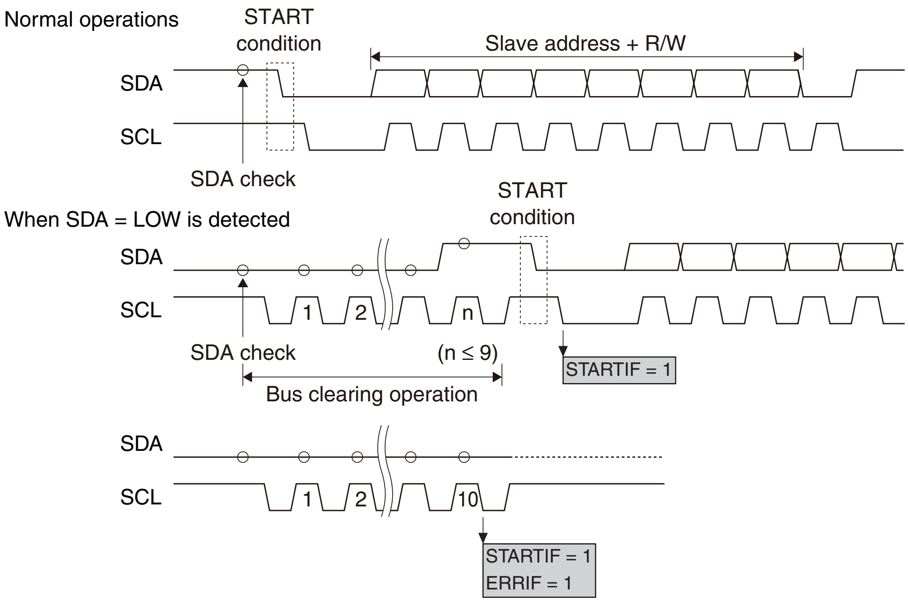


Figure 16.4.8.1 Automatic Bus Clearing Operation

16.4.9 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 16.4.9.1 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

16.5 Interrupts

The I2C has a function to generate the interrupts shown in Table 16.5.1.

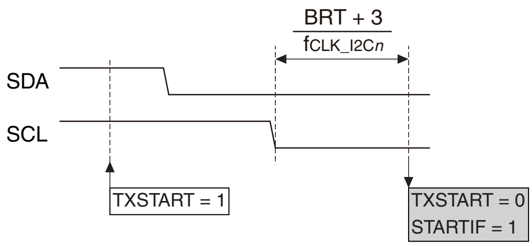
Table 16.5.1 I2C Interrupt Function

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

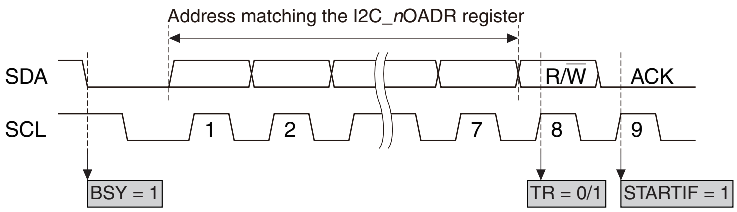
The I2C provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU only when the interrupt flag, of which interrupt has been enabled by the interrupt enable bit, is set. For more information on interrupt control, refer to the “Interrupt” chapter.

(1) START condition interrupt

Master mode

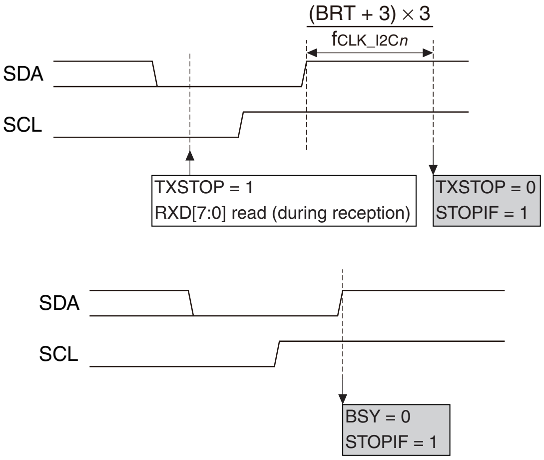


Slave mode



(2) STOP condition interrupt

Master mode



(f_{CLK_I2Cn}: I2C operating clock frequency [Hz], BRT: I2C_*n*BR.BRT[6:0] bits setting value (1 to 127))

Figure 16.5.1 START/STOP Condition Interrupt Timings

16.6 DMA Transfer Requests

The I2C has a function to generate DMA transfer requests from the causes shown in Table 16.6.1.

Table 16.6.1 DMA Transfer Request Causes of I2C

Cause to request DMA transfer	DMA transfer request flag	Set condition	Clear condition
Receive buffer full	Receive buffer full flag (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	Transmit buffer empty flag (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 DMA transfer request enable bits corresponding to each DMA transfer request flag shown above for the number of DMA channels. A DMA transfer request is sent to the pertinent channel of the DMA controller only when the DMA transfer request flag, of which DMA transfer has been enabled by the DMA transfer request enable bit, is set. The DMA transfer request flag also serves as an interrupt flag, therefore, both the DMA transfer request and the interrupt cannot be enabled at the same time. After a DMA transfer has completed, disable the DMA transfer to prevent unintended DMA transfer requests from being issued. For more information on the DMA control, refer to the “DMA Controller” chapter.

16.7 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 during debugging or not.

1 (R/W): Clock supplied during debugging

0 (R/W): No clock supplied during debugging

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 16.7.1 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 while 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

I2C Ch.*n* Transmit Buffer Empty DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C_*n*TBEDMAEN	15–0	TBEDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 TBEDMAEN[15:0]

These bits enable the I2C to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when a transmit buffer empty state has occurred.

1 (R/W): Enable DMA transfer request

0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

I2C Ch.*n* Receive Buffer Full DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
I2C_*n*RBFDMAEN	15–0	RBFDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 RBFDMAEN[15:0]

These bits enable the I2C to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when a receive buffer full state has occurred.

1 (R/W): Enable DMA transfer request

0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

17 16-bit PWM Timers (T16B)

17.1 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 or DMA request 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 or DMA requests. (Can be used to measure external event periods/cycles.)

Figure 17.1.1 shows the T16B configuration.

Table 17.1.1 T16B Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	2 channels (Ch.0 and Ch.1)		
Event counter function	Ch.0: Not available Ch.1: EXCL10 or EXCL11 pin input	Ch.0: EXCL00 or EXCL01 pin input Ch.1: EXCL10 or EXCL11 pin input	
Number of comparator/ capture circuits per channel	4 systems (0 to 3)		
Timer generating signal output	Ch.0: TOUT00 to TOUT03 pin outputs (4 systems) Ch.1: TOUT10 to TOUT13 pin outputs (4 systems)		
Capture signal input	Ch.0: CAP00 to CAP03 pin inputs (4 systems) Ch.1: CAP10 to CAP13 pin inputs (4 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.

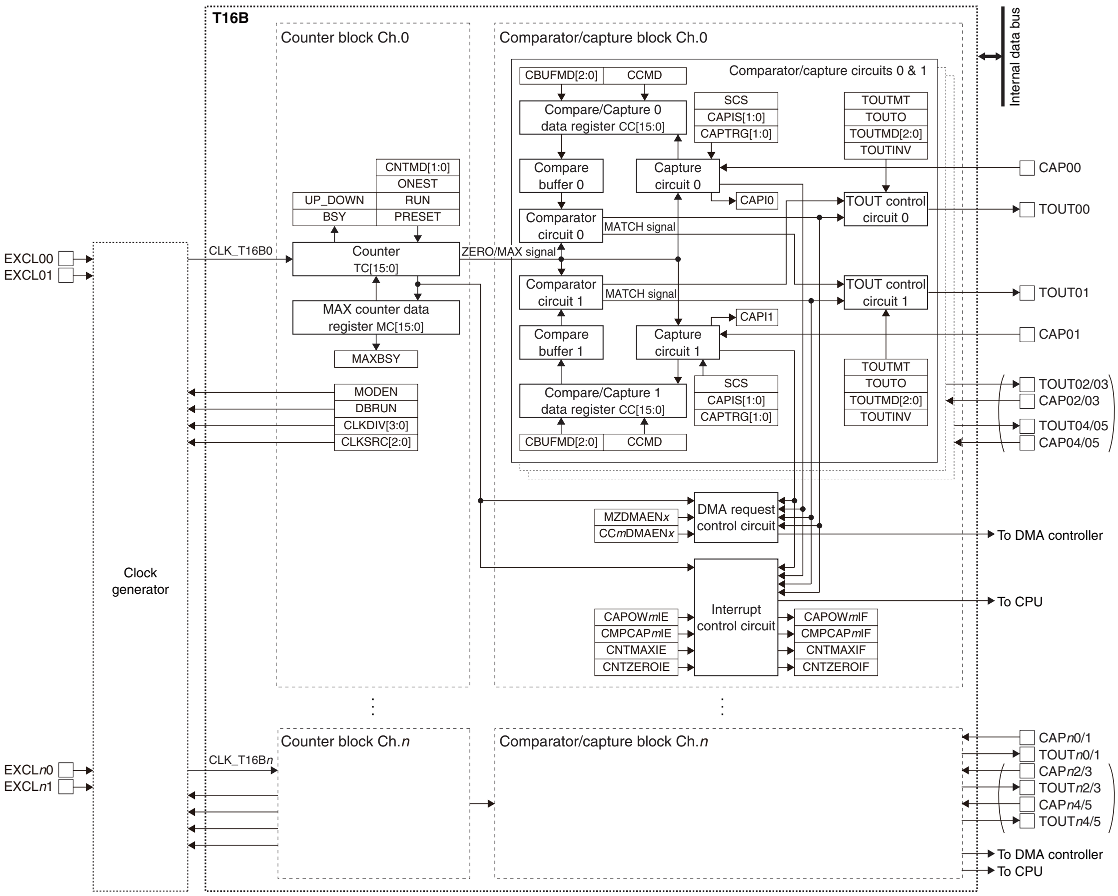


Figure 17.1.1 T16B Configuration

17.2 Input/Output Pins

Table 17.2.1 lists the T16B pins.

Table 17.2.1 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.

17.3 Clock Settings

17.3.1 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)

17.3.2 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.

17.3.3 Clock Supply During Debugging

The CLK_T16B*n* supply during debugging 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 state if the T16B_*n*CLK.DBRUN bit = 0. After the CPU returns to normal operation, 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 state 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 state.

17.3.4 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.

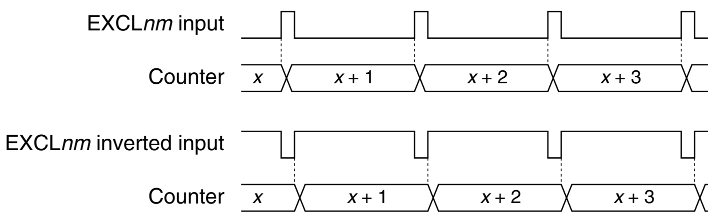


Figure 17.3.4.1 Count Timing (During Count Up Operation)

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.

17.4 Operations

17.4.1 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. Configure the DMA controller and set the following T16B control bits when using DMA transfer:
   - Write 1 to the DMA transfer request enable bits in the T16B_*n*MZDMAEN and T16B_*n*CC*m*DMAEN registers. (Enable DMA transfer requests)
8. 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. Configure the DMA controller and set the following T16B control bits when using DMA transfer:
   - Write 1 to the DMA transfer request enable bits in the T16B_*n*MZDMAEN and T16B_*n*CC*m*DMAEN registers. (Enable DMA transfer requests)
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)

17.4.2 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.

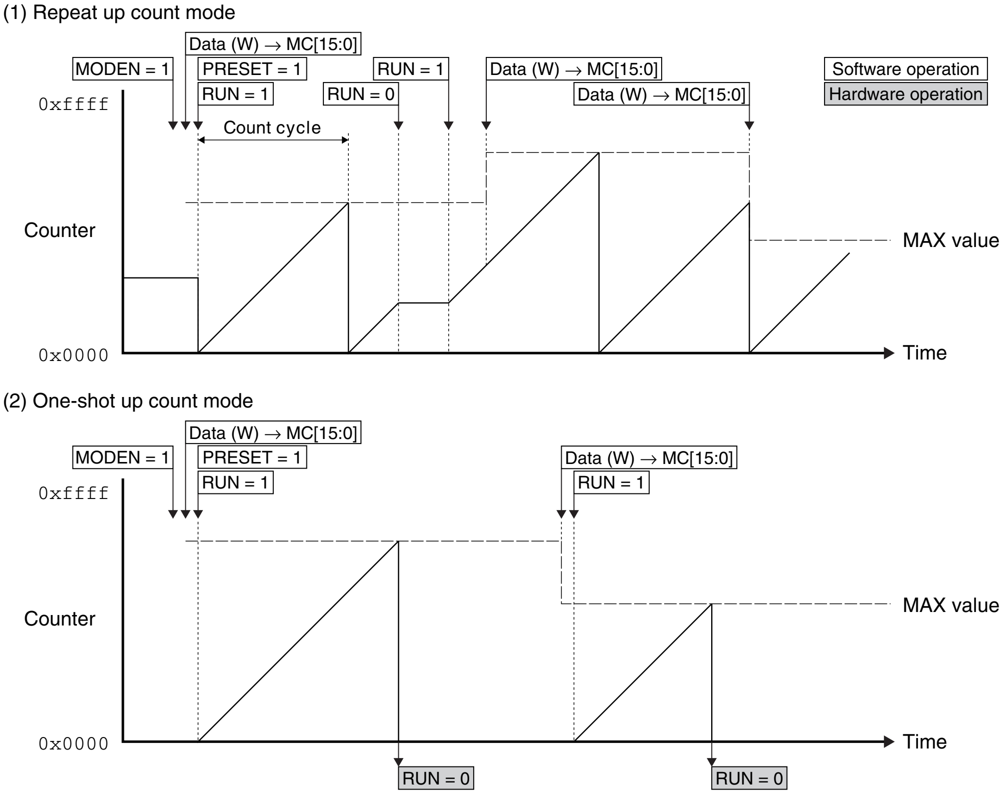


Figure 17.4.2.1 Operations in Repeat Up Count and One-shot Up Count Modes

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.

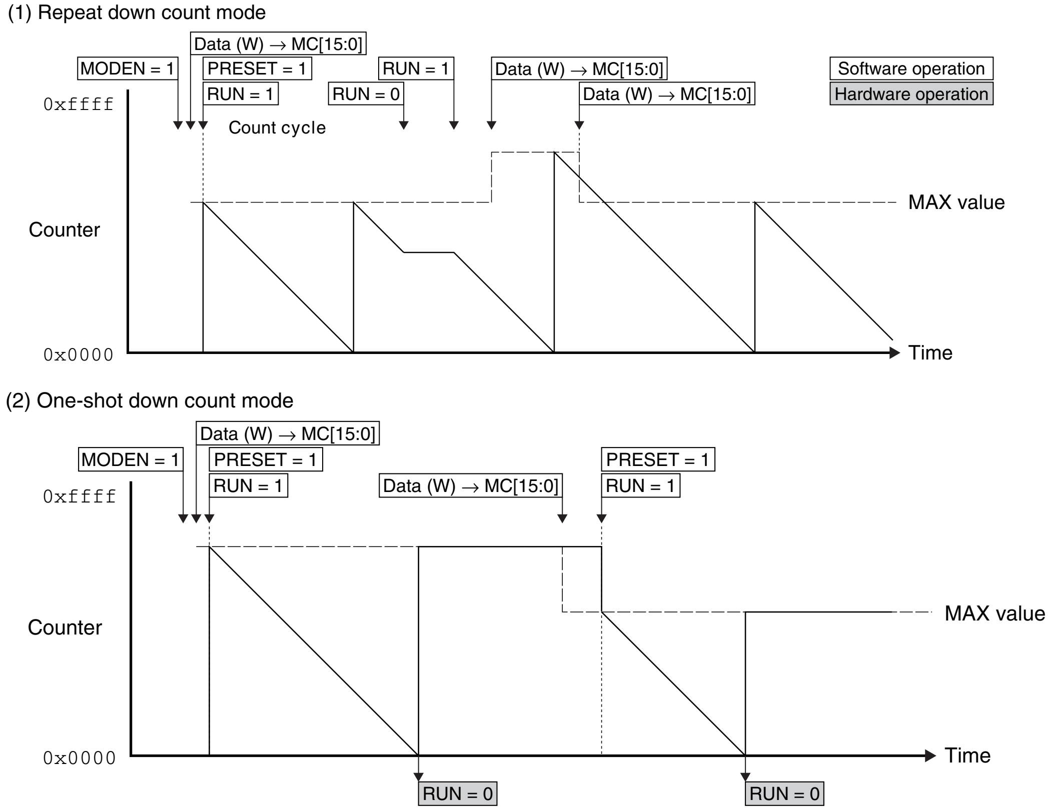


Figure 17.4.2.2 Operations in Repeat Down Count and One-shot Down Count Modes

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.

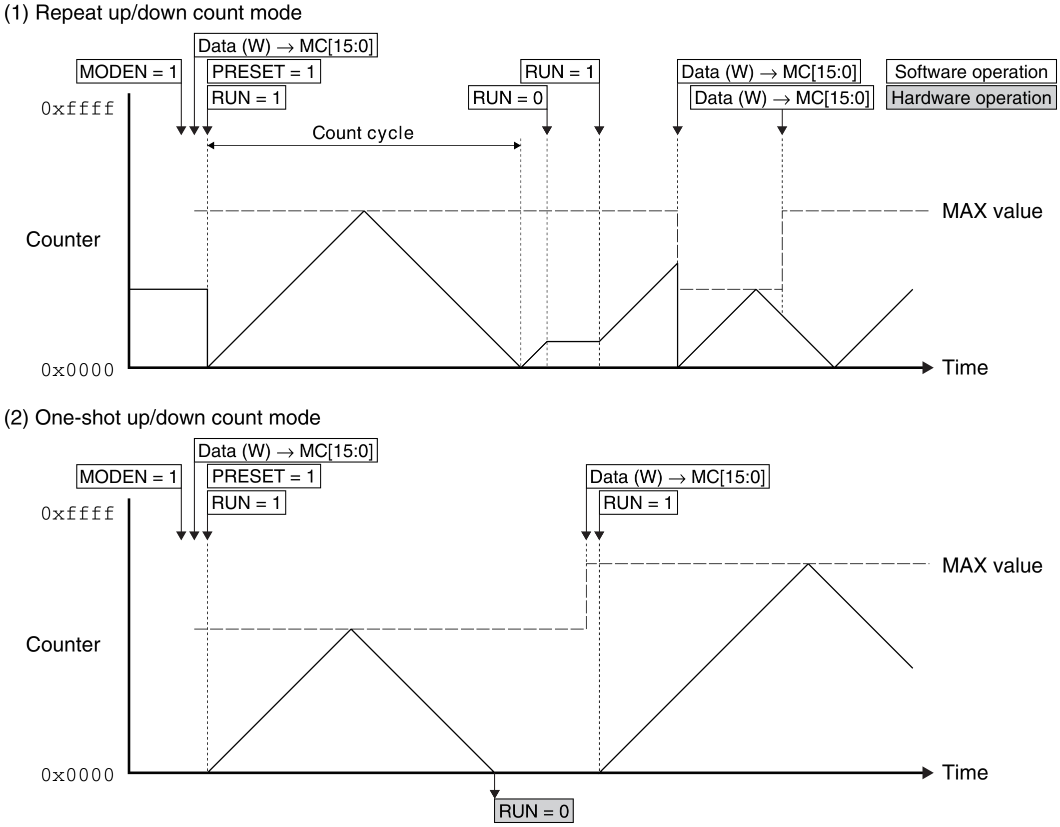


Figure 17.4.2.3 Operations in Repeat Up/Down Count and One-shot Up/Down Count Modes

17.4.3 Comparator/Capture Block Operations

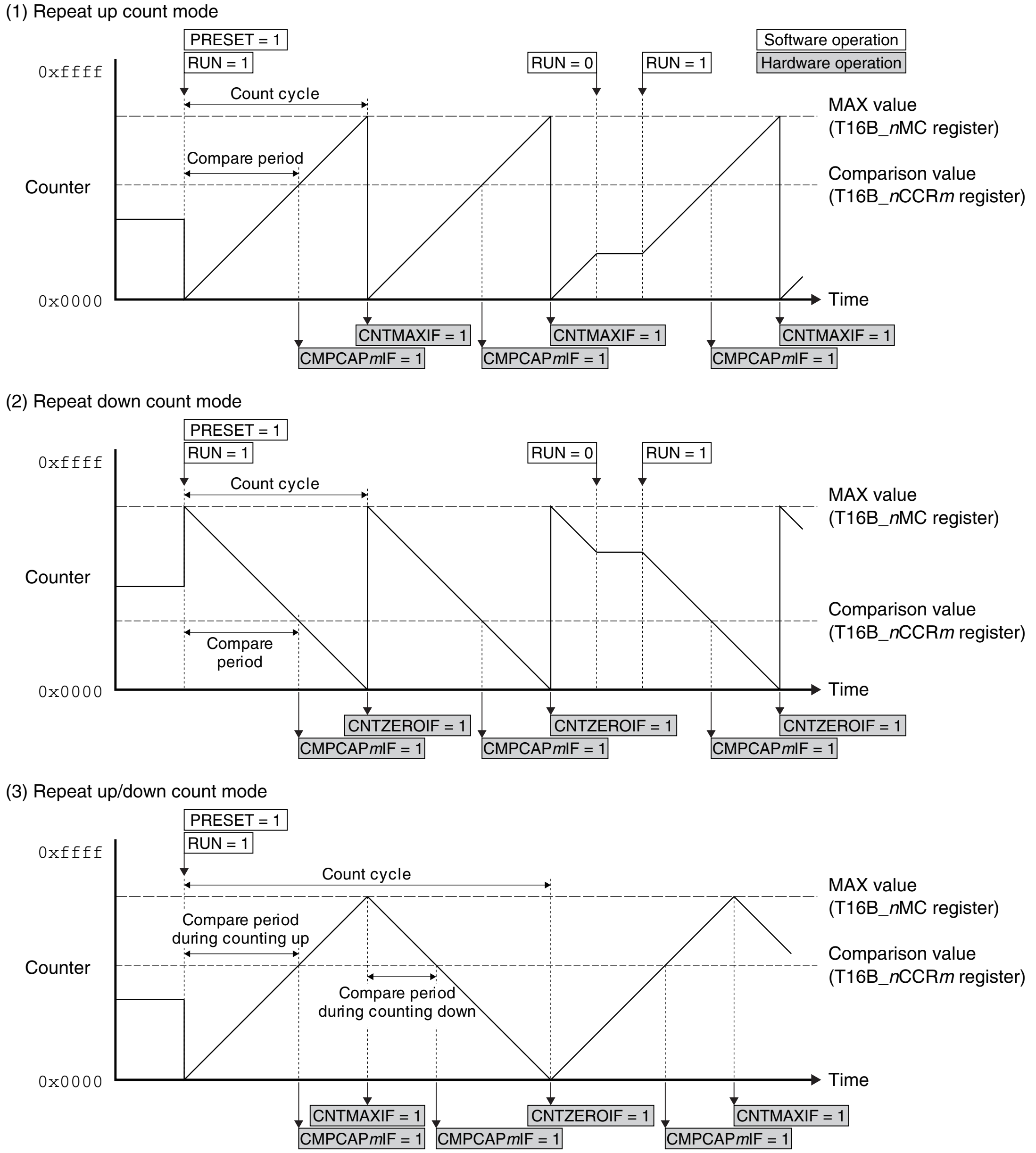
The comparator/capture block functions as a comparator to compare the counter value with the register value set or a capture circuit to capture counter values using the external/software trigger signals.

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.



(Note that the T16B_nINTF.CMPCAPmIF/CNTMAXIF/CNTZEROIF bit clearing operations via software are omitted from the figure.)

Figure 17.4.3.1 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

$$\text{Compare period} = \frac{(CC + 1)}{f_{CLK\_T16B}}\ [s] \qquad \text{Count cycle} = \frac{(MAX + 1)}{f_{CLK\_T16B}}\ [s] \qquad \text{(Eq. 17.1)}$$

During counting down

$$\text{Compare period} = \frac{(MAX - CC + 1)}{f_{CLK\_T16B}}\ [s] \qquad \text{Count cycle} = \frac{(MAX + 1)}{f_{CLK\_T16B}}\ [s] \qquad \text{(Eq. 17.2)}$$

Where

CC: T16B_nCCRm register setting value (0 to 65,535)

MAX: T16B_nMC register setting value (0 to 65,535)

f_{CLK_T16B}: Count clock frequency [Hz]

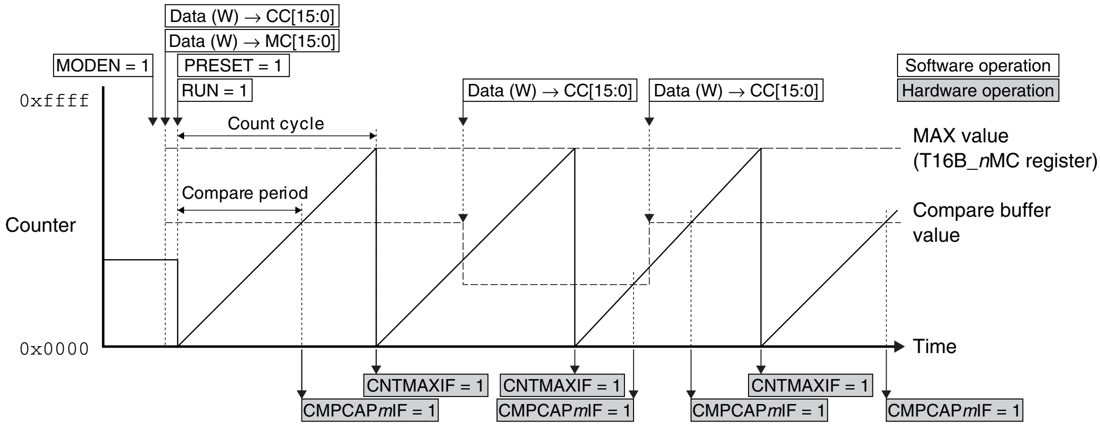
The comparator MATCH signal and counter MAX/ZERO signals are also used to generate a timer output waveform (TOUT). Refer to “TOUT Output Control” for more information.

Compare buffer

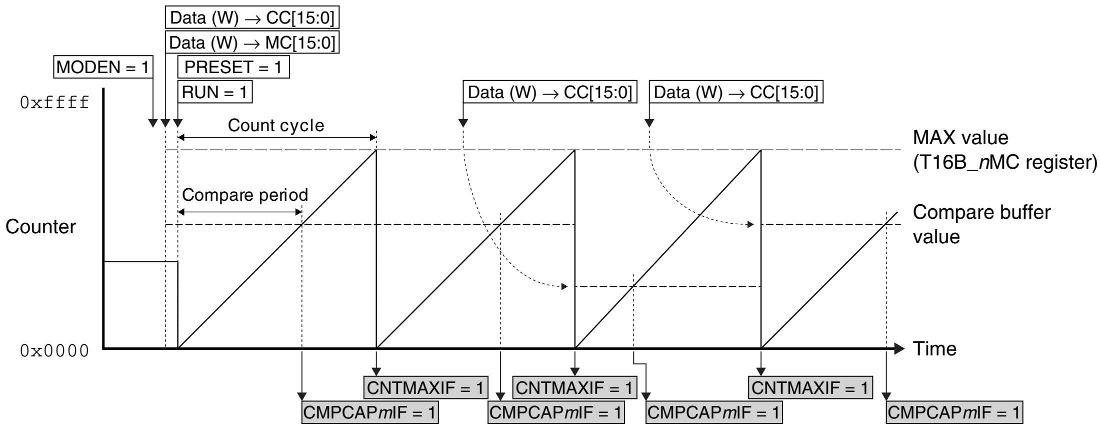
The comparator loads the comparison value, which has been written to the T16B_*n*CCR*m* register, to the compare buffer before comparing it with the counter value. For example, when generating a PWM waveform, the waveform with the desired duty ratio may not be generated if the comparison value is altered 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

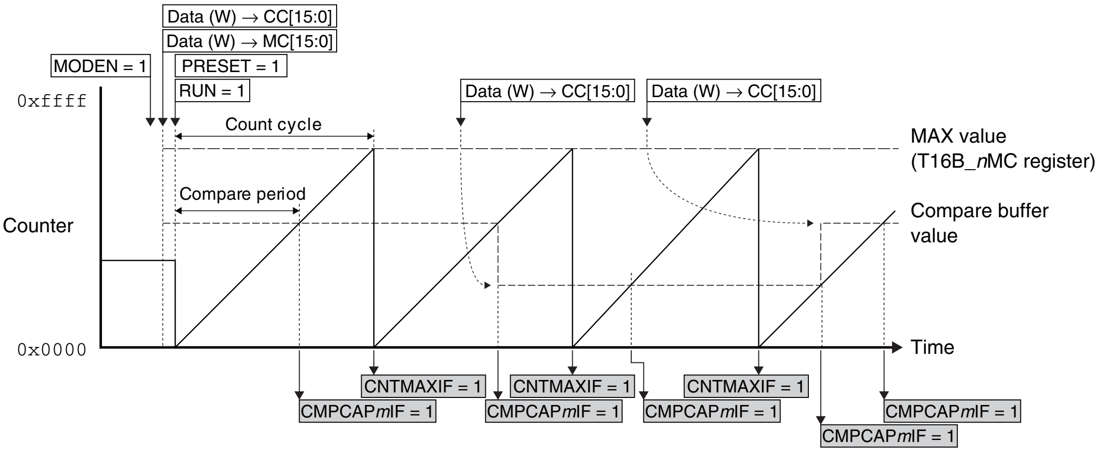
(1.1) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x0



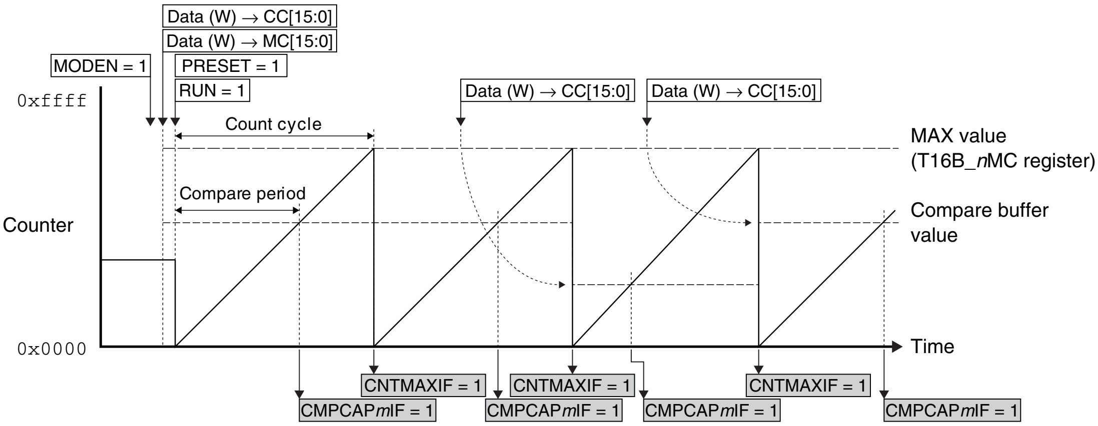
(1.2) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x1



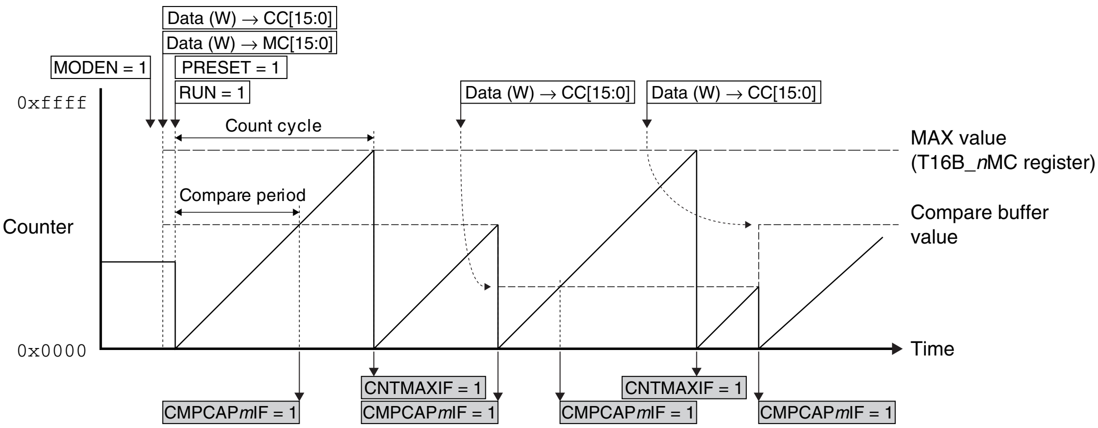
(1.3) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x2



(1.4) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x3

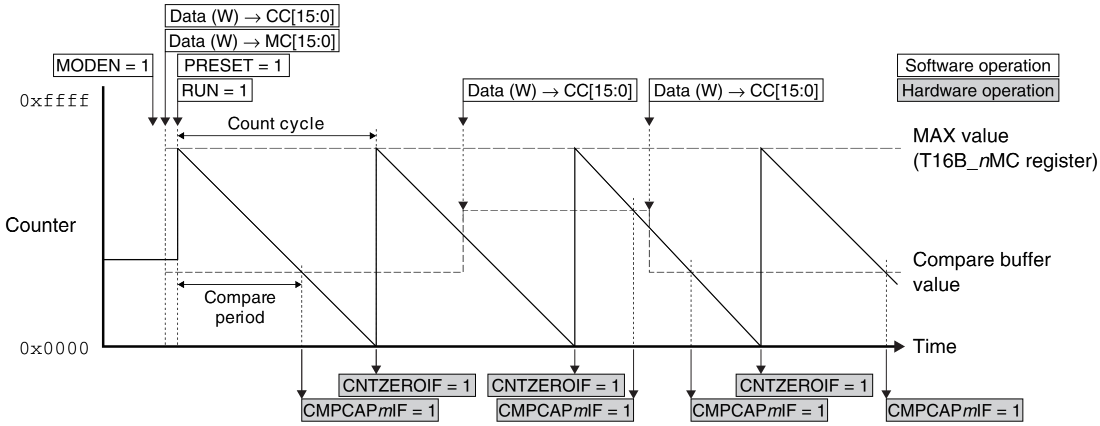


(1.5) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x4

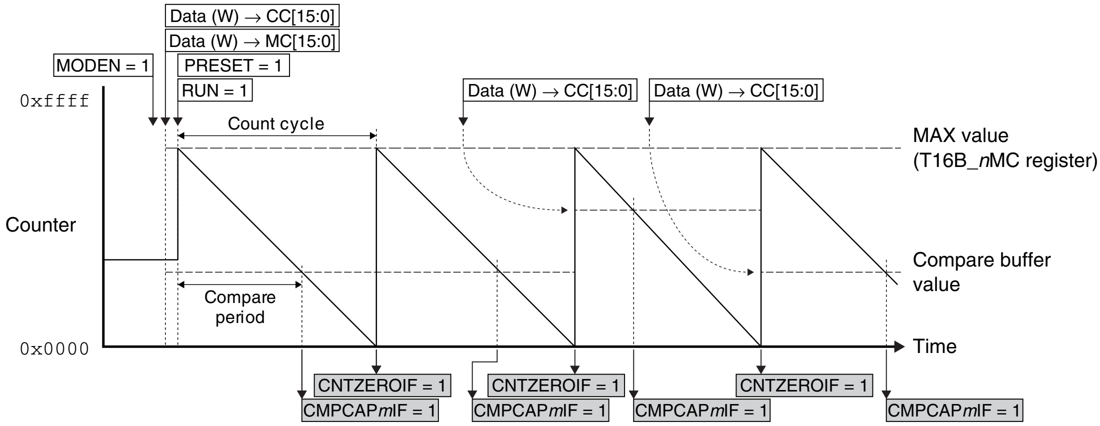


(2) Repeat down count mode

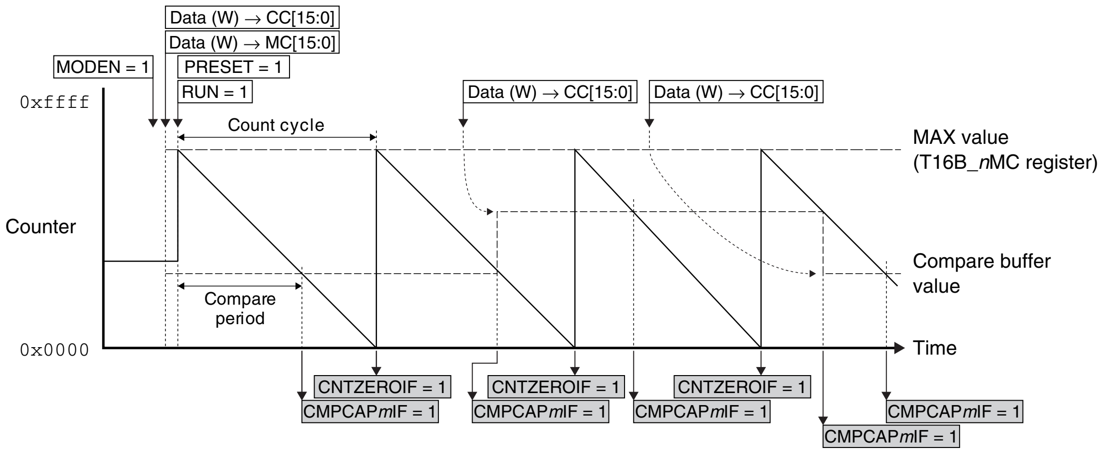
(2.1) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x0



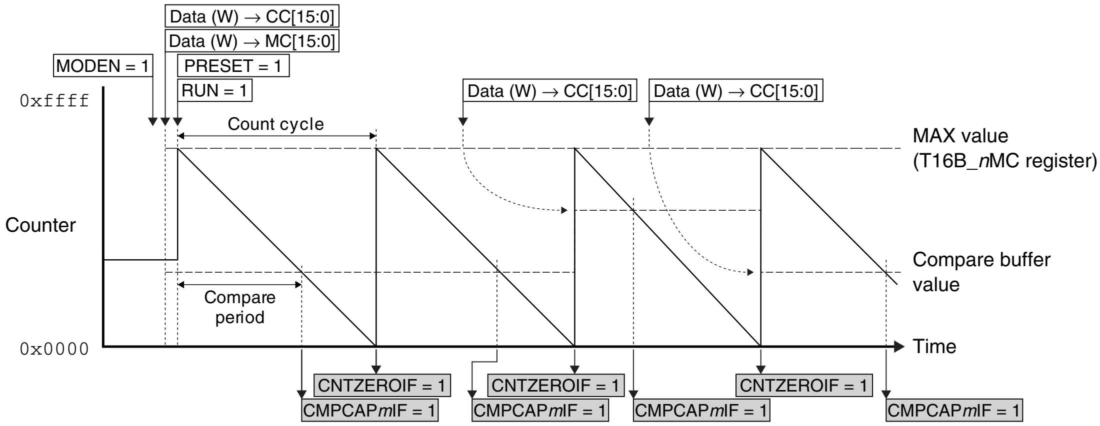
(2.2) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x1



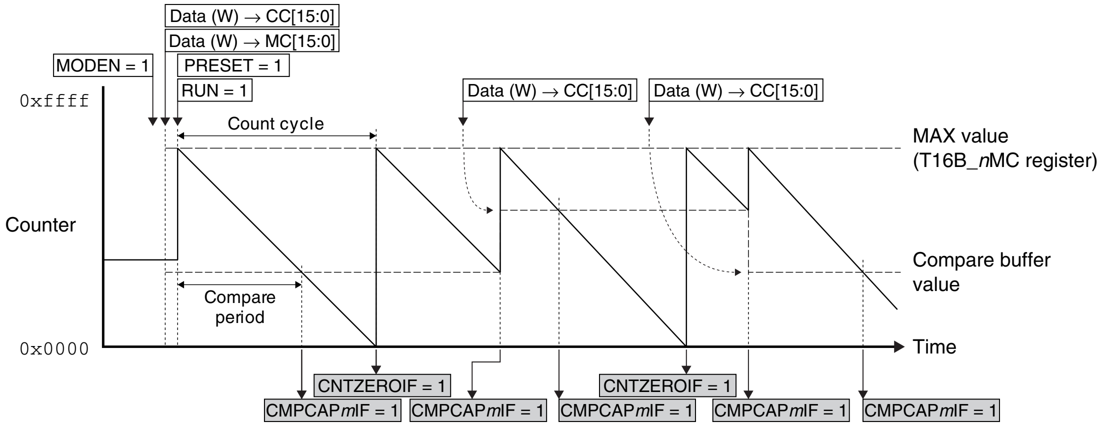
(2.3) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x2



(2.4) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x3

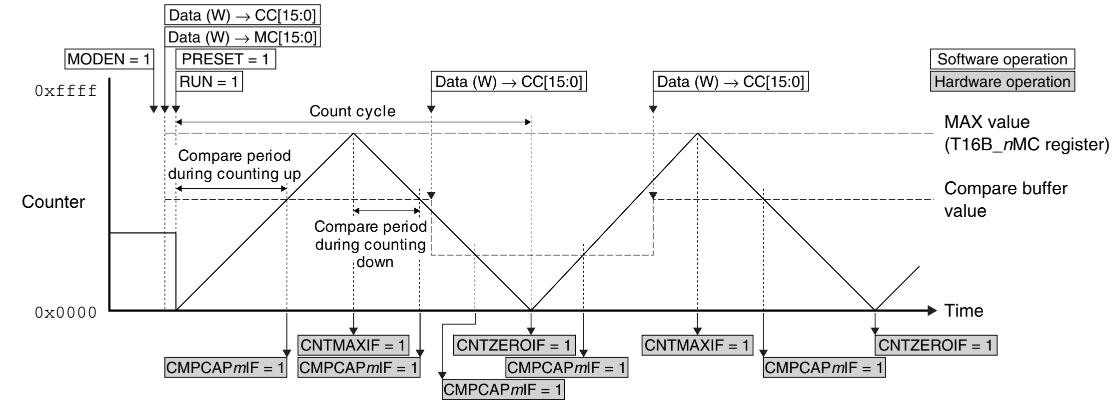


(2.5) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x4

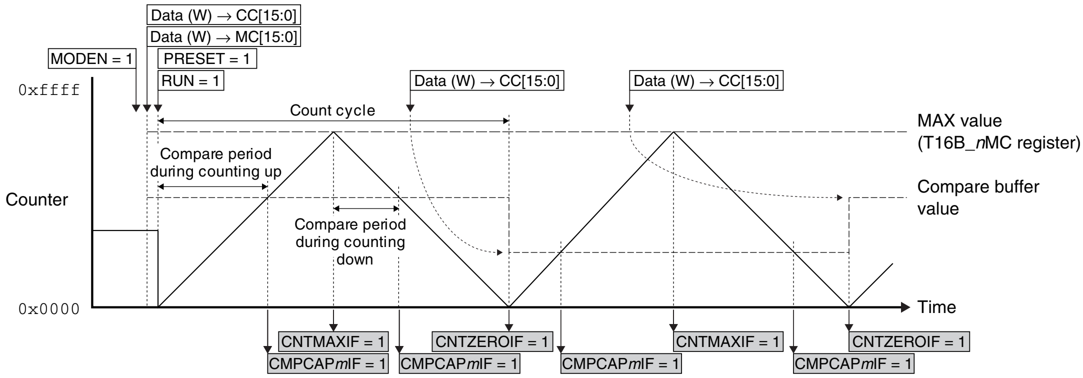


(3) Repeat up/down count mode

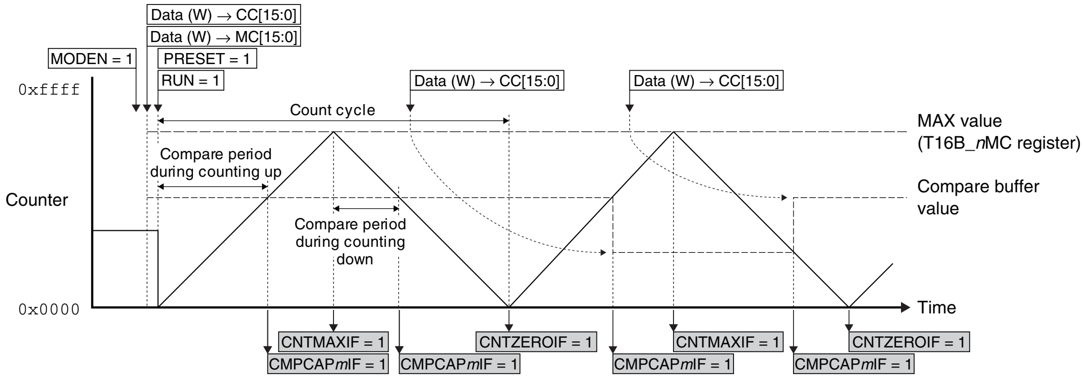
(3.1) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x0



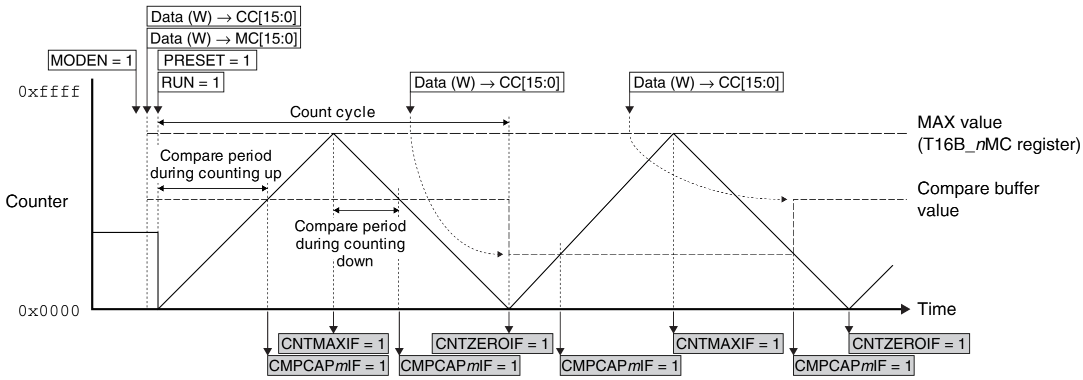
(3.2) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x1



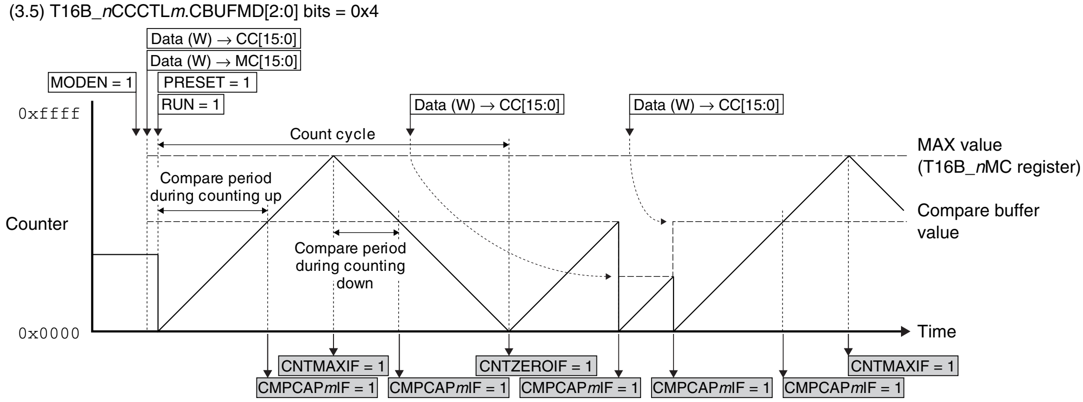
(3.3) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x2



(3.4) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x3



(3.5) T16B_*n*CCCTL*m*.CBUFMD[2:0] bits = 0x4



(Note that the T16B_*n*INTF.CMPCAP*m*IF/CNTMAXIF/CNTZEROIF bit clearing operations via software are omitted from the figure.)

Figure 17.4.3.2 Compare Buffer Operations

Compare period and count cycle settings using DMA

By setting the T16B_*n*CC*m*DMAEN.CC*m*DMAEN*x* bit to 1 (DMA transfer request enabled) in comparator mode, a DMA transfer request is sent to the DMA controller and compare data is transferred from the specified memory to the T16B_*n*CCR*m* register via DMA Ch.*x* when the T16B_*n*INTF.CMPCAP*m*IF bit is set to 1 (when the counter reaches the compare buffer value).

Similarly, by setting the T16B_*n*CC*m*DMAEN.MZDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and a counter MAX value is transferred from the specified memory to the T16B_*n*MC register via DMA Ch.*x* when the T16B_*n*INTF.CNTMAXIF bit is set to 1 (when the counter reaches the MAX value) in up or up/down count mode, or when the T16B_*n*INTF.CNTZEROIF bit is set to 1 (when the counter reaches zero) in down count mode.

This automates the compare period and count cycle settings of the timer counter.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance so that the setting data will be transferred to the T16B_*n*CCR*m* or T16B_*n*MC register. For more information on DMA, refer to the “DMA Controller” chapter.

Table 17.4.3.1 DMA Data Structure Configuration Example (T16B Compare Period and Count Cycle Settings)

Item		Setting example
End pointer	Transfer source	Memory address in which the last setting data is stored
	Transfer destination	T16B_*n*CCR*m* or T16B_*n*MC register address
Control data	dst_inc	0x3 (no increment)
	dst_size	0x1 (haflword)
	src_inc	0x1 (+2)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

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).

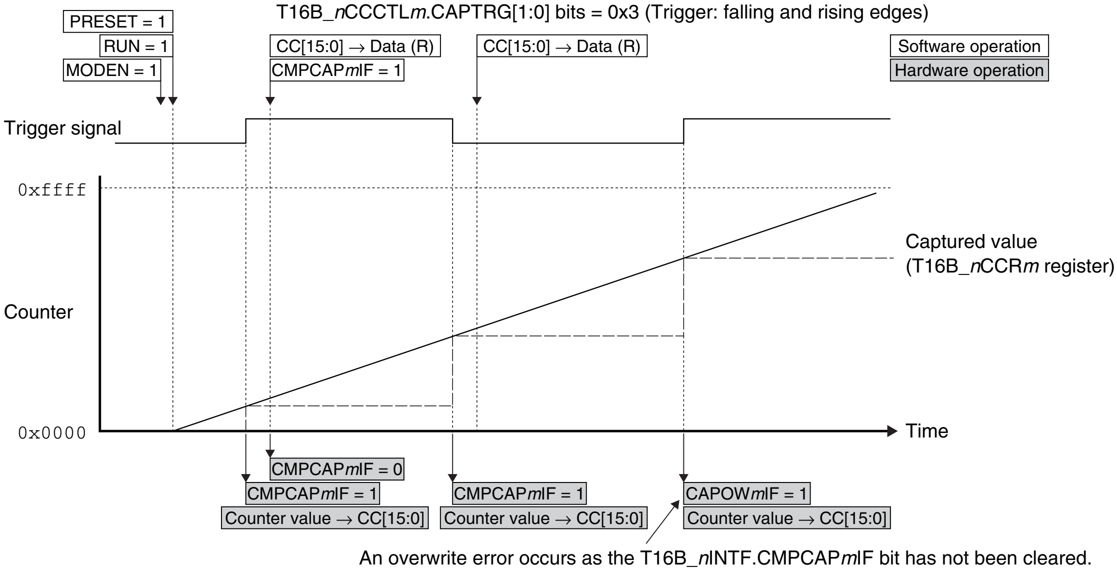


Figure 17.4.3.3 Operations in Capture Mode (Example in One-shot Up Count Mode)

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.

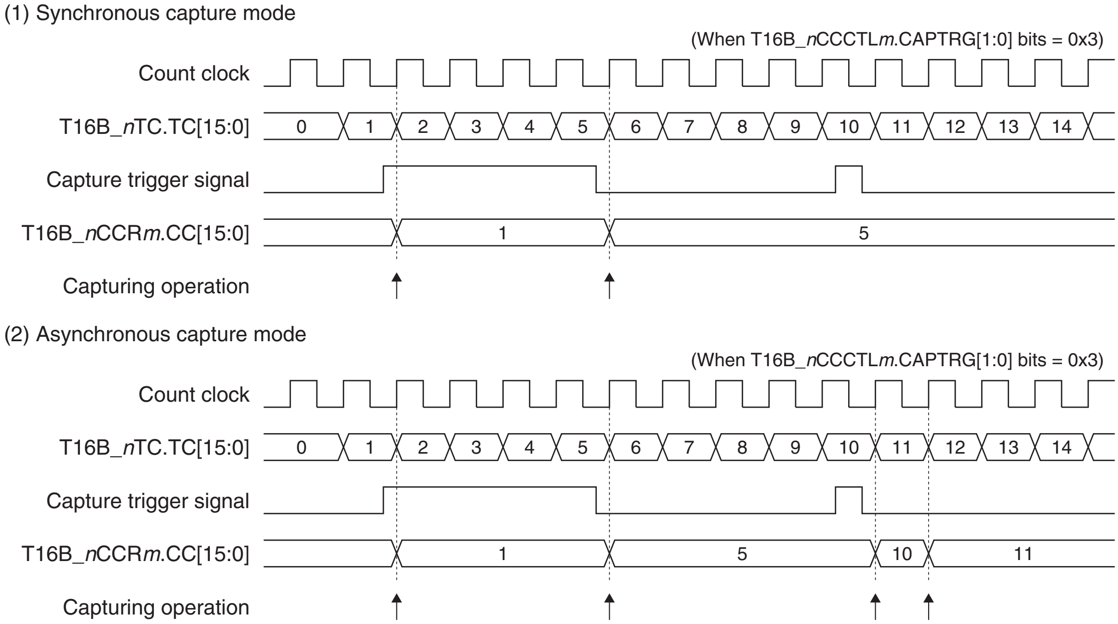


Figure 17.4.3.4 Synchronous Capture Mode/Asynchronous Capture Mode

Capture data transfer using DMA

By setting the T16B_*n*CC*m*DMAEN.CC*m*DMAEN*x* bit to 1 (DMA transfer request enabled) in capture mode, a DMA transfer request is sent to the DMA controller and the T16B_*n*CCR*m* register value is transferred to the specified memory via DMA Ch.*x* when the T16B_*n*INTF.CMPCAP*m*IF bit is set to 1 (when data has been captured).

This automates reading and saving of capture data.

The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 17.4.3.2 DMA Data Structure Configuration Example (Capture Data Transfer)

Item		Setting example
End pointer	Transfer source	T16B_*n*CCR*m* register address
	Transfer destination	Memory address to which the last capture data is stored
Control data	dst_inc	0x1 (+2)
	dst_size	0x1 (haflword)
	src_inc	0x3 (no increment)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

17.4.4 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 17.4.4.1 shows the TOUT output circuits (circuits 0 and 1).

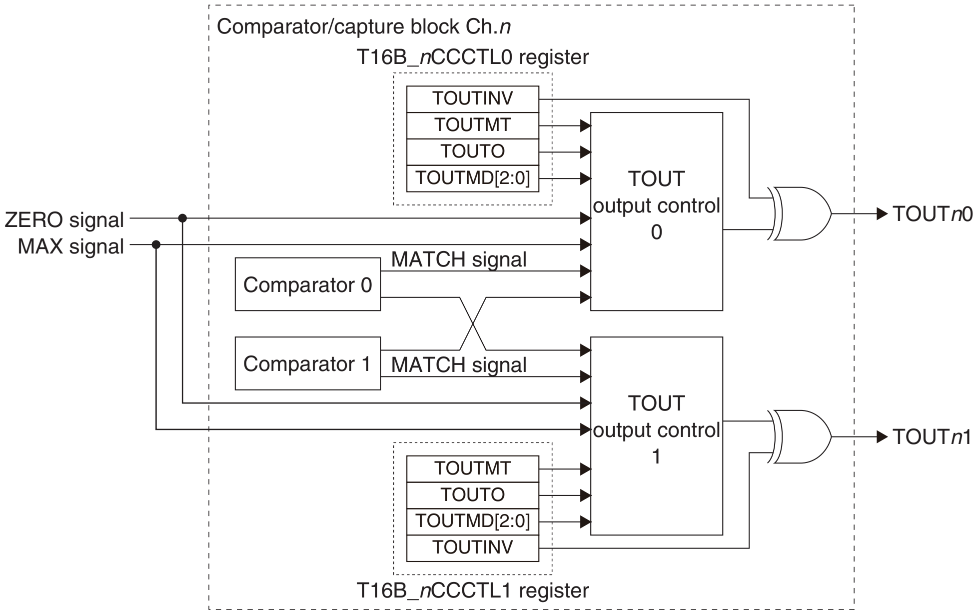


Figure 17.4.4.1 TOUT Output Circuits (Circuits 0 and 1)

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.

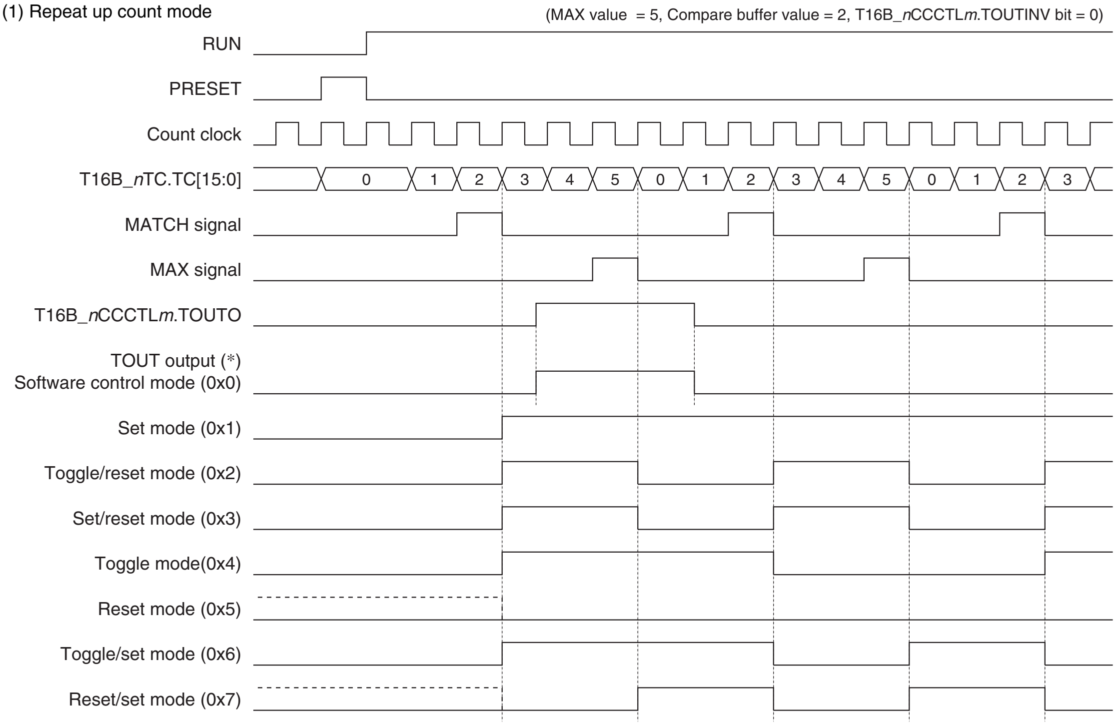
Furthermore, when the T16B_*n*CCCTL*m*.TOUTMT bit is set to 1, the TOUT circuit uses the MATCH signal output from another system in the circuit pair (0 and 1, 2 and 3, 4 and 5). This makes it possible to change the signal twice within a counter cycle.

TOUT signal polarity

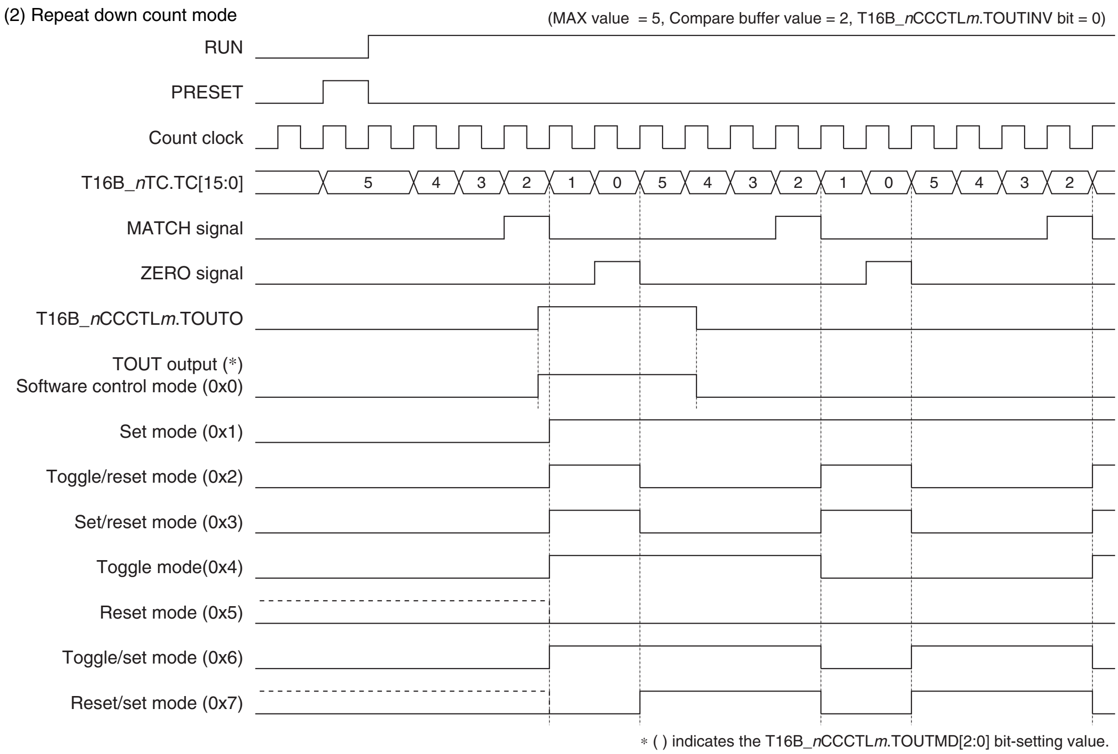
The TOUT signal polarity (active level) can be set using the T16B_*n*CCCTL*m*.TOUTINV bit. It is set to active high by setting the T16B_*n*CCCTL*m*.TOUTINV bit to 0 and active low by setting to 1.

Figures 17.4.4.2 and 17.4.4.3 show the TOUT output waveforms.

(1) Repeat up count mode



* ( ) indicates the T16B_*n*CCCTL*m*.TOUTMD[2:0] bit-setting value.



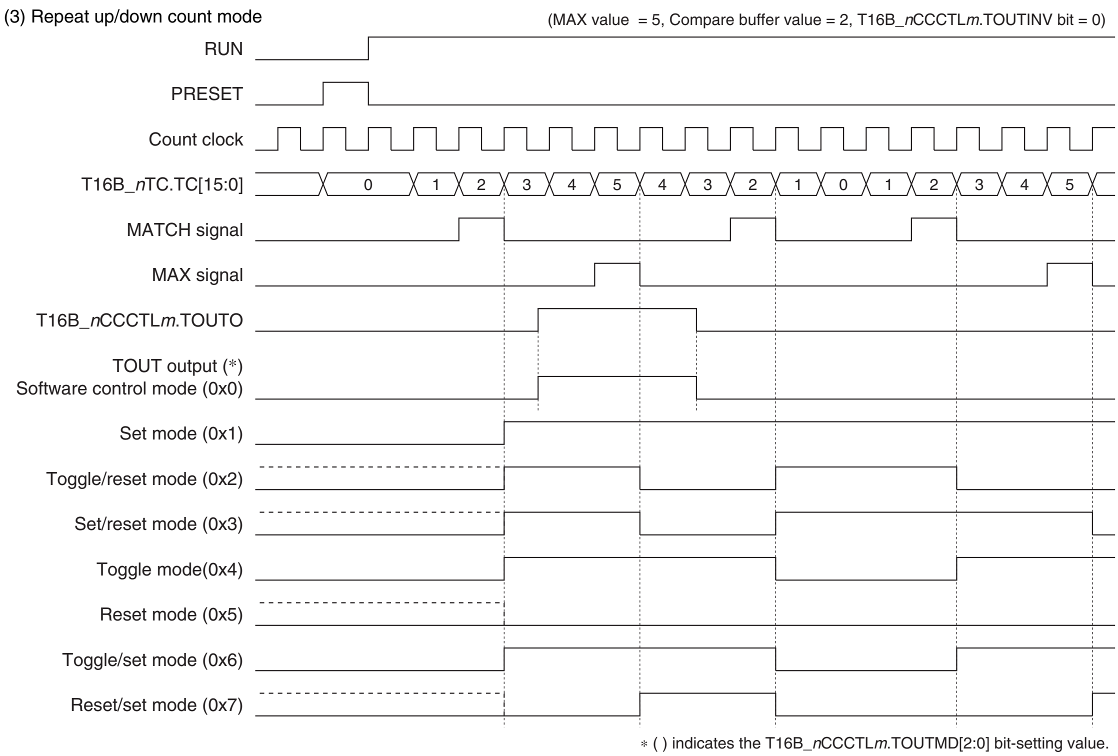
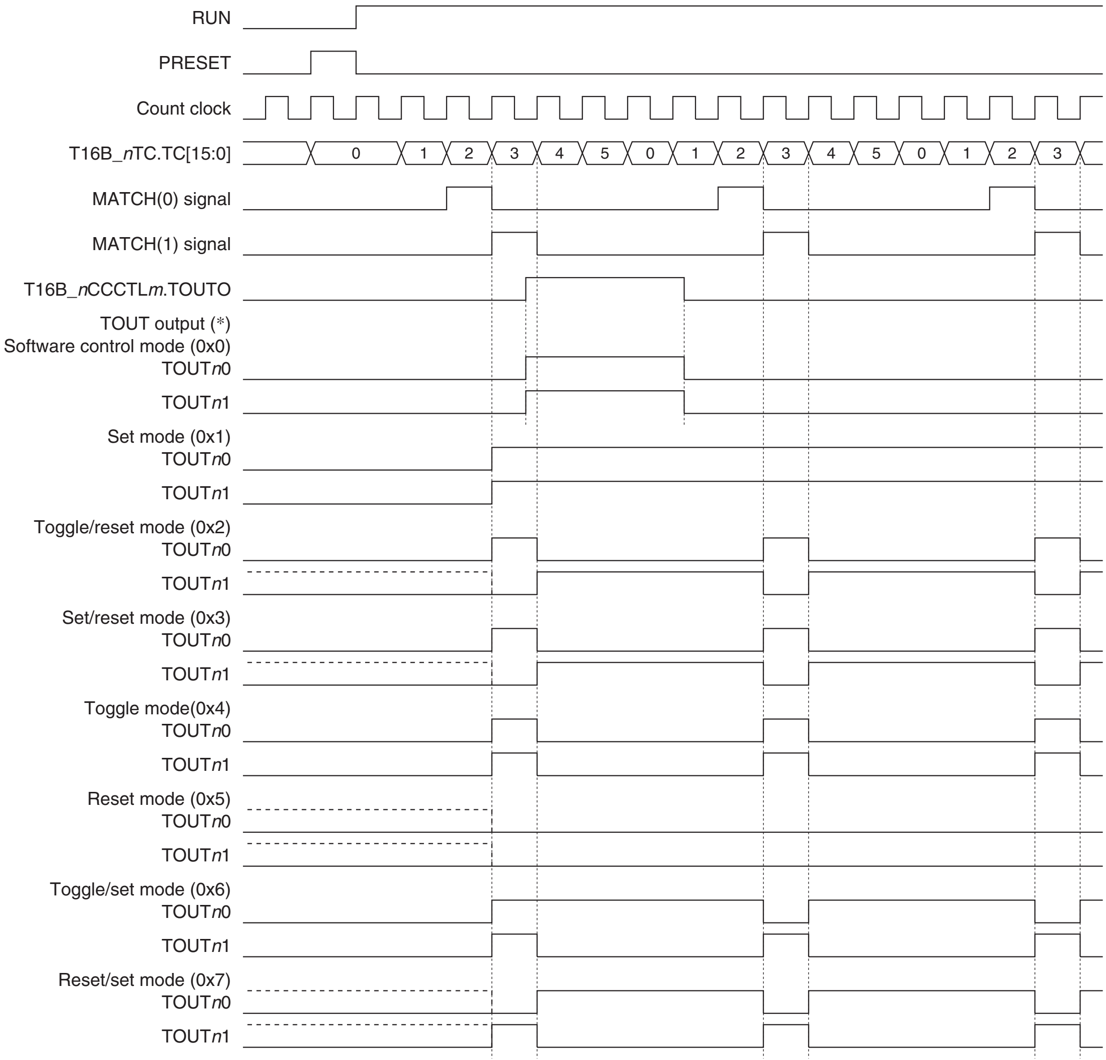


Figure 17.4.4.2 TOUT Output Waveform (T16B_*n*CCCTL*m*.TOUTMT bit = 0)

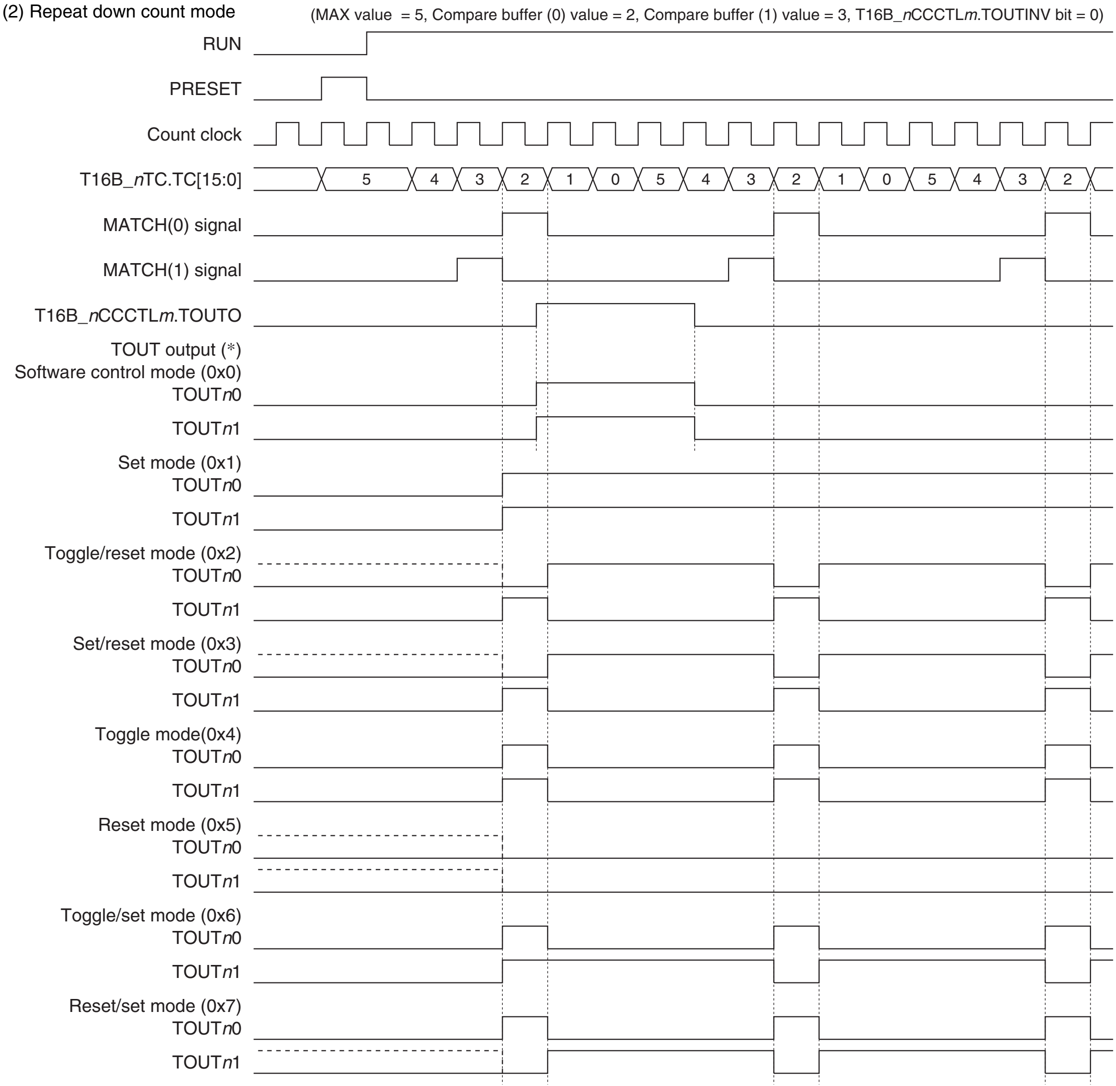
(1) Repeat up count mode

(MAX value = 5, Compare buffer (0) value = 2, Compare buffer (1) value = 3, T16B_nCCCTLm.TOUTINV bit = 0)



* ( ) indicates the T16B_nCCCTLm.TOUTMD[2:0] bit-setting value.

(2) Repeat down count mode



* ( ) indicates the T16B_*n*CCCTL*m*.TOUTMD[2:0] bit-setting value.

(3) Repeat up/down count mode

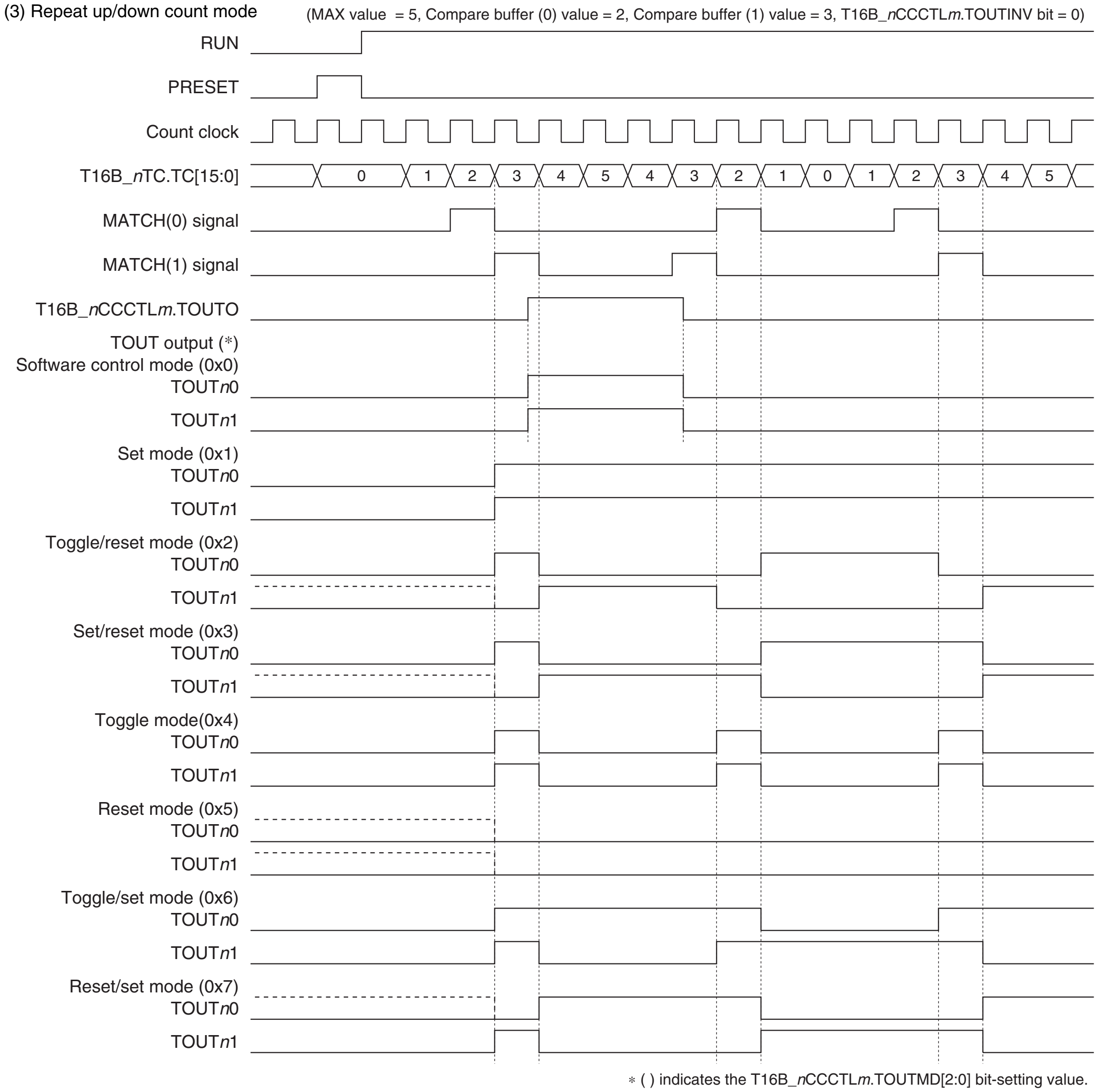


Figure 17.4.4.3 TOUT Output Waveform (T16B_nCCCTL0.TOUTMT bit = 1, T16B_nCCCTL1.TOUTMT bit = 0)

17.5 Interrupt

Each T16B channel has a function to generate the interrupt shown in Table 17.5.1.

Table 17.5.1 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 CPU 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" chapter.

17.6 DMA Transfer Requests

The T16B has a function to generate DMA transfer requests from the causes shown in Table 17.6.1.

Table 17.6.1 DMA Transfer Request Causes of T16B

Cause to request DMA transfer	DMA transfer request flag	Set condition	Clear condition
Compare/capture	Compare/capture flag (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	When the DMA transfer request is accepted
Counter MAX/zero	Counter MAX flag (T16B_*n*INTF.CNTMAXIF) Counter zero flag (T16B_*n*INTF.CNTZEROIF)	When the counter reaches the MAX value in up or up/down count mode When the counter reaches 0x0000 in down count mode	When the DMA transfer request is accepted

The T16B provides DMA transfer request enable bits corresponding to each DMA transfer request flag shown above for the number of DMA channels. A DMA transfer request is sent to the pertinent channel of the DMA controller only when the DMA transfer request flag, of which DMA transfer has been enabled by the DMA transfer request enable bit, is set. The DMA transfer request flag also serves as an interrupt flag, therefore, both the DMA transfer request and the interrupt cannot be enabled at the same time. After a DMA transfer has completed, disable the DMA transfer to prevent unintended DMA transfer requests from being issued. For more information on the DMA control, refer to the "DMA Controller" chapter.

17.7 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 during debugging or not.

1 (R/W): Clock supplied during debugging

0 (R/W): No clock supplied during debugging

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 17.7.1 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 17.7.2).

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 17.7.2).

Table 17.7.2 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 17.7.3 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_T16B*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 17.7.4). 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 17.7.4). The T16B_*n*CCCTL*m*.CAPTRG[1:0] bits are control bits for capture mode and are ineffective in comparator mode.

Table 17.7.4 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 17.7.5 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 (T16B_*n*CCR*m* register = capture register)
0 (R/W): Comparator mode (T16B_*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.

T16B Ch.*n* Counter Max/Zero DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B_*n*MZDMAEN	15–0	MZDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 MZDMAEN[15:0]

These bits enable T16B to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when the counter value reaches the MAX value or 0x0000.

1 (R/W): Enable DMA transfer request

0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

T16B Ch.*n* Compare/Capture *m* DMA Request Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
T16B_*n*CC*m*DMAEN	15–0	CC*m*DMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 CC*m*DMAEN[15:0]

These bits enable T16B to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when the counter value reaches the compare data or is captured.

1 (R/W): Enable DMA transfer request

0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

18 IR Remote Controller (REMC3)

18.1 Overview

The REMC3 circuit generates infrared remote control output signals. This circuit can also be applicable to an EL lamp drive circuit by adding a simple external circuit.
The features of the REMC3 are listed below.

- Outputs an infrared remote control signal.
- Includes a carrier generator.
- Flexible carrier signal generation and data pulse width modulation.
- Automatic data setting function for continuous data transmission.
- Output signal inverting function supporting various formats.
- EL lamp drive waveform can be generated for an application example.

Figure 18.1.1 shows the REMC3 configuration.

Table 18.1.1 REMC3 Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	–	1 transmitter channel	

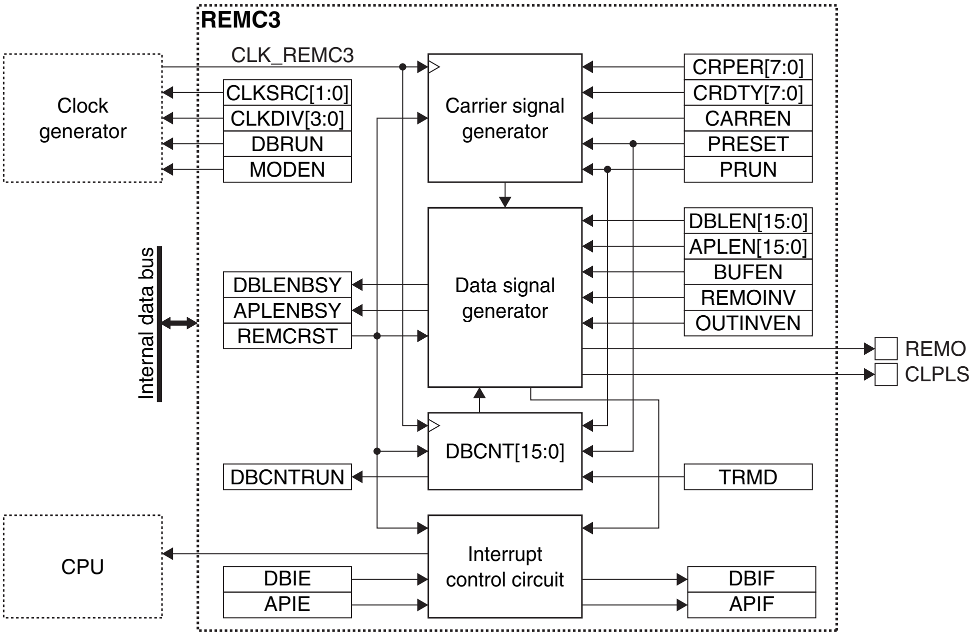


Figure 18.1.1 REMC3 Configuration

18.2 Output Pins and External Connections

18.2.1 List of Output Pins

Table 18.2.1.1 shows the REMC3 pin.

Table 18.2.1.1 REMC3 Pin

Pin name	I/O*	Initial status*	Function
REMO	O	O (L)	IR remote controller transmit data output
CLPLS	O	O (L)	IR remote controller clear pulse output

* Indicates the status when the pin is configured for the REMC3.

If the port is shared with the REMC3 pin and other functions, the REMC3 output function must be assigned to the port before activating the REMC3. For more information, refer to the "I/O Ports" chapter.

18.2.2 External Connections

Figure 18.2.2.1 shows a connection example between the REMC3 and an external infrared module.

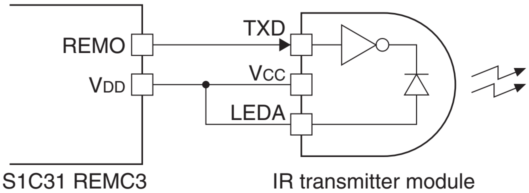


Figure 18.2.2.1 Connection Example Between REMC3 and External Infrared Module

18.3 Clock Settings

18.3.1 REMC3 Operating Clock

When using the REMC3, the REMC3 operating clock CLK_REMC3 must be supplied to the REMC3 from the clock generator. The CLK_REMC3 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 REMC3CLK register bits:
   - REMC3CLK.CLKSRC[1:0] bits (Clock source selection)
   - REMC3CLK.CLKDIV[3:0] bits (Clock division ratio selection = Clock frequency setting)

18.3.2 Clock Supply in SLEEP Mode

When using REMC3 during SLEEP mode, the REMC3 operating clock CLK_REMC3 must be configured so that it will keep supplying by writing 0 to the CLGOSC.*xxxx*SLPC bit for the CLK_REMC3 clock source.
If the CLGOSC.*xxxx*SLPC bit for the CLK_REMC3 clock source is 1, the CLK_REMC3 clock source is deactivated during SLEEP mode and REMC3 stops with the register settings maintained at those before entering SLEEP mode. After the CPU returns to normal mode, CLK_REMC3 is supplied and the REMC3 operation resumes.

18.3.3 Clock Supply During Debugging

The CLK_REMC3 supply during debugging should be controlled using the REMC3CLK.DBRUN bit.
The CLK_REMC3 supply to the REMC3 is suspended when the CPU enters debug state if the REMC3CLK.DBRUN bit = 0. After the CPU returns to normal operation, the CLK_REMC3 supply resumes. Although the REMC3 stops operating when the CLK_REMC3 supply is suspended, the output pin and registers retain the status before debug state was entered. If the REMC3CLK.DBRUN bit = 1, the CLK_REMC3 supply is not suspended and the REMC3 will keep operating in debug state.

18.4 Operations

18.4.1 Initialization

The REMC3 should be initialized with the procedure shown below.

1. Write 1 to the REMC3DBCTL.REMCRST bit. (Reset REMC3)
2. Configure the REMC3CLK.CLKSRC[1:0] and REMC3CLK.CLKDIV[3:0] bits. (Configure operating clock)
3. Assign the REMC3 output function to the port. (Refer to the "I/O Ports" chapter.)

4. Configure the following REMC3DBCTL register bits:
   - Set the REMC3DBCTL.MODEN bit to 1. (Enable count operation clock)
   - REMC3DBCTL.TRMD bit (Select repeat mode/one-shot mode)
   - Set the REMC3DBCTL.BUFEN bit to 1. (Enable compare buffer)
   - REMC3DBCTL.REMOINV bit (Configure inverse logic output signal)
5. Configure the following REMC3CARR register bits:
   - REMC3CARR.CRPER[7:0] bit (Set carrier signal cycle)
   - REMC3CARR.CRDTY[7:0] bit (Set carrier signal duty)
6. Configure the following REMC3CCTL register bits:
   - REMC3CCTL.CARREN bit (Enable/disable carrier modulation)
   - REMC3CCTL.OUTINVEN bit (Configure output signal polarity)
7. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the REMC3INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the REMC3INTE register to 1. (Enable interrupts)

18.4.2 Data Transmission Procedures

Starting data transmission

The following shows a procedure to start data transmission.

1. Set the REMC3APLEN.APLEN[15:0] bits. (Set data signal duty)
2. Set the REMC3DBLEN.DBLEN[15:0] bits. (Set data signal cycle)
3. Set the following REMC3DBCTL register bits:
   - Set the REMC3DBCTL.PRESET bit to 1. (Reset internal counters)
   - Set the REMC3DBCTL.PRUN bit to 1. (Start counting)

Continuous data transmission control

The following shows a procedure to send data continuously after starting data transmission (after Step 3 above).

1. Set the duty and cycle for the subsequent data to the REMC3APLEN.APLEN[15:0] and REMC3DBLEN.DBLEN[15:0] bits, respectively, before a compare DB interrupt (REMC3INTF.DBIF bit = 1) occurs. (It is not necessary to rewrite settings when sending the same data with the current settings.)
2. Wait for a compare DB interrupt (REMC3INTF.DBIF bit = 1).
3. Repeat Steps 1 and 2 until the end of data.

Terminating data transmission

The following shows a procedure to terminate data transmission.

1. Wait for a compare DB interrupt (REMC3INTF.DBIF bit = 1).
2. Set the REMC3DBCTL.PRUN bit to 0. (Stop counting)
3. Set the REMC3DBCTL.MODEN bit to 0. (Disable count operation clock)

18.4.3 REMO Output Waveform

Carrier refers to infrared frequency in infrared remote control communication. Note, however, that carrier in this manual refers to sub-carrier used in infrared remote control communication, as REMC3 does not control infrared rays directly.

The REMC3 outputs the logical AND between the carrier signal output from the carrier generator and the data signal output from the data signal generator. Figure 18.4.3.1 shows an example of the output waveform.

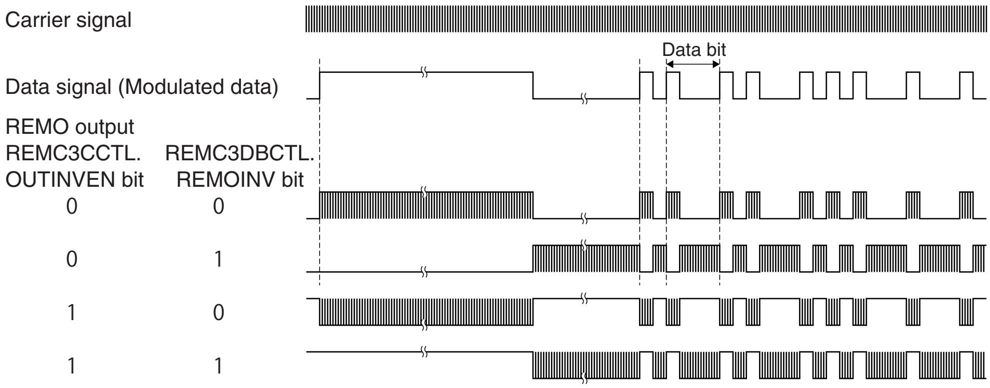


Figure 18.4.3.1 REMO Output Waveform Example

Carrier signal

The carrier signal is generated by comparing the values of the 8-bit counter for carrier generation that runs with CLK_REMC3 and the setting values of the REMC3CARR.CRDTY[7:0] and REMC3CARR.CRPER[7:0] bits. Figure 18.4.3.2 shows an example of the carrier signal generated.

Example) REMC3CARR.CRDTY[7:0] bits = 2, REMC3CARR.CRPER[7:0] bits = 8

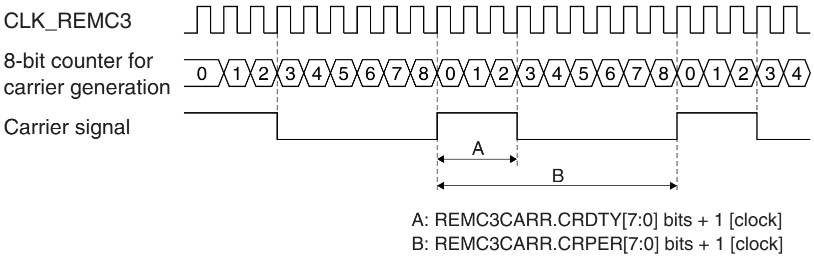


Figure 18.4.3.2 Example of Carrier Signal Generated

The carrier signal frequency and duty ratio can be calculated by the equations shown below.

$$\text{Carrier frequency} = \frac{f_{CLK\_REMC3}}{CRPER + 1} \qquad \text{Duty ratio} = \frac{CRDTY + 1}{CRPER + 1} \qquad \text{(Eq. 18.1)}$$

Where

f_{CLK_REMC3}: CLK_REMC3 frequency [Hz]
CRPER: REMC3CARR.CRPER[7:0] bit-setting value (1–255)
CRDTY: REMC3CARR.CRDTY[7:0] bit-setting value (0–254)
* REMC3CARR.CRDTY[7:0] bits < REMC3CARR.CRPER[7:0] bits

The 8-bit counter for carrier generation is reset by the REMC3DBCTL.PRESET bit and is started/stopped by the REMC3DBCTL.PRUN bit in conjunction with the 16-bit counter for data signal generation. When the counter value is matched with the REMC3CARR.CRDTY[7:0] bits, the carrier signal waveform is inverted. When the counter value is matched with the REMC3CARR.CRPER[7:0] bits, the carrier signal waveform is inverted and the counter is reset to 0x00.

Data signal

The data signal is generated by comparing the values of the 16-bit counter for data signal generation (REMC3DBCNT.DBCNT[15:0] bits) that runs with CLK_REMC3 and the setting values of the REMC3APLEN.APLEN[15:0] and REMC3DBLEN.DBLEN[15:0] bits. Figure 18.4.3.3 shows an example of the data signal generated.

Example) REMC3APLEN.APLEN[15:0] bits = 0x0bd0, REMC3DBLEN.DBLEN[15:0] bits = 0x11b8, REMC3DBCTL.TRMD bit = 0 (repeat mode), REMC3DBCTL.REMOINV bit = 0 (signal logic non-inverted)

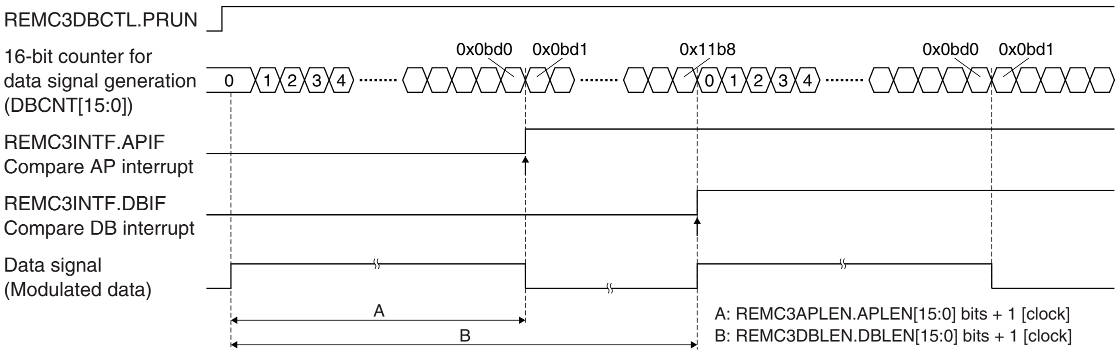


Figure 18.4.3.3 Example of Data Signal Generated

The data length and duty ratio of the pulse-width-modulated data signal can be calculated with the equations shown below.

$$\text{Data length} = \frac{\text{DBLEN} + 1}{f_{\text{CLK\_REMC3}}} \qquad \text{Duty ratio} = \frac{\text{APLEN} + 1}{\text{DBLEN} + 1} \qquad \text{(Eq. 18.2)}$$

Where
$f_{\text{CLK\_REMC3}}$: CLK_REMC3 frequency [Hz]
DBLEN: REMC3DBLEN.DBLEN[15:0] bit-setting value (1–65,535)
APLEN: REMC3APLEN.APLEN[15:0] bit-setting value (0–65,534)
* REMC3APLEN.APLEN[15:0] bits < REMC3DBLEN.DBLEN[15:0] bits

The 16-bit counter for data signal generation is reset by the REMC3DBCTL.PRESET bit and is started/stopped by the REMC3DBCTL.PRUN bit. When the counter value is matched with the REMC3APLEN.APLEN[15:0] bits (compare AP), the data signal waveform is inverted. When the counter value is matched with the REMC3DBLEN.DBLEN[15:0] bits (compare DB), the data signal waveform is inverted and the counter is reset to 0x0000.
A different interrupt can be generated when the counter value is matched with the REMC3DBLEN.DBLEN[15:0] and REMC3APLEN.APLEN[15:0] bits, respectively.

Repeat mode and one-shot mode

When the 16-bit counter for data signal generation is set to repeat mode (REMC3DBCTL.TRMD bit = 0), the counter keeps operating until it is stopped using the REMC3DBCTL.PRUN bit. When the counter is set to one-shot mode (REMC3DBCTL.TRMD bit = 1), the counter stops automatically when the counter value is matched with the REMC3DBLEN.DBLEN[15:0] bit-setting value.

18.4.4 Continuous Data Transmission and Compare Buffers

Figure 18.4.4.1 shows an operation example of continuous data transmission with the compare buffer enabled.

Example) REMC3DBCTL.TRMD bit = 0 (repeat mode), REMC3DBCTL.BUFEN bit = 1 (compare buffer enabled), REMC3DBCTL.REMOINV bit = 0 (signal logic non-inverted)

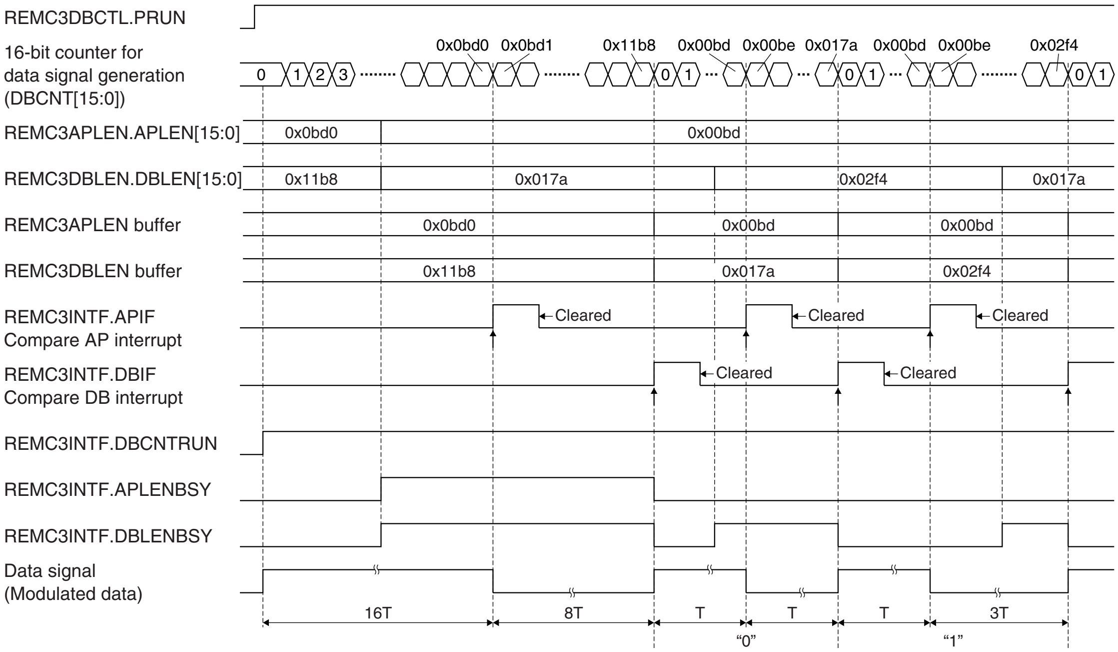


Figure 18.4.4.1 Continuous Data Transmission Example

When the compare buffer is disabled (REMC3DBCTL.BUFEN bit = 0), the 16-bit counter value is directly compared with the REMC3APLEN.APLEN[15:0] and REMC3DBLEN.DBLEN[15:0] bit values. The comparison value is altered immediately after the REMC3APLEN.APLEN[15:0] or REMC3DBLEN.DBLEN[15:0] bits are rewritten.

When the compare buffer is enabled (REMC3DBCTL.BUFEN bit = 1), the REMC3APLEN.APLEN[15:0] and REMC3DBLEN.DBLEN[15:0] bit values are loaded into the compare buffers provided respectively (REMC3APLEN buffer and REMC3DBLEN buffer) and the 16-bit counter value is compared with the compare buffers.

The comparison values are loaded into the compare buffers when the 16-bit counter is matched with the REMC3DBLEN buffer (when the count for the data length has completed). Therefore, the next transmit data can be set during the current data transmission. When the compare buffers are enabled, the buffer status flags (REMC3INTF.APLENBSY bit and REMC3INTF.DBLENBSY bit) become effective. The flag is set to 1 when the setting value is written to the register and cleared to 0 when the written value is transferred to the buffer.

18.5 Interrupts

The REMC3 has a function to generate the interrupts shown in Table 18.5.1.

Table 18.5.1 REMC3 Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Compare AP	REMC3INTF.APIF	When the REMC3APLEN register (or REMC3APLEN buffer) value and the 16-bit counter for data signal generation are matched	Writing 1 to the interrupt flag or the REMC3DBCTL.REMCRST bit
Compare DB	REMC3INTF.DBIF	When the REMC3DBLEN register (or REMC3DBLEN buffer) value and the 16-bit counter for data signal generation are matched	Writing 1 to the interrupt flag or the REMC3DBCTL.REMCRST bit

The REMC3 provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU 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” chapter.

18.6 Application Example: Driving EL Lamp

The REMC3 can be used to simply drive an EL lamp as an application example. Figures 18.6.1 and 18.6.2 show an example of an EL lamp drive circuit and an example of the drive waveform generated, respectively. For details of settings and an example of components, refer to the Application Note provided separately.

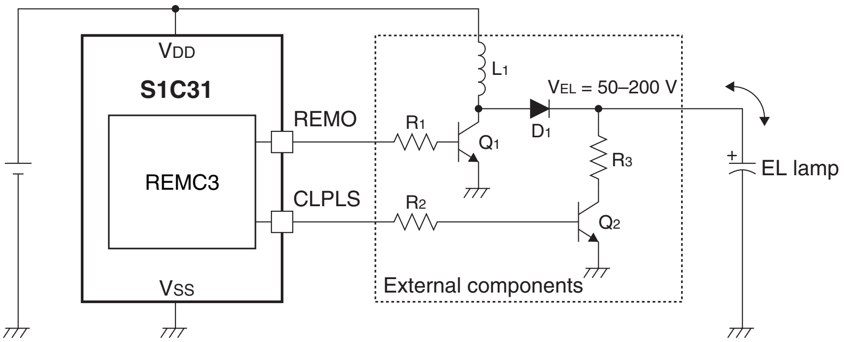


Figure 18.6.1 Example of EL Lamp Drive Circuit

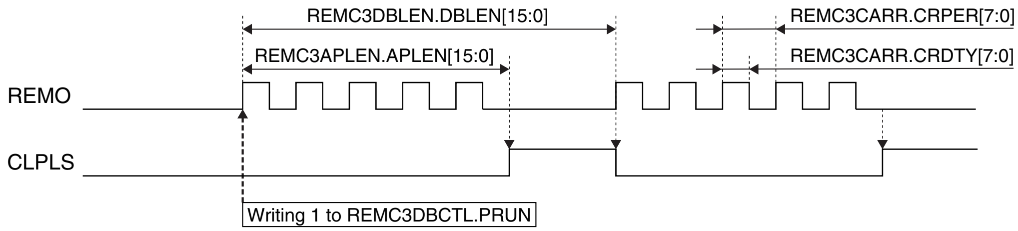


Figure 18.6.2 Example of Generated Drive Waveform

The REMO and CLPLS signals are output from the respective pins while the REMC3DBCTL.PRUN bit = 1. The difference between the setting values of the REMC3DBLEN.DBLEN[15:0] bits and REMC3APLEN.APLEN[15:0] bits becomes the CLPLS pulse width (high period).

18.7 Control Registers

REMC3 Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3CLK	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 REMC3 operating clock is supplied during debugging or not.
1 (R/W): Clock supplied during debugging
0 (R/W): No clock supplied during debugging

Bits 7–4 CLKDIV[3:0]

These bits select the division ratio of the REMC3 operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of the REMC3.

Table 18.7.1 Clock Source and Division Ratio Settings

REMC3CLK. CLKDIV[3:0] bits	REMC3CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
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) The oscillation circuits/external input that are not supported in this IC cannot be selected as the clock source.

Note: The REMC3CLK register settings can be altered only when the REMC3DBCTL.MODEN bit = 0.

REMC3 Data Bit Counter Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3DBCTL	15–10	–	0x00	–	R	–
	9	PRESET	0	H0/S0	R/W	Cleared by writing 1 to the REMC3DBCTL.REMCRST bit.
	8	PRUN	0	H0/S0	R/W	
	7–5	–	0x0	–	R	–
	4	REMOINV	0	H0	R/W	
	3	BUFEN	0	H0	R/W	
	2	TRMD	0	H0	R/W	
	1	REMCRST	0	H0	W	
	0	MODEN	0	H0	R/W	

Bits 15–10 Reserved

Bit 9 PRESET

This bit resets the internal counters (16-bit counter for data signal generation and 8-bit counter for carrier generation).

1 (W): Reset
0 (W): Ineffective
1 (R): Resetting in progress
0 (R): Resetting finished or normal operation

Before the counter can be reset using this bit, the REMC3DBCTL.MODEN bit must be set to 1.
This bit is cleared to 0 after the counter reset operation has finished or when 1 is written to the REMC3DBCTL.REMCRST bit.

Bit 8 PRUN

This bit starts/stops counting by the internal counters (16-bit counter for data signal generation and 8-bit counter for carrier generation).

1 (W): Start counting
0 (W): Stop counting
1 (R): Counting
0 (R): Idle

Before the counter can start counting by this bit, the REMC3DBCTL.MODEN bit must be set to 1. While the counter is running, writing 0 to the REMC3DBCTL.PRUN bit stops count operations. When the counter stops by occurrence of a compare DB in one-shot mode, this bit is automatically cleared to 0.

Bits 7–5 Reserved

Bit 4 REMOINV

This bit inverts the REMO output signal.

1 (R/W): Inverted
0 (R/W): Non-inverted

For more information, see Figure 18.4.3.1.

Bit 3 BUFEN

This bit enables or disables the compare buffers.

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

For more information, refer to “Continuous Data Transmission and Compare Buffers.”

Note: The REMC3DBCTL.BUFEN bit must be set to 0 when setting the data signal duty and cycle for the first time.

Bit 2 TRMD

This bit selects the operation mode of the 16-bit counter for data signal generation.

1 (R/W): One-shot mode
0 (R/W): Repeat mode

For more information, refer to “REMO Output Waveform, Data signal.”

Bit 1 REMCRST

This bit issues software reset to the REMC3.

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 REMC3 internal counters and interrupt flags. This bit is automatically cleared after the reset processing has finished.

Note: After the data signal is output in one-shot mode, set the REMC3DBCTL.REMCRST bit to 1.

Bit 0 MODEN

This bit enables the REMC3 operations.

1 (R/W): Enable REMC3 operations (The operating clock is supplied.)
0 (R/W): Disable REMC3 operations (The operating clock is stopped.)

Note: If the REMC3DBCTL.MODEN bit is altered from 1 to 0 while sending data, the data being sent cannot be guaranteed. When setting the REMC3DBCTL.MODEN bit to 1 again after that, be sure to write 1 to the REMC3DBCTL.REMCRST bit as well.

REMC3 Data Bit Counter Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3DBCNT	15–0	DBCNT[15:0]	0x0000	H0/S0	R	Cleared by writing 1 to the REMC3DBCTL.REMCRST bit.

Bits 15–0 DBCNT[15:0]

The current value of the 16-bit counter for data signal generation can be read out through these bits.

REMC3 Data Bit Active Pulse Length Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3APLEN	15–0	APLEN[15:0]	0x0000	H0	R/W	Writing enabled when REMC3DBCTL. MODEN bit = 1.

Bits 15–0 APLEN[15:0]

These bits set the active pulse length of the data signal (high period when the REMC3DBCTL.REMOINV bit = 0 or low period when the REMC3DBCTL.REMOINV bit = 1).

The REMO pin output is set to the active level from the 16-bit counter for data signal generation = 0x0000 and it is inverted to the inactive level when the counter exceeds the REMC3APLEN. APLEN[15:0] bit-setting value. The data signal duty ratio is determined by this setting and the REMC3DBLEN.DBLEN[15:0] bit-setting. (See Figure 18.4.3.3.)

Before this register can be rewritten, the REMC3DBCTL.MODEN bit must be set to 1.

REMC3 Data Bit Length Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3DBLEN	15–0	DBLEN[15:0]	0x0000	H0	R/W	Writing enabled when REMC3DBCTL. MODEN bit = 1.

Bits 15–0 DBLEN[15:0]

These bits set the data length of the data signal (length of one cycle).

A data signal cycle begins with the 16-bit counter for data signal generation = 0x0000 and ends when the counter exceeds the REMC3DBLEN.DBLEN[15:0] bit-setting value. (See Figure 18.4.3.3.)

Before this register can be rewritten, the REMC3DBCTL.MODEN bit must be set to 1.

REMC3 Status and Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3INTF	15–11	–	0x00	–	R	–
	10	DBCNTRUN	0	H0/S0	R	Cleared by writing 1 to the REMC3DBCTL.REMCRST bit.
	9	DBLENBSY	0	H0	R	Effective when the REMC3DBCTL. BUFEN bit = 1.
	8	APLENBSY	0	H0	R	
	7–2	–	0x00	–	R	–
	1	DBIF	0	H0/S0	R/W	Cleared by writing 1 to this bit or the REMC3DBCTL.REMCRST bit.
	0	APIF	0	H0/S0	R/W	

Bits 15–11 Reserved

Bit 10 DBCNTRUN

This bit indicates whether the 16-bit counter for data signal generation is running or not. (See Figure 18.4.4.1.)

1 (R): Running (Counting)

0 (R): Idle

Bit 9 DBLENBSY

This bit indicates whether the value written to the REMC3DBLEN.DBLEN[15:0] bits is transferred to the REMC3DBLEN buffer or not. (See Figure 18.4.4.1.)

1 (R): Transfer to the REMC3DBLEN buffer has not completed.

0 (R): Transfer to the REMC3DBLEN buffer has completed.

While this bit is set to 1, writing to the REMC3DBLEN.DBLEN[15:0] bits is ineffective.

Bit 8 APLENBSY

This bit indicates whether the value written to the REMC3APLEN.APLEN[15:0] bits is transferred to the REMC3APLEN buffer or not. (See Figure 18.4.4.1.)

1 (R): Transfer to the REMC3APLEN buffer has not completed.

0 (R): Transfer to the REMC3APLEN buffer has completed.

While this bit is set to 1, writing to the REMC3APLEN.APLEN[15:0] bits is ineffective.

Bits 7–2 Reserved

Bit 1 DBIF

Bit 0 APIF

These bits indicate the REMC3 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:
REMC3INTF.DBIF bit: Compare DB interrupt
REMC3INTF.APIF bit: Compare AP interrupt

These interrupt flags are also cleared to 0 when 1 is written to the REMC3DBCTL.REMCRST bit.

REMC3 Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3INTE	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	DBIE	0	H0	R/W	
	0	APIE	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 DBIE

Bit 0 APIE

These bits enable REMC3 interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
REMC3INTE.DBIE bit: Compare DB interrupt
REMC3INTE.APIE bit: Compare AP interrupt

REMC3 Carrier Waveform Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3CARR	15–8	CRDTY[7:0]	0x00	H0	R/W	–
	7–0	CRPER[7:0]	0x00	H0	R/W	

Bits 15–8 CRDTY[7:0]

These bits set the high level period of the carrier signal.
The carrier signal is set to high level from the 8-bit counter for carrier generation = 0x00 and it is inverted to low level when the counter exceeds the REMC3CARR.CRDTY[7:0] bit-setting value. The carrier signal duty ratio is determined by this setting and the REMC3CARR.CRPER[7:0] bit-setting. (See Figure 18.4.3.2.)

Bits 7–0 CRPER[7:0]

These bits set the carrier signal cycle.
A carrier signal cycle begins with the 8-bit counter for carrier generation = 0x00 and ends when the counter exceeds the REMC3CARR.CRPER[7:0] bit-setting value. (See Figure 18.4.3.2.)

REMC3 Carrier Modulation Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REMC3CCTL	15–9	–	0x00	–	R	–
	8	OUTINVEN	0	H0	R/W	
	7–1	–	0x00	–	R	
	0	CARREN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 OUTINVEN

This bit inverts the REMO output polarity.

1 (R/W): Inverted

0 (R/W): Non-inverted

For more information, see Figure 18.4.3.1.

Bits 7–1 Reserved

Bit 0 CARREN

This bit enables carrier modulation.

1 (R/W): Enable carrier modulation

0 (R/W): Disable carrier modulation (output data signal only)

Note: When carrier modulation is disabled, the REMC3DBCTL.REMOINV bit should be set to 0.

19 12-bit A/D Converter (ADC12A)

19.1 Overview

The ADC12A is a successive approximation type 12-bit A/D converter.
The features of the ADC12A are listed below.

- Conversion method: Successive approximation type
- Resolution: 12 bits
- Analog input voltage range: Reference voltage VREFA to Vss
- Supports two conversion modes: 1. One-time conversion mode
  2. Continuous conversion mode
- Supports three conversion triggers: 1. Software trigger
  2. 16-bit timer underflow trigger
  3. External trigger
- Can convert multiple analog input signals sequentially.
- Can generate conversion completion and overwrite error interrupts.
- Can issue a DMA transfer request when a conversion has completed.

Figure 19.1.1 shows the ADC12A configuration.

Table 19.1.1 ADC12A Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	1 channel (Ch.0)		
Number of analog signal inputs per channel	Ch.0: 6 inputs (ADIN00–04, 07 *1)		Ch.0: 8 inputs (ADIN00–07 *1)
16-bit timer used as conversion clock and trigger sources	Ch.0 ← 16-bit timer Ch.7		
VREFA pin (reference voltage input)	Can be input externally or generated internally *2		

*1 ADIN07 is connected to the temperature sensor output.
*2 The reference voltage generator output can be input as the reference voltage.
For more information, refer to the “Temperature Sensor/Reference Voltage Generator” chapter.

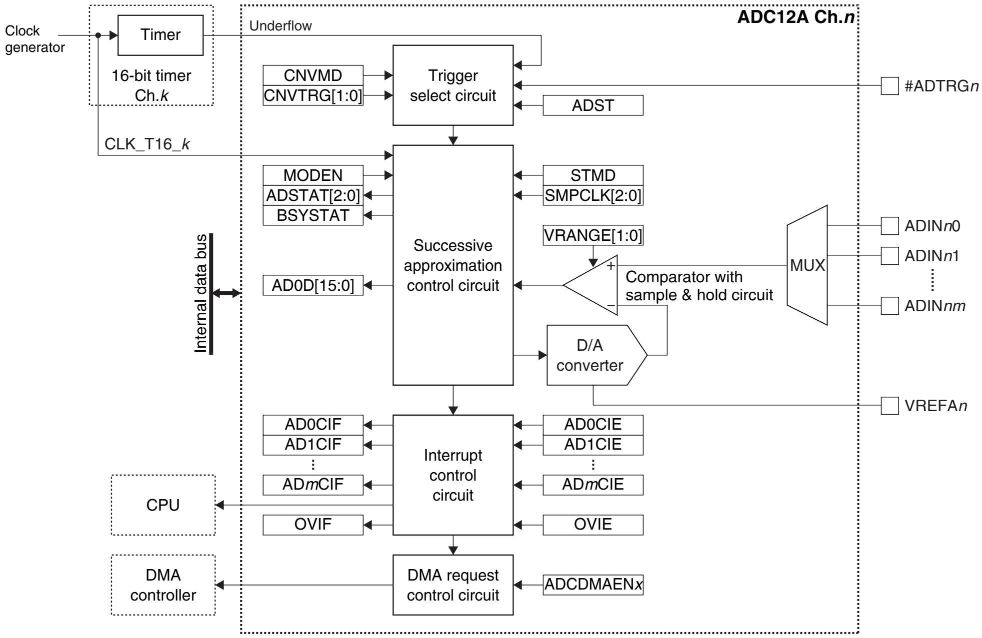


Figure 19.1.1 ADC12A Configuration

Note: In this chapter, *n*, *m*, and *k* refer to an ADC12A channel number, an analog input pin number, and a 16-bit timer channel number, respectively.

19.2 Input Pins and External Connections

19.2.1 List of Input Pins

Table 19.2.1.1 lists the ADC12A pins.

Table 19.2.1.1 List of ADC12A Pins

Pin name	I/O*	Initial status*	Function
ADIN*nm*	A	Hi-Z	Analog signal input
#ADTRG*n*	I	I	External trigger input
VREFA*n*	A	Hi-Z	Reference voltage input

* Indicates the status when the pin is configured for the ADC12A.

If the port is shared with the ADC12A pin and other functions, the ADC12A input function must be assigned to the port before activating the ADC12A. For more information, refer to the "I/O Ports" chapter.

19.2.2 External Connections

Figure 19.2.2.1 shows a connection diagram between the ADC12A and external devices.

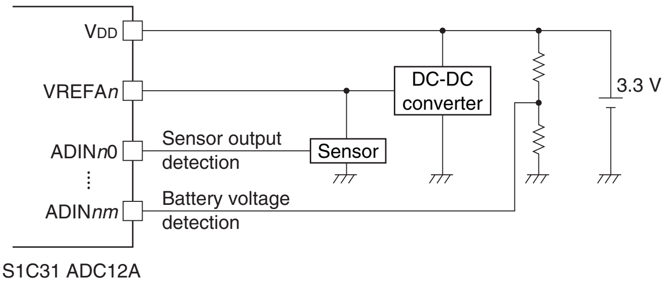


Figure 19.2.2.1 Connections between ADC12A and External Devices

19.3 Clock Settings

19.3.1 ADC12A Operating Clock

The 16-bit timer Ch.*k* operating clock CLK_T16_*k* is also used as the ADC12A operating clock. For more information on the CLK_T16_*k* settings and clock supply in SLEEP and DEBUG modes, refer to "Clock Settings" in the "16-bit Timers" chapter.

Note: When the CLK_T16_*k* supply stops during A/D conversion (e.g., when the CPU enters SLEEP or DEBUG mode), correct conversion results cannot be obtained even if the clock supply is resumed after that. In this case, perform A/D conversion again.

19.3.2 Sampling Time

The ADC12A includes a sample and hold circuit. The sampling time must be set so that it will satisfy the time required for acquiring input voltage (t_{ACQ}: acquisition time). Figure 19.3.2.1 shows an equivalent circuit of the analog input portion.

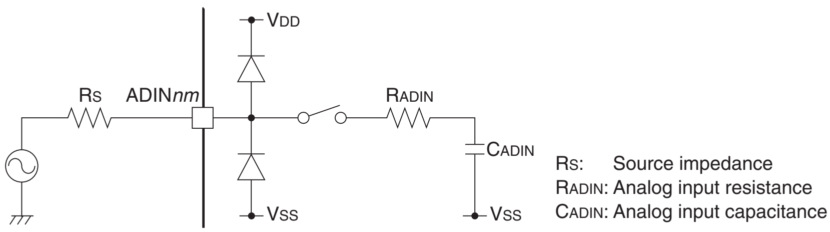


Figure 19.3.2.1 Equivalent Circuit of Analog Input Portion

For the RADIN and CADIN values in the equivalent circuit, refer to "12-bit A/D Converter Characteristics" in the "Electrical Characteristics" chapter. Based on these values, configure the ADC12A operating clock CLK_T16_*k* and the ADC12A_*n*TRG.SMPCLK[2:0] bits that set the sampling time so that these settings will satisfy the equations shown below.

$$t_{ACQ} = 8 \times (R_S + R_{ADIN}) \times C_{ADIN} \qquad \text{(Eq. 19.1)}$$

$$\frac{1}{f_{CLK\_ADC}} \times SMPCLK > t_{ACQ} \qquad \text{(Eq. 19.2)}$$

Where

f_{CLK_ADC}: CLK_T16_*k* frequency [Hz]

SMPCLK: Sampling time = ADC12A_*n*TRG.SMPCLK[2:0] bit-setting (4 to 11 CLK_T16_*k* cycles)

The following shows the relationship between the sampling time and the maximum sampling rate.

$$\text{Maximum sampling rate [sps]} = \frac{f_{CLK\_ADC}}{SMPCLK + 13} \qquad \text{(Eq. 19.3)}$$

19.4 Operations

19.4.1 Initialization

The ADC12A should be initialized with the procedure shown below.

1. Assign the ADC12A input function to the ports. (Refer to the "I/O Ports" chapter.)
2. Configure the 16-bit timer Ch.*k* operating clock so that it will satisfy the sampling time.
3. Set the ADC12A_*n*CTL.MODEN bit to 1. (Enable ADC12A operations)
4. Configure the following ADC12A_*n*TRG register bits:
   - ADC12A_*n*TRG.SMPCLK[2:0] bits (Set sampling time)
   - ADC12A_*n*TRG.CNVTRG[1:0] bits (Select conversion start trigger source)
   - ADC12A_*n*TRG.CNVMD bit (Set conversion mode)
   - ADC12A_*n*TRG.STMD bit (Set data storing mode)
   - ADC12A_*n*TRG.STAAIN[2:0] bits (Set analog input pin to be A/D converted first)
   - ADC12A_*n*TRG.ENDAIN[2:0] bits (Set analog input pin to be A/D converted last)
5. Set the ADC12A_*n*CFG.VRANGE[1:0] bits. (Set operating voltage range according to V_{DD})
6. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the ADC12A_*n*INTF register. (Clear interrupt flags)
   - Set the interrupt enable bits in the ADC12A_*n*INTE register to 1. (Enable interrupts)
7. Configure the DMA controller and set the following ADC12A control bit when using DMA transfer:
   - Write 1 to the DMA transfer request enable bit in the ADC12A_*n*DMAEN register. (Enable DMA transfer requests)

19.4.2 Conversion Start Trigger Source

The trigger source, which starts A/D conversion, can be selected from the three types shown below using the ADC12A_*n*TRG.CNVTRG[1:0] bits.

External trigger (#ADTRG*n* pin)

Writing 1 to the ADC12A_*n*CTL.ADST bit enables the ADC12A to accept trigger inputs. After that, the falling edge of the signal input to the #ADTRG*n* pin starts A/D conversion.

16-bit timer Ch.*k* underflow trigger

Writing 1 to the ADC12A_*n*CTL.ADST bit enables the ADC12A to accept trigger inputs. After that, A/D conversion is started when an underflow occurs in the 16-bit timer Ch.*k*.

Software trigger

Writing 1 to the ADC12A_*n*CTL.ADST bit starts A/D conversion.

Trigger inputs can be accepted while the ADC12A_*n*CTL.BSYSTAT bit is set to 0 and are ignored while set to 1.
A/D conversion is actually started in sync with CLK_T16_*k* after a trigger is accepted.
Writing 0 to the ADC12A_*n*CTL.ADST bit stops A/D conversion after the one currently being executed has completed.

19.4.3 Conversion Mode and Analog Input Pin Settings

The ADC12A can be put into two conversion modes shown below using the ADC12A_*n*TRG.CNVMD bit. Each mode allows setting of analog input pin range to be A/D converted. The analog input pin range can be set using the ADC12A_*n*TRG.STAAIN[2:0] bits for specifying the first analog input pin and the ADC12A_*n*TRG.ENDAIN[2:0] bits for specifying the last analog input pin. The analog input signals within the specified range are A/D converted successively in ascending order of the pin numbers.

One-time conversion mode

Once the ADC12A executes A/D conversion for all the analog input signals within the specified range, it is automatically stopped.

Continuous conversion mode

The ADC12A repeatedly executes A/D conversion within the specified range until 0 is written to the ADC12A_*n*CTL.ADST bit.

19.4.4 A/D Conversion Operations and Control Procedures

The following shows A/D conversion control procedures and the ADC12A operations.

Control procedure in one-time conversion mode

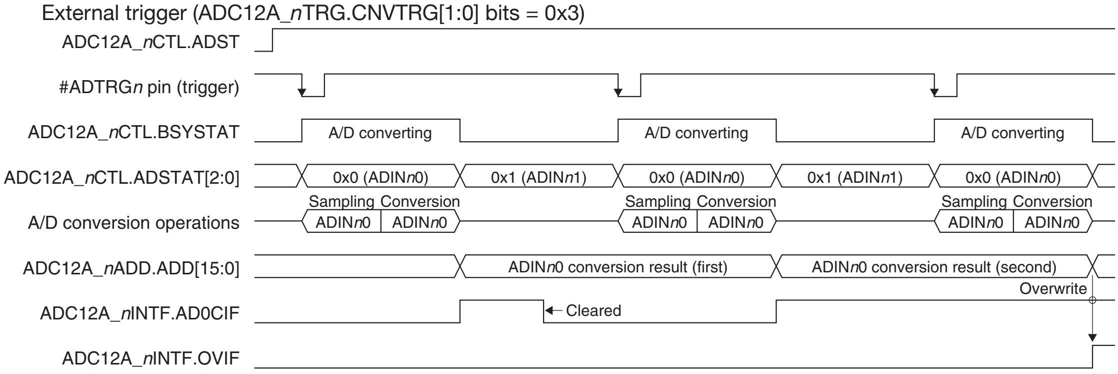
1. Write 1 to the ADC12A_*n*CTL.ADST bit.
2. Wait for an ADC12A interrupt.
   i. If the ADC12A_*n*INTF.AD*m*CIF bit = 1 (analog input signal *m* A/D conversion completion interrupt), clear the ADC12A_*n*INTF.AD*m*CIF bit and then go to Step 3.
   ii. If the ADC12A_*n*INTF.OVIF bit = 1 (A/D conversion result overwrite error interrupt), clear the ADC12A_*n*INTF.OVIF bit and terminate as an error or retry A/D conversion.
3. Read the A/D conversion result of the analog input *m* (ADC12A_*n*ADD.ADD[15:0] bits).
   * The 12-bit conversion results are located at the low-order 12 bits or high-order 12-bits within the ADC12A_*n*ADD.ADD[15:0] bits according to the ADC12A_*n*TRG.STMD bit setting.
4. Repeat Steps 2 and 3 until A/D conversion for all the analog input pins within the specified range is completed.
5. To forcefully terminate the A/D conversion being executed, write 0 to the ADC12A_*n*CTL.ADST bit.
   The ADC12A stops operating after the A/D conversion currently being executed has completed.

   The ADC12A_*n*CTL.ADST bit must be cleared by writing 0 even if A/D conversion is completed and automatically stopped.

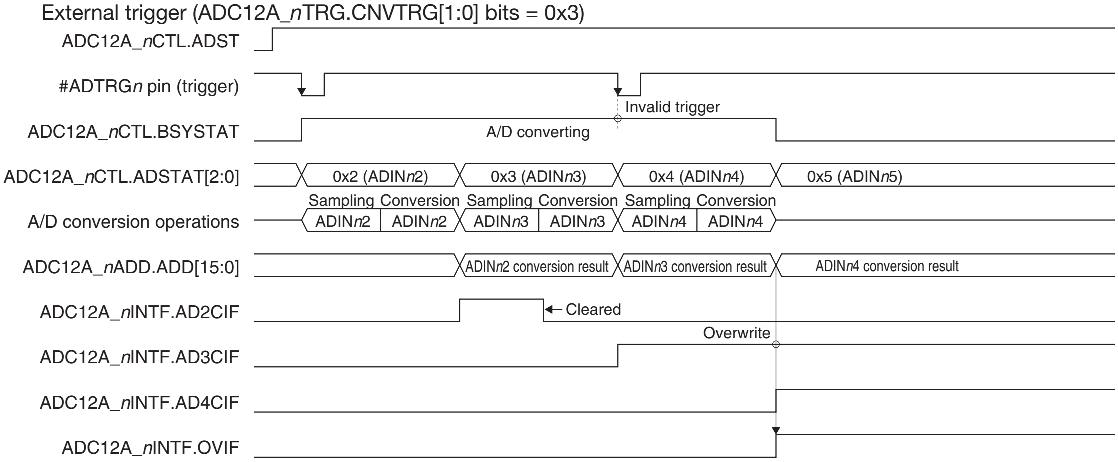
Control procedure in continuous conversion mode

1. Write 1 to the ADC12A_*n*CTL.ADST bit.
2. Wait for an ADC12A interrupt.
   i. If the ADC12A_*n*INTF.AD*m*CIF bit = 1 (analog input signal *m* A/D conversion completion interrupt), clear the ADC12A_*n*INTF.AD*m*CIF bit and then go to Step 3.
   ii. If the ADC12A_*n*INTF.OVIF bit = 1 (A/D conversion result overwrite error interrupt), clear the ADC12A_*n*INTF.OVIF bit and terminate as an error or retry A/D conversion.
3. Read the A/D conversion result of the analog input *m* (ADC12A_*n*ADD.ADD[15:0] bits).
4. Repeat Steps 2 and 3 until terminating A/D conversion.
5. Write 0 to the ADC12A_*n*CTL.ADST bit.
   The ADC12A stops operating after the A/D conversion currently being executed has completed.

(1) One-time conversion mode (ADC12A_*n*TRG.CNVMD bit = 0)
A/D conversion for ADIN*n*0 (ADC12A_*n*TRG.STAAIN[2:0] bits = 0x0, ADC12A_*n*TRG.ENDAIN[2:0] bits = 0x0)
External trigger (ADC12A_*n*TRG.CNVTRG[1:0] bits = 0x3)



(2) One-time conversion mode (ADC12A_*n*TRG.CNVMD bit = 0)
A/D conversion for ADIN*n*2–4 (ADC12A_*n*TRG.STAAIN[2:0] bits = 0x2, ADC12A_*n*TRG.ENDAIN[2:0] bits = 0x4)
External trigger (ADC12A_*n*TRG.CNVTRG[1:0] bits = 0x3)



(3) Continuous conversion mode (ADC12A_*n*TRG.CNVMD bit = 1)
A/D conversion for ADIN*n*3–4 (ADC12A_*n*TRG.STAAIN[2:0] bits = 0x3, ADC12A_*n*TRG.ENDAIN[2:0] bits = 0x4)
Software trigger (ADC12A_*n*TRG.CNVTRG[1:0] bits = 0x0)

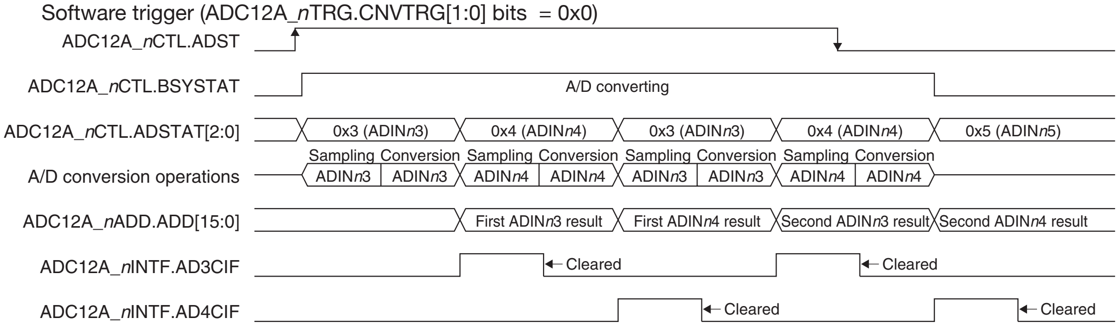


Figure 19.4.4.1 A/D Conversion Operations

A/D converted data transfer using DMA

By setting the ADC12A_*n*DMAEN.ADCDMAEN*x* bit to 1 (DMA transfer request enabled), a DMA transfer request is sent to the DMA controller and the ADC12A_*n*ADD register value is transferred to the specified memory via DMA Ch.*x* when the ADC12A_*n*INTF.AD*m*CIF bit is set to 1 (when A/D conversion for the analog input signal *m* has completed).
This automates reading and saving of A/D converted data.
The transfer source/destination and control data must be set for the DMA controller and the relevant DMA channel must be enabled to start a DMA transfer in advance. For more information on DMA, refer to the "DMA Controller" chapter.

Table 19.4.4.1 DMA Data Structure Configuration Example (Capture Data Transfer)

Item		Setting example
End pointer	Transfer source	ADC12A_*n*ADD register address
	Transfer destination	Memory address to which the last A/D converted data is stored
Control data	dst_inc	0x1 (+2)
	dst_size	0x1 (haflword)
	src_inc	0x3 (no increment)
	src_size	0x1 (halfword)
	R_power	0x0 (arbitrated for every transfer)
	n_minus_1	Number of transfer data
	cycle_ctrl	0x1 (basic transfer)

19.5 Interrupts

The ADC12A has a function to generate the interrupts shown in Table 19.5.1.

Table 19.5.1 ADC12A Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Analog input signal *m* A/D conversion completion	ADC12A_*n*INTF.AD*m*CIF	When an analog input signal *m* A/D conversion result is loaded to the ADC12A_*n*ADD register	Writing 1
A/D conversion result overwrite error	ADC12A_*n*INTF.OVIF	When a new A/D conversion result is loaded to the ADC12A_*n*ADD register while the ADC12A_*n*INTF.AD*m*CIF bit = 1	Writing 1

Note that the A/D conversion continues even if an A/D conversion result overwrite error has occurred. A/D conversion result overwrite errors are decided regardless of whether the ADC12A_*n*ADD register has been read or not.

The ADC12A provides interrupt enable bits corresponding to each interrupt flag. An interrupt request is sent to the CPU 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” chapter.

19.6 DMA Transfer Requests

The ADC12A has a function to generate DMA transfer requests from the causes shown in Table 19.6.1.

Table 19.6.1 DMA Transfer Request Causes of ADC12A

Cause to request DMA transfer	DMA transfer request flag	Set condition	Clear condition
Analog input signal *m* A/D conversion completion	A/D conversion completion flag (ADC12A_*n*INTF.AD*m*CIF)	When an analog input signal *m* A/D conversion result is loaded to the ADC12A_*n*ADD register	When the DMA transfer request is accepted

The ADC12A provides DMA transfer request enable bits corresponding to each DMA transfer request flag shown above for the number of DMA channels. A DMA transfer request is sent to the pertinent channel of the DMA controller only when the DMA transfer request flag, of which DMA transfer has been enabled by the DMA transfer request enable bit, is set. The DMA transfer request flag also serves as an interrupt flag, therefore, both the DMA transfer request and the interrupt cannot be enabled at the same time. After a DMA transfer has completed, disable the DMA transfer to prevent unintended DMA transfer requests from being issued. For more information on the DMA control, refer to the “DMA Controller” chapter.

19.7 Control Registers

ADC12A Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*CTL	15	–	0	–	R	–
	14–12	ADSTAT[2:0]	0x0	H0	R	
	11	–	0	–	R	
	10	BSYSTAT	0	H0	R	
	9–8	–	0x0	–	R	
	7–2	–	0x00	–	R	
	1	ADST	0	H0	R/W	
	0	MODEN	0	H0	R/W	

Bit 15 Reserved

Bits 14–12 ADSTAT[2:0]

These bits indicate the analog input pin number *m* being A/D converted.

Table 19.7.1 Relationship Between Control Bit Value and Analog Input Pin

ADC12A_*n*CTL.ADSTAT[2:0] bits ADC12A_*n*TRG.STAAIN[2:0] bits ADC12A_*n*TRG.ENDAIN[2:0] bits	Analog input pin
0x7	ADIN*n*7
0x6	ADIN*n*6
0x5	ADIN*n*5
0x4	ADIN*n*4
0x3	ADIN*n*3
0x2	ADIN*n*2
0x1	ADIN*n*1
0x0	ADIN*n*0

These bits indicate the last converted analog input pin number after A/D conversion is forcefully terminated by writing 0 to the ADC12A_*n*CTL.ADST bit or automatically terminated in one-time conversion mode (ADC12A_*n*TRG.CNVMD = 0). If A/D conversion is stopped after the maximum analog input pin number (different in each model) has been completed, these bits indicate ADIN*n*0.

Bit 11 Reserved

Bit 10 BSYSTAT

This bit indicates whether the ADC12A is executing A/D conversion or not.

1 (R/W): A/D converting

0 (R/W): Idle

Bits 9–2 Reserved

Bit 1 ADST

This bit starts A/D conversion or enables to accept triggers.

1 (R/W): Start sampling and conversion (software trigger)/
Enable trigger acceptance (external trigger, 16-bit timer underflow trigger)

0 (R/W): Terminate conversion

This bit does not revert to 0 automatically after A/D conversion has completed. Write 0 to this bit once and write 1 again to start another A/D conversion. After 0 is written to this bit to forcefully terminate conversion, the ADC12A stops after the A/D conversion being executed is completed. Therefore, this bit cannot be used to determine whether the ADC12A is executing A/D conversion or not.

Note: The data written to the ADC12A_*n*CTL.ADST bit must be retained for one or more CLK_T16_*k* clock cycles when 1 is written or two or more CLK_T16_*k* clock cycles when 0 is written.

Bit 0 MODEN

This bit enables the ADC12A operations.
1 (R/W): Enable ADC12A operations (The operating clock is supplied.)
0 (R/W): Disable ADC12A operations (The operating clock is stopped.)

Note: After 0 is written to the ADC12A_*n*CTL.MODEN bit, the ADC12A executes a terminate processing. Before the clock source is deactivated, read the ADC12A_*n*CTL.MODEN bit to make sure that it is set to 0.

ADC12A Ch.*n* Trigger/Analog Input Select Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*TRG	15–14	–	0x0	–	R	–
	13–11	ENDAIN[2:0]	0x0	H0	R/W	
	10–8	STAAIN[2:0]	0x0	H0	R/W	
	7	STMD	0	H0	R/W	
	6	CNVMD	0	H0	R/W	
	5–4	CNVTRG[1:0]	0x0	H0	R/W	
	3	–	0	–	R	
	2–0	SMPCLK[2:0]	0x7	H0	R/W	

Note: Make sure that the ADC12A_*n*CTL.BSYSTAT bit is set to 0 before altering the ADC12A_*n*TRG register.

Bits 15–14 Reserved

Bits 13–11 ENDAIN[2:0]

These bits set the analog input pin to be A/D converted last.
See Table 19.7.1 for the relationship between analog input pins and bit setting values.

Note: The analog input pin range to perform A/D conversion must be set as ADC12A_*n*TRG. ENDAIN[2:0] bits ≥ ADC12A_*n*TRG.STAAIN[2:0] bits.

Bits 10–8 STAAIN[2:0]

These bits set the analog input pin to be A/D converted first.
See Table 19.7.1 for the relationship between analog input pins and bit setting values.

Bit 7 STMD

This bit selects the data alignment when the conversion results are loaded into the A/D conversion result register (ADC12A_*n*ADD.ADD[15:0] bits).
1 (R/W): Left justify
0 (R/W): Right justify

All the A/D conversion result registers change their data alignment immediately after this bit is altered. This does not affect the conversion results.

ADC12A_*n*ADD.ADD[15:0] bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Left justified (ADC12A_*n*TRG.STMD bit = 1)	(MSB)			12-bit conversion result								(LSB)	0	0	0	0
Right justified (ADC12A_*n*TRG.STMD bit = 0)	0	0	0	0	(MSB)			12-bit conversion result								(LSB)

Figure 19.7.1 Conversion Data Alignment

Bit 6 CNVMD

This bit sets the A/D conversion mode.
1 (R/W): Continuous conversion mode
0 (R/W): One-time conversion mode

Bits 5–4 CNVTRG[1:0]

These bits select a trigger source to start A/D conversion.

Table 19.7.2 Trigger Source Selection

ADC12A_*n*TRG.CNVTRG[1:0] bits	Trigger source
0x3	#ADTRG*n* pin (external trigger)
0x2	Reserved
0x1	16-bit timer Ch.*k* underflow
0x0	ADC12A_*n*CTL.ADST bit (software trigger)

Bit 3 Reserved

Bits 2–0 SMPCLK[2:0]

These bits set the analog input signal sampling time.

Table 19.7.3 Sampling Time Settings

ADC12A_*n*TRG.SMPCLK[2:0] bits	Sampling time (Number of CLK_T16_*k* cycles)
0x7	11 cycles
0x6	10 cycles
0x5	9 cycles
0x4	8 cycles
0x3	7 cycles
0x2	6 cycles
0x1	5 cycles
0x0	4 cycles

ADC12A Ch.*n* Configuration Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*CFG	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1–0	VRANGE[1:0]	0x0	H0	R/W	

Note: Make sure that the ADC12A_*n*CTL.BSYSTAT bit is set to 0 before altering the ADC12A_*n*CFG register.

Bits 15–2 Reserved

Bits 1–0 VRANGE[1:0]

These bits set the A/D converter operating voltage range.

Table 19.7.4 A/D Converter Operating Voltage Range Setting

ADC12A_*n*CFG.VRANGE[1:0] bits	A/D converter operating voltage range
0x3	1.8 to 5.5 V
0x2	3.6 to 5.5 V
0x1	4.8 to 5.5 V
0x0	Conversion disabled

Notes:
- A/D conversion will not be performed if the ADC12_*n*CFG.VRANGE[1:0] bits = 0x0. Set these bits to the value according to the operating voltage to perform A/D conversion.
- Be aware that ADC circuit current I_{ADC} flows if the ADC12_*n*CFG.VRANGE[1:0] bits are set to a value other than 0x0 when the ADC12_*n*CTL.BSYSTAT bit = 1.

ADC12A Ch.*n* Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*INTF	15–9	–	0x00	–	R	–
	8	OVIF	0	H0	R/W	Cleared by writing 1.
	7	AD7CIF	0	H0	R/W	
	6	AD6CIF	0	H0	R/W	
	5	AD5CIF	0	H0	R/W	
	4	AD4CIF	0	H0	R/W	
	3	AD3CIF	0	H0	R/W	
	2	AD2CIF	0	H0	R/W	
	1	AD1CIF	0	H0	R/W	
	0	AD0CIF	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 OVIF

Bits 7–0 AD*m*CIF

These bits indicate the ADC12A 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:
ADC12A_*n*INTF.OVIF bit: A/D conversion result overwrite error interrupt
ADC12A_*n*INTF.AD*m*CIF bit: Analog input signal *m* A/D conversion completion interrupt

ADC12A Ch.*n* Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*INTE	15–9	–	0x00	–	R	–
	8	OVIE	0	H0	R/W	
	7	AD7CIE	0	H0	R/W	
	6	AD6CIE	0	H0	R/W	
	5	AD5CIE	0	H0	R/W	
	4	AD4CIE	0	H0	R/W	
	3	AD3CIE	0	H0	R/W	
	2	AD2CIE	0	H0	R/W	
	1	AD1CIE	0	H0	R/W	
	0	AD0CIE	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 OVIE

Bits 7–0 AD*m*CIE

These bits enable ADC12A interrupts.

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

The following shows the correspondence between the bit and interrupt:
ADC12A_*n*INTE.OVIE bit: A/D conversion result overwrite error interrupt
ADC12A_*n*INTE.AD*m*CIE bit: Analog input signal *m* A/D conversion completion interrupt

ADC12A Ch.*n* DMA Request Enable Register *m*

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*DMAEN*m*	15–0	ADCDMAEN[15:0]	0x0000	H0	R/W	–

Bits 15–0 ADCDMAEN[15:0]

These bits enable ADC12A to issue a DMA transfer request to the corresponding DMA controller channel (Ch.0–Ch.15) when the A/D conversion for each analog input has completed.

1 (R/W): Enable DMA transfer request

0 (R/W): Disable DMA transfer request

Each bit corresponds to a DMA controller channel. The high-order bits for the unimplemented channels are ineffective.

ADC12A Ch.*n* Result Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ADC12A_*n*ADD	15–0	ADD[15:0]	0x0000	H0	R	–

Bits 15–0 ADD[15:0]

The A/D conversion results are set to these bits.

20 Temperature Sensor/Reference Voltage Generator (TSRVR)

20.1 Overview

The TSRVR is a peripheral circuit for the internal A/D converter that outputs the internal temperature sensor detection values and generates the reference voltage. The features of the TSRVR are listed below.

- Includes a temperature sensor that has a linear output characteristic and the sensor output can be measured using the internal A/D converter without external components being attached.
- Can supply a reference voltage (2.0 V, 2.5 V, or VDD selectable) to the internal A/D converter.
- Can supply the reference voltage generated in this circuit to external devices if this IC has the VREFA exclusive pin.

Figure 20.1.1 shows the TSRVR configuration.

Table 20.1.1 TSRVR Configuration of S1C31D41

Item	S1C31D41
Number of channels	1 channel (Ch.0)
Correspondence between TSRVR and internal A/D converter channels	TSRVR Ch.0 → ADC12A Ch.0
A/D converter input connected to temperature sensor	ADIN07
Reference voltage output to external devices	Unavailable

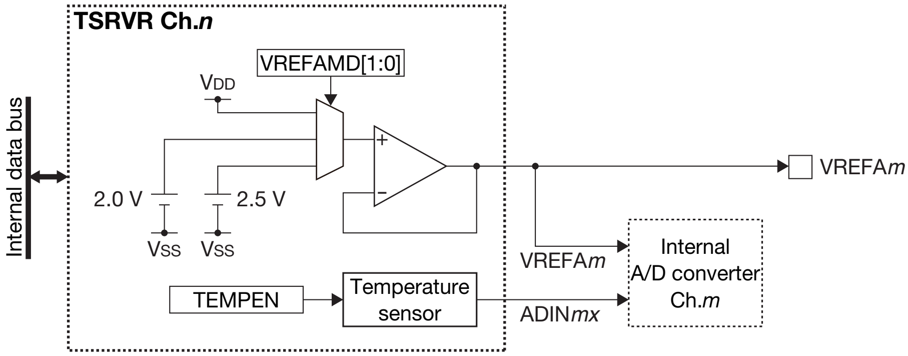


Figure 20.1.1 TSRVR Configuration

Note: In this chapter, *n* and *m* refer to a TSRVR channel number and an internal A/D converter channel number, respectively.

20.2 Output Pin and External Connections

20.2.1 Output Pin

Table 20.2.1.1 shows the TSRVR pin.

Table 20.2.1.1 TSRVR Pin

Pin name	I/O	Initial status	Function
VREFA*m*	A	Hi-Z	Reference voltage output

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

20.2.2 External Connections

Figure 20.2.2.1 shows connection diagrams between the TSRVR and external components.

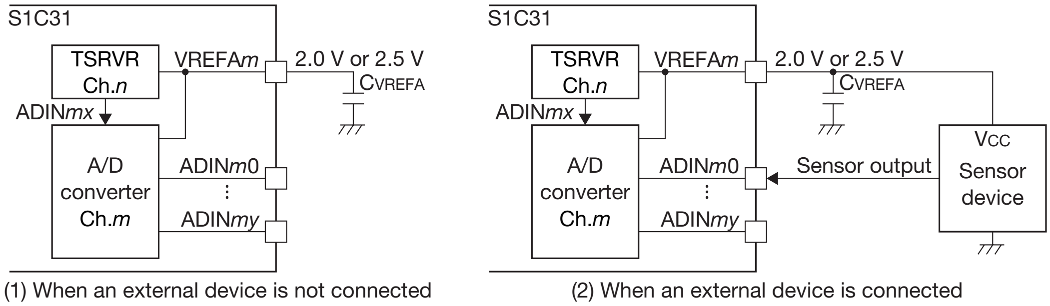


Figure 20.2.2.1 Connections between TSRVR and External Components

20.3 Operations

TSRVR should be configured before starting measurements using the internal A/D converter.

20.3.1 Reference Voltage Setting

The TSRVR output voltage can be supplied to the internal A/D converter as the reference voltage VREFA*m* when it is not supplied externally. The output voltage can be selected using the TSRVR_*n*VCTL.VREFAMD[1:0] bits. Connect C_{VREFA} to the VREFA*m* pin when supplying the reference voltage from TSRVR. A/D conversion by the internal A/D converter should be started after the reference voltage stabilization time t_{VREFA} has elapsed from the time when the output voltage is selected.

20.3.2 Temperature Sensor Setting

The temperature sensor output voltage can be directly measured using the internal A/D converter. The measurement should be started after the temperature sensor output stabilization time t_{TEMP} has elapsed from writing 1 to the TSRVR_*n*TCTL.TEMPEN bit to activate the temperature sensor.

From the temperature sensor output voltage, the measured temperature can be calculated by the equations shown below.

$$T_{SEN} = \frac{(V_{TSEN} - V_{TREF}) \times 1{,}000}{\Delta V_{TEMP}} + T_{REF} \qquad \text{(Eq. 20.1)}$$

Where

T_{SEN}: Actual temperature [°C]
V_{TSEN}: Temperature sensor output voltage at temperature T_{SEN} [V]
T_{REF}: Reference temperature for calibration [°C]
V_{TREF}: Temperature sensor output voltage at temperature T_{REF} [V]
ΔV_{TEMP}: Temperature sensor output voltage temperature coefficient [mV/°C] (Refer to the “Electrical Characteristics” chapter.)

Convert the digital values corresponding to the respective temperatures, that are obtained by the internal A/D converter, into voltage values and assign them to V_{TSEN} and V_{TREF}.

$$V_{(TSEN,\,TREF)} = \frac{ADD}{4{,}096} \times V_{REFA} \qquad \text{(Eq. 20.2)}$$

Where

ADD: A/D conversion result at temperature T_{SEN} or T_{REF} (decimal)
V_{REFA}: A/D converter reference voltage [V]

For details of the internal A/D converter, refer to the “12-bit A/D Converter” chapter.

20.4 Control Registers

TSRVR Ch.*n* Temperature Sensor Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TSRVR_*n*TCTL	15–8	–	0x00	–	R	–
	7–1	–	0x00	H0	R	
	0	TEMPEN	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 TEMPEN

This bit enables the temperature sensor operation.

1 (R/W): Enable temperature sensor output

0 (R/W): Disable temperature sensor output

TSRVR Ch.*n* Reference Voltage Generator Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
TSRVR_*n*VCTL	15–8	–	0x00	–	R	–
	7–2	–	0x00	H0	R	
	1–0	VREFAMD[1:0]	0x0	H0	R/W	

Bits 15–2 Reserved

Bits 1–0 VREFAMD[1:0]

These bits set the reference voltage generator output voltage.

Table 20.4.1 Output Voltage Settings

TSRVR_*n*VCTL.VREFAMD[1:0] bits	Output voltage
0x3	2.5 V output
0x2	2.0 V output
0x1	V_{DD} level output
0x0	Hi-Z (An external voltage can be applied.)

Notes:
- Be aware that VREFA operating current I_{VREFA} flows when the TSRVR_*n*VCTL.VREFAMD[1:0] bits are set to 0x2 or 0x3.
- When the TSRVR_*n*VCTL.VREFAMD[1:0] bits are not set to 0x0, do not apply an external voltage to the VREFA*m* pin.

21 R/F Converter (RFC)

21.1 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. (Note: See the table below.)
  (A thermometer/hygrometer can be easily implemented by connecting a thermistor or a humidity sensor and a few passive elements (resistor and capacitor).)
- Allows measurement (counting) by inputting external clocks.
- Provides an output and continuous oscillation function for monitoring the oscillation frequency.
- Can generate reference oscillation completion, sensor (A and B) oscillation completion, measurement counter overflow error, and time base counter overflow error interrupts.

Figure 21.1.1 shows the RFC configuration.

Table 21.1.1 RFC Channel Configuration of S1C31D41

Item	32-pin package	48-pin package	64-pin package
Number of channels	–	1 channel (Ch.0) **Note**: DC oscillation mode for resistive sensor measurements can only be used.	

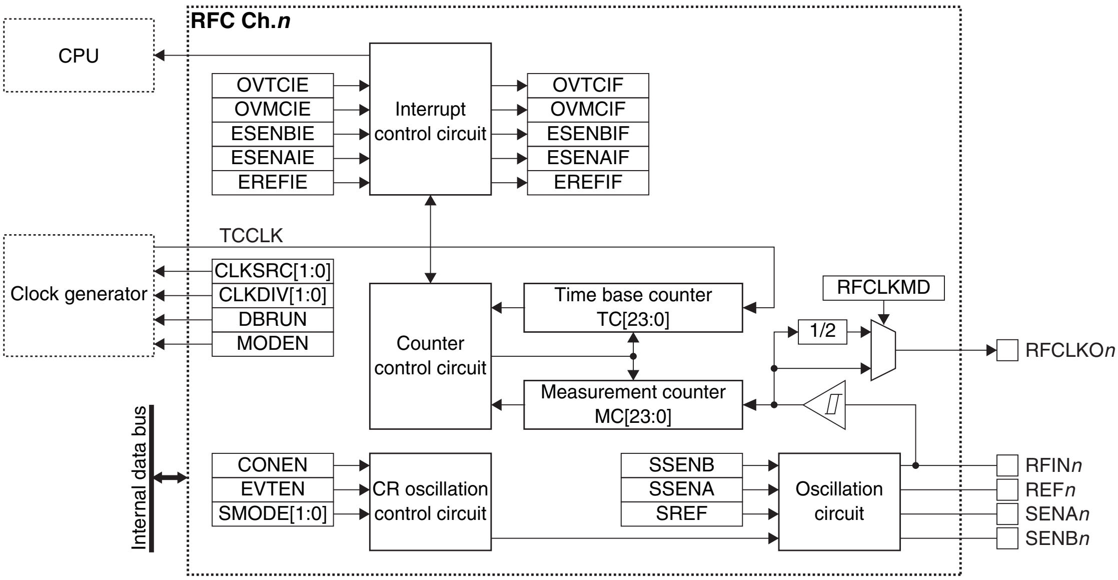


Figure 21.1.1 RFC Configuration

21.2 Input/Output Pins and External Connections

21.2.1 List of Input/Output Pins

Table 21.2.1.1 lists the RFC pins.

Table 21.2.1.1 List of RFC Pins

Pin name	I/O*	Initial status*	Function
SENB*n*	A	Hi-Z	Sensor B oscillation control pin
SENA*n*	A	Hi-Z	Sensor A oscillation control pin
REF*n*	A	Hi-Z	Reference oscillation control pin
RFIN*n*	A	VSS	RFCLK input or oscillation control pin
RFCLKO*n*	O	Hi-Z	RFCLK monitoring output pin RFCLK is output to monitor the oscillation frequency.

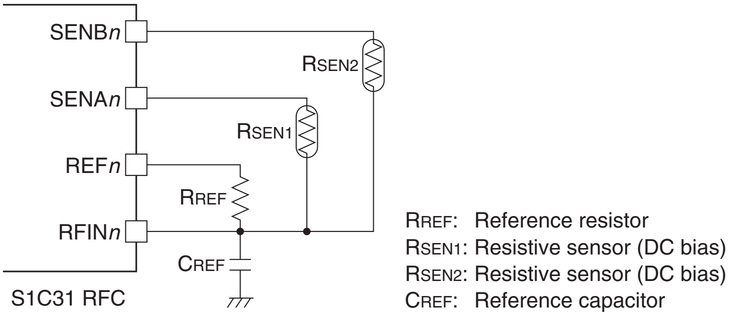
* Indicates the status when the pin is configured for the RFC.

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

Note: The RFIN*n* pin goes to VSS level when the port is switched. Be aware that large current may flow if the pin is biased by an external circuit.

21.2.2 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 21.2.2.1 Connection Example in Resistive Sensor DC Oscillation Mode

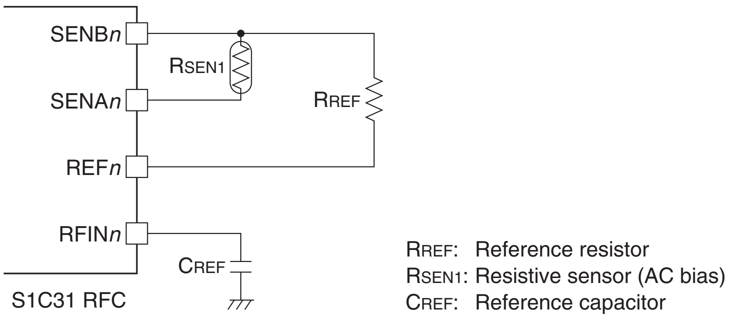
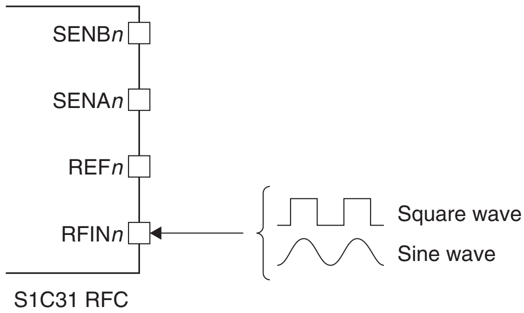


Figure 21.2.2.2 Connection Example in Resistive Sensor AC Oscillation Mode



∗ Leave the unused pins open.

Figure 21.2.2.3 External Clock Input in External Clock Input Mode

21.3 Clock Settings

21.3.1 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.

21.3.2 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.

21.3.3 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.

21.4 Operations

21.4.1 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)

21.4.2 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.

21.4.3 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.

21.4.4 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.

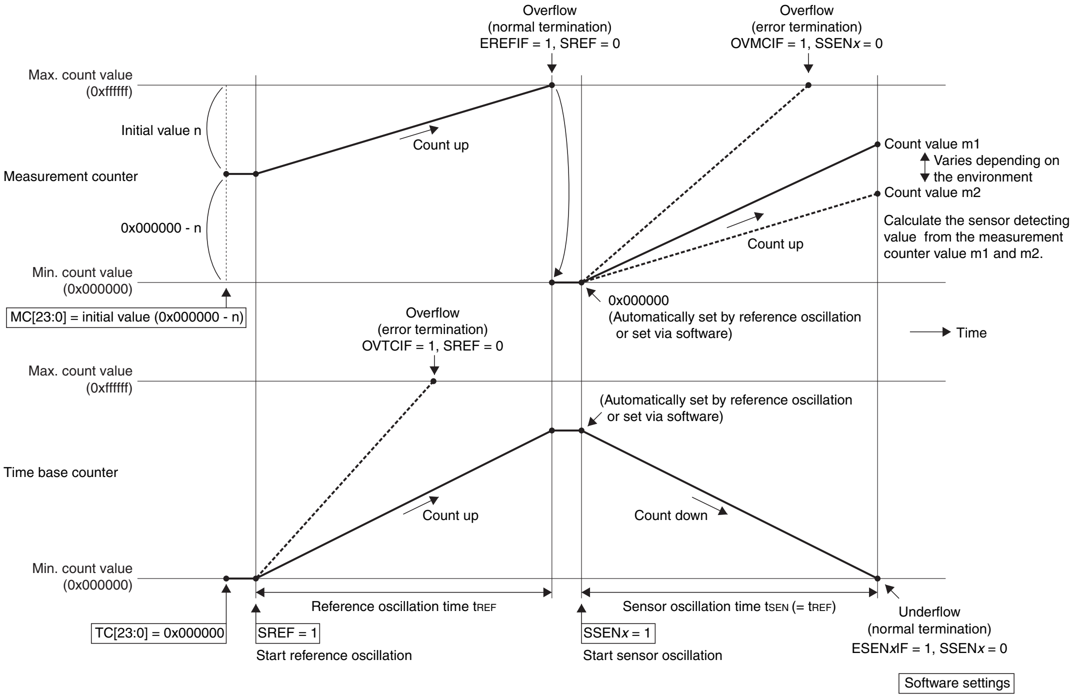


Figure 21.4.4.1 Counter Operations During Reference/Sensor Oscillation

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 21.4.4.1 lists the error factors. (n: measurement counter initial value, m: measurement counter value at the end of sensor oscillation)

Table 21.4.4.1 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

21.4.5 CR Oscillation Frequency Monitoring Function

The CR oscillation clock (RFCLK) generated during converting operation can be output from the RFCLKO*n* pin for monitoring. By setting the RFC_*n*CTL.CONEN bit to 1, the RFC Ch.*n* enters continuous oscillation mode that disables oscillation stop conditions to continue oscillating operations. In this case, set the the RFC_*n*TRG.SREF bit (reference oscillation), the RFC_*n*TRG.SSENA bit (sensor A oscillation), or the RFC_*n*TRG.SSENB bit (sensor B oscillation) to 1 to start oscillation. Set the bit to 0 to stop oscillation. Using this function helps easily measure the CR oscillation clock frequency. Furthermore, setting the RFC_*n*CTL.RFCLKMD bit to 1 changes the output clock to the divided-by-two RFCLK clock.

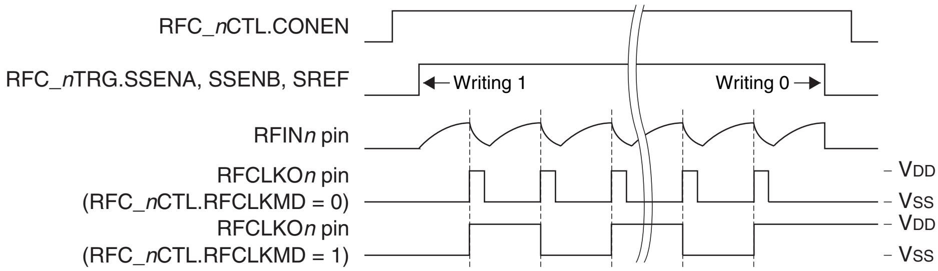


Figure 21.4.5.1 CR Oscillation Clock (RFCLK) Waveform

21.5 Interrupts

The RFC has a function to generate the interrupts shown in Table 21.5.1.

Table 21.5.1 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 CPU 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” chapter.

21.6 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 21.6.1 Clock Source and Division Ratio Settings

RFC_*n*CLK. CLKDIV[1:0] bits	RFC_*n*CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	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 RFC_*n*CLK register settings can be altered only when the RFC_*n*CTL.MODEN bit = 0.

RFC Ch.*n* Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RFC_*n*CTL	15–9	–	0x00	–	R	–
	8	RFCLKMD	0	H0	R/W	
	7	CONEN	0	H0	R/W	
	6	EVTEN	0	H0	R/W	
	5–4	SMODE[1:0]	0x0	H0	R/W	
	3–1	–	0x0	–	R	
	0	MODEN	0	H0	R/W	

Bits 15–9 Reserved

Bit 8 RFCLKMD

This bit sets the RFCLKO*n* pin to output the divided-by-two oscillation clock.

1 (R/W): Divided-by-two clock output

0 (R/W): Oscillation clock output

For more information, refer to “CR Oscillation Frequency Monitoring Function.”

Bit 7 CONEN

This bit disables the automatic CR oscillation stop function to enable continuous oscillation function.

1 (R/W): Enable continuous oscillation

0 (R/W): Disable continuous oscillation

For more information, refer to "CR Oscillation Frequency Monitoring Function."

Bit 6 EVTEN

This bit enables external clock input mode (event counter mode).

1 (R/W): External clock input mode

0 (R/W): Normal mode

For more information, refer to "Operating Modes."

Note: Do not input an external clock before the RFC_*n*CTL.EVTEN bit is set to 1. The RFIN*n* pin is pulled down to Vss level when the port function is switched for the R/F converter.

Bits 5–4 SMODE[1:0]

These bits configure the oscillation mode. For more information, refer to "Operating Modes."

Table 21.6.2 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

22 HW Processor (HWP) and Sound Output (SDAC2)

22.1 Overview

HWP is a functional block having the "Sound Play" and "Memory Check" functions. It can work without any CPU resources by only issuing a command. SDAC2 converts the sound data generated by the HWP into PWM signals and outputs them to an external audio amplifier or an external amplifier circuit composed of discrete parts. The features of the HWP are listed below.

Sound Play function

- EPSON high quality and high compression algorithm (EOV: EPSON Original Sound Format)
  - Sampling Frequency: 15.625 kHz
  - Bitrate: 16/24 kbps
- Multi-channel mixer * See Table 21.1.1.
- Playback speed conversion (channel 0)
  - When using the speed conversion function alone, playback speed can be converted from 75% to 125% in 5% steps.
  - When using in combination with the pitch conversion function, playback speed can be converted from 85% to 115% in 5% steps.
- Playback pitch conversion (channel 0)
  - When using the pitch conversion function alone, playback pitch can be converted from 75% to 125% in 5% steps.
  - When using in combination with the speed conversion function, playback pitch can be converted from 90% to 110% in 5% steps.
- A silent period is inserted between phrases. Gapless play can also be selected. * See Table 21.1.1.
- Sequential playing of multiple sound files
- External QSPI-Flash* supported for storing sound data

Memory Check function

- Embedded RAM check (read/write check algorithm or March-C algorithm)
- Embedded Flash check (checksum or CRC algorithm)
- External QSPI-Flash* check (checksum or CRC algorithm)

* Supports only QSPI Flash memories with XIP (eXecute-In-Place) mode.

Figure 21.1.1 shows the HWP configuration.

Table 22.1.1 HWP Channel Configuration of S1C31D41

Item		S1C31D41
Sound Play function	EOV play	Sampling frequency: 15.625 kHz Bitrate: 16/24 kbps
	Sound channel	2 channels
	Speed conversion	75% to 125% (5% steps) * Channel 0 only
	Pitch conversion	75% to 125% (5% steps) * Channel 0 only
	Simultaneous speed and pitch conversion	Speed: 85% to 115% (5% steps) Pitch: 90% to 110% (5% steps) * Channel 0 only
	Sound output circuit	SDAC2
	Gapless play	Available
	Buzzer output	Available
Memory Check function		Available

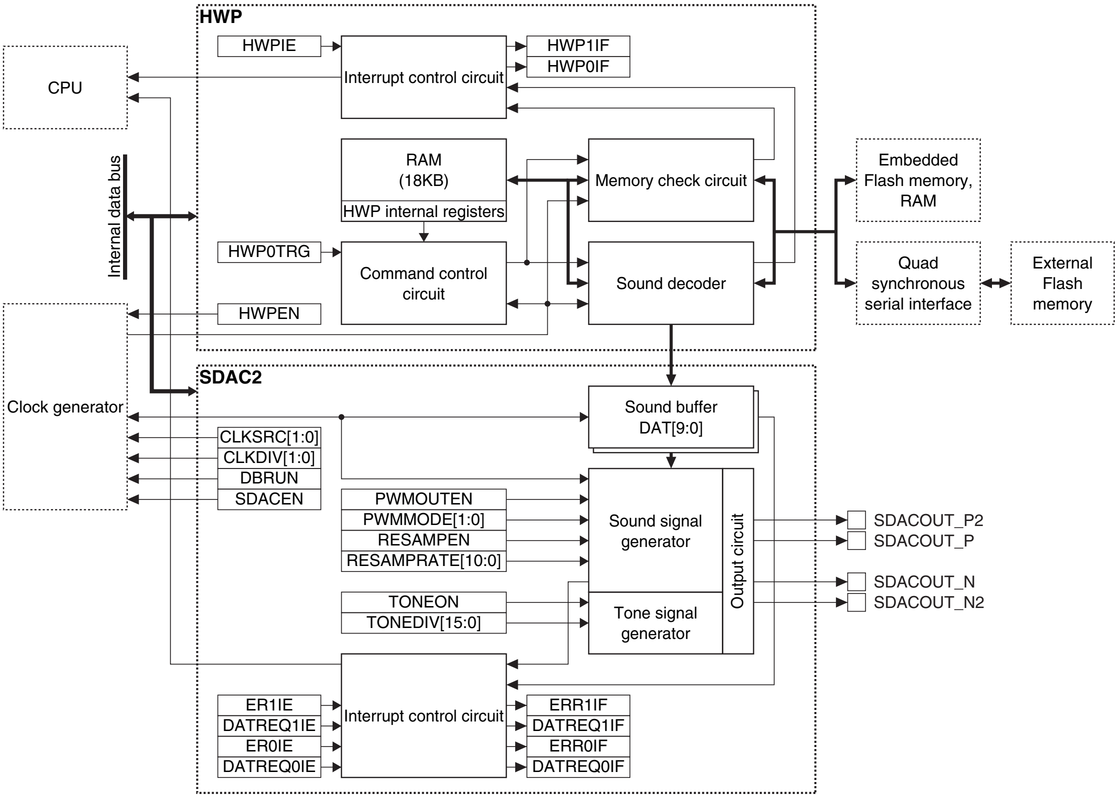


Figure 22.1.1 HWP Configuration

Notes:
- In addition to the control registers in the MCU peripheral circuit area, the HWP has HWP internal registers (address 0x00156700 to address 0x0015674c) that are used for specifying the function and command to be executed and monitoring the operation state. These internal registers are switched between the Sound Play function registers and the Memory Check function registers according to the selected function.
  In this chapter, the register names that begin with HWP or SDAC2 represent a control register in the MCU peripheral circuit area, other names represent a HWP internal register.
- For the specifications of sound data and the setting of the external Flash memory for storing sound data, refer to the application note or the sample software manual.

22.2 Output Pins and External Connections

22.2.1 List of Output Pins

Table 22.2.1.1 lists the SDAC2 output pins.

Table 22.2.1.1 SDAC2 Output Pins

Pin name	I/O	Initial status*	Function
SDACOUT_P	O	O (L)	SDAC2 positive sound signal output (common to two-pin and four-pin output mode)
SDACOUT_N	O	O (L)	SDAC2 negative sound signal output (common to two-pin and four-pin output mode)
SDACOUT_P2	O	O (L)	SDAC2 positive sound signal 2 output (dedicated for four-pin output mode)
SDACOUT_N2	O	O (L)	SDAC2 negative sound signal 2 output (dedicated for four-pin output mode)

* Indicates the status when the pin is configured for SDAC2.

The SDAC2 output pins are shared with general-purpose I/O ports. These pins can be used in either two-pin output mode or four-pin output mode according to the external circuit configuration and are initially set to two-pin output mode (two-pin output mode combination 1 in the table below). When using them in four-pin output mode, switch the function of the ports for four-pin output dedicated pins to SDAC2 outputs. If the Sound Play 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.

Note: If the audio amplifier has an enable signal input, use a general-purpose I/O port pin to output the external amplifier enable signal.

Table 22.2.1.2 Output Mode Selection

Pin	Two-pin output mode		Four-pin output mode	
	Combination 1	Combination 2	Combination 1	Combination 2
SDACOUT_P/P50	SDACOUT_P *1	–	–	SDACOUT_P *2
SDACOUT_N/P51	SDACOUT_N *1	–	–	SDACOUT_N *2
P03/SDACOUT_P2/UPMUX	–	–	SDACOUT_P2	SDACOUT_P2
P04/SDACOUT_P/UPMUX	–	SDACOUT_P *2	SDACOUT_P *1	–
P05/SDACOUT_N/UPMUX	–	SDACOUT_N *2	SDACOUT_N *1	–
P06/SDACOUT_N2/UPMUX	–	–	SDACOUT_N2	SDACOUT_N2

Note: The SDACOUT_P and SDACOUT_N outputs are each assigned to two pins. Basically, use combination 1 (*1) and another (*2) should be configured for a general-purpose input/output pin or a UPMUX pin.

22.2.2 External Connections

Figures 22.2.2.1 and 22.2.2.2 show external circuit examples to input the SDAC2 sound signals to an external audio amplifier. The circuit configuration and component values should be modified according to the specifications of the audio amplifier to be used.

Single mode

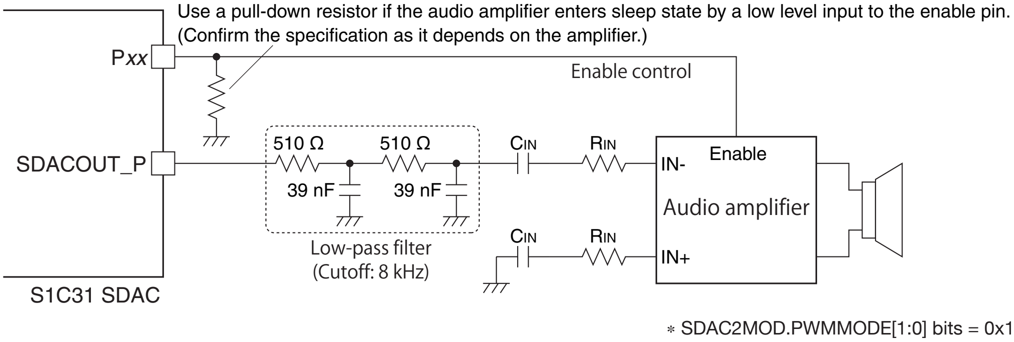


Figure 22.2.2.1 Single Mode Connection Example

Differential mode

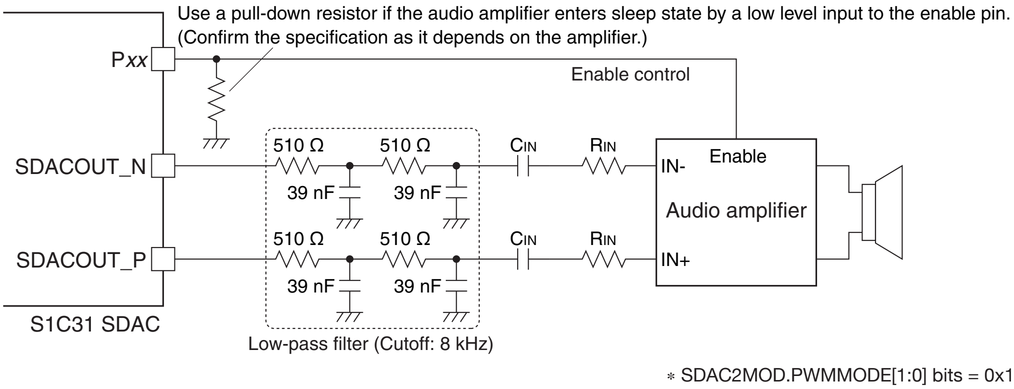


Figure 22.2.2.2 Differential Mode Connection Example

Figures 22.2.2.3 and 22.2.2.4 show examples to connect the SDAC2 sound signals to an external amplifier circuit composed of discrete parts. For more information on the amplifier circuit configuration, refer to an evaluation manual or application note.

Two-pin output mode

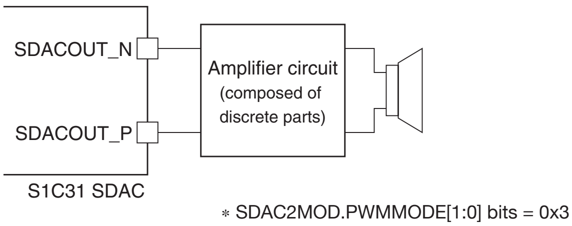


Figure 22.2.2.3 Connection Example of SDAC2 with External Discrete Amplifier Circuit (Two-pin Output Mode)

Four-pin output mode

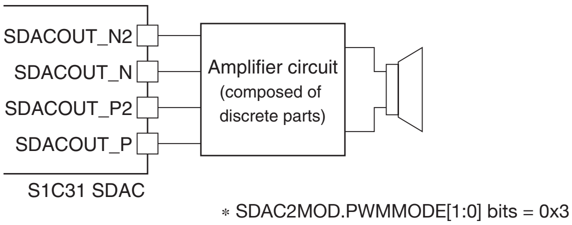


Figure 22.2.2.4 Connection Example of SDAC2 with External Discrete Amplifier Circuit (Four-pin Output Mode)

22.3 Clock Settings

22.3.1 HWP Operating Clock

The HWP and the SDAC2 used for the Sound Play function operate with the SYSCLK (system clock) supplied from the clock generator. The HWP operating clock should be controlled as in the procedure shown below.

When executing the Sound Play function

1. Configure SYSCLK in the clock generator (refer to “Clock Generator” in the “Power Supply, Reset, and Clocks” chapter).
   - SYSCLK source = OSC3
   - OSC3 oscillation frequency = 16 MHz
2. Set the following SDAC2CLK register bits (set in S1C31D41 regardless of the sound output destination):
   - Set the SDAC2CLK.CLKSRC[1:0] bits to 0x02. (Clock source = OSC3)
   - Set the SDAC2CLK.CLKDIV[1:0] bits to 0x0. (Clock division ratio = 1/1)

When executing the Memory Check function

1. Configure SYSCLK for operating the Memory Check function in the clock generator (refer to “Clock Generator” in the “Power Supply, Reset, and Clocks” chapter).

The Memory Check function allows the HWP to use SYSCLK at any arbitrary frequency.

22.3.2 Clock Supply in SLEEP Mode

The HWP and SDAC2 stop operating in SLEEP mode, as the SYSCLK stops. Do not put the IC into SLEEP mode while the HWP is operating. It is possible to put the IC into HALT mode while the HWP is operating.

22.3.3 Clock Supply in DEBUG Mode

The sound play and Memory Check functions can operate even in DEBUG mode, as SYSCLK is supplied.
The SYSCLK supply to the SDAC2 during DEBUG mode can be controlled using the SDAC2CLK.DBRUN bit. The SDAC2CLK.DBRUN bit must be set to 1 when using the Sound Play function during DEBUG mode. Be aware that the sound cannot be output normally when SDAC2CLK.DBRUN bit = 0.

22.4 Operations

22.4.1 Sound Play Function

Initialization

When using the Sound Play function, initialize the SDAC2 and HWP in this order as shown below.

Initializing SDAC2

1. If the SDAC2 output pins are configured for general-purpose ports, assign the SDAC2 output function to the ports. (Refer to the "I/O Ports" chapter.)

   To select two-pin output mode
   - P5MODSEL register = (P5MODSEL register / 0x0003)
   - P0MODSEL register = (P0MODSEL register & 0x0087)

   To select four-pin output mode
   - P0MODSEL register = (P0MODSEL register / 0x0078)
   - P5MODSEL register = (P5MODSEL register & 0x00fc)

2. Configure the SDAC2 operating clock. (Refer to Section 21.3, "Clock Settings.")
   - Set the SDAC2CLK.CLKSRC[1:0] bits to 0x02. (Clock source = OSC3)
   - Set the SDAC2CLK.CLKDIV[1:0] bits to 0x0. (Clock division ratio = 1/1)
   - Set the SDAC2CLK.DBRUN bit to 1. (Enable clock supply in DEBUG mode)

3. Set the SDAC2 control register.
   - Set the SDAC2CTL.SDACEN bit. (Enable SDAC2)
   - Set the SDAC2CTL.RESAMPRATE[10:0] bits to 0x400. (Initialize audio sampling frequency)
   - Set the SDAC2CTL.RESAMPEN bit to 1. (Enable resampler)
   - Set the SDAC2CTL.TONEON bit to 0. (Disable square-wave tone generator)

4. Clear SDAC2 sound data registers.
   - Set the SDAC2_0DAT.DAT[9:0] bits to 0x000. (Clear Ch.0 sound data register)
   - Set the SDAC2_1DAT.DAT[9:0] bits to 0x000. (Clear Ch.1 sound data register)

5. Disable SDAC2 interrupts to occur.
   - Set the SDAC2INTE register to 0x0000. (Disable interrupts)
   - Write 0x000f to the SDAC2INTF register. (Clear interrupt flags)

6. Set the SDAC2 operating mode.

   To use external audio amplifier
   - Set the SDAC2MOD.PWMMODE[1:0] bits to 0x1. (Normal mode)

   To use external amplifier circuit composed of discrete parts
   - Set the SDAC2MOD.PWMMODE[1:0] bits to 0x3. (CPLM mode 2)

Initializing HWP (Sound Play function)

7. Set the HWPCTL.HWPEN bit to 0. (Disable HWP)

8. Set the following HWP internal register bits (Sound Play function register bits):
   - Set the FUNCTION.ID[7:0] bits to 0x01. (Select Sound Play function)
   - INTMASK.TO_MUTE bit (Set mute interrupt mask)
   - INTMASK.TO_PAUSE bit (Set pause interrupt mask)
   - INTMASK.TO_PLAY bit (Set playback start interrupt mask)
   - INTMASK.TO_IDLE bit (Set idle state interrupt mask)
   - ROMADDR.ADDRESS[31:0] bits (Set sound ROM data start address)
   - ROMSIZE.SIZE[31:0] bits (Set sound ROM data size)
   - KEYCODE.KEYCODE[31:0] bits (Set key code)

9. Perform the following settings when using the HWP interrupt:
   - Set the HWP interrupt level (refer to the documents introduced in Section 3.4, such as "Cortex®-M0+ Devices Generic User Guide").
   - Write 0 to all the interrupt flags in the HWPINTF register. (Clear interrupt flags)

   Note: Be aware that the write value to clear flags is different from other peripheral circuits.

   - Set the HWPINTE.HWPIE bit to 1. (Enable interrupts)

Checking HWP operation (Sound Play function)

10. Set the SDAC2MOD.PWMOUTEN bit to 1. (Enable sound output)
11. Enable the external amplifier using a general-purpose output port (if necessary).

    * Set a wait time according to the amplifier specifications after being enabled.
12. Set the HWPCTL.HWPEN bit to 1. (Enable HWP)
13. Wait until the HWPINTF.HWP0IF bit is set to 1 and the STATE_*n*.STATE[15:0] bits are set to 0x0001 (sp_state_idle = Sound Play function idle state).
    Initialize the SDAC2 and HWP in this order again if the HWPINTF.HWP1IF bit = 1.

Sound play state transition

Figure 22.4.1.1 shows the sound play state transition diagram.

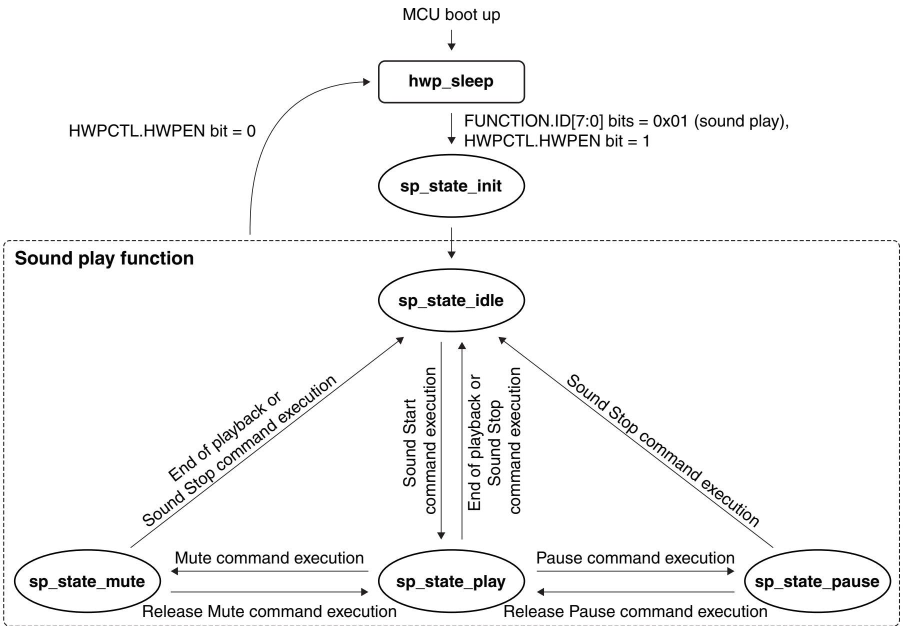


Figure 22.4.1.1 Sound Play State Transition Diagram

As shown in Figure 22.4.1.1, there are six operating states in the Sound Play function.

1) hwp_sleep

After the MCU boots up, the HWP enters this state (HWPCTL.HWPEN bit = 0). In this state, the clock supply to the HWP stops. By setting the HWPCTL.HWPEN bit to 1 after configuring the Sound Play function registers as shown in "Initialization" above, the HWP transits to sp_state_init state.

2) sp_state_init

After the HWPCTL.HWPEN bit is set to 1, the HWP enters this state and initializes the internal circuit according to the settings of the Sound Play function registers. Upon completion of the initial processing, the HWP transits to sp_state_idle state.

3) sp_state_idle

This is a standby state in which the Sound Play function stops playback output. This state allows issuance of the Sound Start command. After the Sound Start command is issued, the HWP transits to sp_state_play state to start playback output.

4) sp_state_play

This is the state in which the HWP is performing playback output. This state allows issuance of the Sound Stop, Pause, or Mute command. When the sound data ends or the Sound Stop command is issued, the HWP stops playback output and returns to sp_state_idle state. When the Pause command is issued, the HWP transits to sp_state_pause state to pause playback output. When the Mute command is issued, the HWP transits to sp_state_mute state to mute the sound.

5) sp_state_pause

This is the state in which the playback output is paused. This state allows issuance of the Release Pause or Sound Stop command. When the Release Pause command is issued, the HWP transits to sp_state_play state to resume playback output. When the Sound Stop command is issued, the HWP terminates playback output and returns to sp_state_idle state.

6) sp_state_mute

This is the state in which the playback output is being continued with the sound muted. This state allows issuance of the Release Mute or Sound Stop command. When the Release Mute command is issued, the HWP transits to sp_state_play state to restore the volume. When the sound data ends or the Sound Stop command is issued, the HWP terminates playback output and returns to sp_state_idle state.

The current Ch.*n* operating state can be monitored by reading the STATE_*n*.STATE[15:0] bits (except hwp_sleep). Furthermore, an interrupt can be generated when a state transition to the designated state occurs.

Sound play commands

Table 22.4.1.1 lists the Sound Play function commands.

Table 22.4.1.1 List of Sound Play Commands

Command	Function	Issuable state	Transit destination state
Sound Start	Start playback output	sp_state_idle	sp_state_play
Sound Stop Immediately	Stop playback output immediately	sp_state_play, sp_state pause, sp_state_mute	sp_state_idle
Sound Stop after Current Phrase	Stop playback output after ending current phrase	sp_state_play, sp_state pause, sp_state_mute	sp_state_idle
Pause Immediately	Pause playback output immediately	sp_state_play	sp_state_pause
Pause after Current Phrase	Pause playback output after ending current phrase	sp_state_play	sp_state_pause
Release Pause	Release pause state	sp_state_pause	sp_state_play
Mute Immediately	Mute playback output immediately	sp_state_play	sp_state_mute
Mute after Current Phrase	Mute playback output after ending current phrase	sp_state_play	sp_state_mute
Release Mute	Release mute state	sp_state_mute	sp_state_play

Each sound play command can be issued in the specific states. Follow the procedure below to issue a command.

1. Confirm that the STATE_*n*.STATE[15:0] bits = issuable state.
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Configure the Sound Play function registers required to execute the command (if necessary).
4. Set the COMMAND_*n*.COMMAND[7:0] bits. (Select command)
5. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
6. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
7. Confirm that the STATE_*n*.STATE[15:0] bits = transit destination state (if necessary).

Playback start/stop

Single channel playback output start procedure

The following shows a Ch.*n* playback output start procedure:

1. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0001 (sp_state_idle).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)

3. Configure the following sound play register bits:
   - Set the COMMAND_*n*.COMMAND[7:0] bits to 0x01. (Select Sound Start command)
   - Set the COMMAND_*n*.OPTION[7:0] bits. (Select gapless play option)
   - SENTENCE_*n*.SENTENCE_NO[15:0] bits (Specify sentence number)
   - VOLUME_*n*.VOLUME[15:0] bits (Specify volume level)
   - REPEAT_*n*.REPEAT[15:0] bits (Specify repeat count)
   - SPEED_0.SPEED[15:0] bits (Specify playback speed, Ch.0 only)
   - PITCH_0.PITCH[15:0] bits (Specify playback pitch, Ch.0 only)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   The HWP starts sound data output of the specified sentence number from this point.
6. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0002 (sp_state_play) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)
   :
   Playback is in progress.
   :
8. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
9. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0001 (sp_state_idle) as necessary.

When the sound data ends, playback output is automatically terminated and the Sound Play function transits to sp_state_idle state.

Note: The volume level can be adjusted by rewriting the VOLUME_*n*.VOLUME[15:0] bits even while playback is in progress. The playback speed and pitch cannot be changed while playback is in progress.

2-channel mix output start procedure

The HWP can output sound by mixing two channels, for instance, Ch.0 is used for voice and Ch.1 is used for BGM. To do this, the channels should be controlled continuously to start playback as shown below.

Ch.1 (BGM) output start procedure

1. Confirm that the STATE_1.STATE[15:0] bits = 0x0001 (sp_state_idle).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Configure the following sound play register bits:
   - Set the COMMAND_1.COMMAND[7:0] bits to 0x01. (Select Sound Start command)
   - SENTENCE_1.SENTENCE_NO[15:0] bits (Specify sentence number)
   - VOLUME_1.VOLUME[15:0] bits (Specify volume level)
   - REPEAT_1.REPEAT[15:0] bits (Specify repeat count)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   The HWP starts BGM data output of the specified sentence number from this point.
6. Confirm that the STATE_1.STATE[15:0] bits = 0x0002 (sp_state_play) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

Ch.0 (voice) output start procedure

8. Confirm that the STATE_0.STATE[15:0] bits = 0x0001 (sp_state_idle).
9. Confirm that the STATUS.READY bit = 1. (Command acceptable)
10. Configure the following sound play register bits:
    - Set the COMMAND_0.COMMAND[7:0] bits to 0x01. (Select Sound Start command)
    - SENTENCE_0.SENTENCE_NO[15:0] bits (Specify sentence number)
    - VOLUME_0.VOLUME[15:0] bits (Specify volume level)
    - REPEAT_0.REPEAT[15:0] bits (Specify repeat count)
    - SPEED_0.SPEED[15:0] bits (Specify playback speed)
    - PITCH_0.PITCH[15:0] bits (Specify playback pitch)

11. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
12. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
    The HWP starts voice data output of the specified sentence number from this point.
13. Confirm that the STATE_0.STATE[15:0] bits = 0x0002 (sp_state_play) as necessary.
14. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

    :

    Playback is in progress. (The SDAC2 outputs the mixed sound of Ch.0 and Ch.1.)

    :

Confirming end of Ch.0 (voice) output

15. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
16. Confirm that the STATE_0.STATE[15:0] bits = 0x0001 (sp_state_idle) as necessary.
17. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

Confirming end of Ch.1 (BGM) output

18. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
19. Confirm that the STATE_1.STATE[15:0] bits = 0x0001 (sp_state_idle) as necessary.
20. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

Mute

Mute control

The sound can be muted during playback with the procedure shown below.

1. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0002 (sp_state_play).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND_*n*.COMMAND[7:0] bits to 0x07 or 0x08.* (Select Mute command)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   The HWP mutes the sound from this point.
6. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0004 (sp_state_mute) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

The above operation mutes the sound while playback output continues.

* Two mute commands are available. Setting the COMMAND_*n*.COMMAND[7:0] bits to 0x07 selects the Mute Immediately command; setting to 0x08 selects the Mute after Current Phrase command.

Mute Immediately command

When this command is issued by the trigger bit, the sound is muted immediately. At this time, a smoothing (fade-out) process for the playback output signal is carried out to suppress the occurrence of noise.

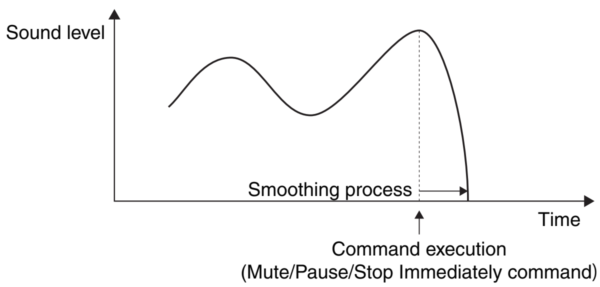


Figure 22.4.1.2 Smoothing Process when Playback Output is Suspended

Mute after Current Phrase command

The sound is muted after ending the phrase that is being output when the command is issued.

Sentence example: "The temperature is set at / 41 degrees."

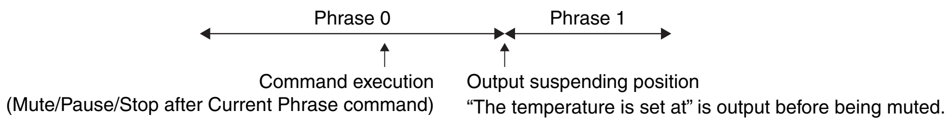


Figure 22.4.1.3 Example of Waiting for End of Phrase

Mute release

The mute state can be released with the procedure shown below.

1. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0004 (sp_state_mute).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND_*n*.COMMAND[7:0] bits to 0x09. (Select Release Mute command)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   From this point, the volume returns to the level it was before being muted.
6. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0002 (sp_state_play) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

When the volume returns back to the original level, a smoothing (fade-in) process for the playback output signal is carried out to suppress the occurrence of noise due to a sudden rise of the signal.

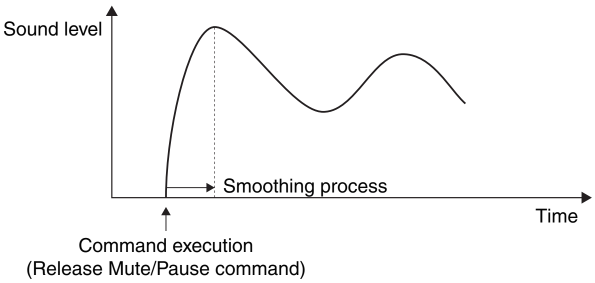


Figure 22.4.1.4 Smoothing Process when Playback Output is Resumed

When sound data ends in mute state

End of sound data automatically stops the muted playback output and the Sound Play function transits to sp_state_idle state. If the HWPINTF.HWP0IF bit is set to 1 (occurrence of state transition) before releasing the mute state, read the STATE_*n*.STATE[15:0] bits and check to see if they are set to 0x0001 (sp_state_idle).

Pause

Pause control

During playback, it can be paused with the procedure shown below.

1. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0002 (sp_state_play).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND_*n*.COMMAND[7:0] bits to 0x04 or 0x05.* (Select Pause command)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   The HWP pauses the playback output from this point.
6. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0003 (sp_state_pause) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

* Two pause commands are available. Setting the COMMAND_*n*.COMMAND[7:0] bits to 0x04 selects the Pause Immediately command; setting to 0x05 selects the Pause after Current Phrase command.

Pause Immediately command

When this command is issued by the trigger bit, the playback output is paused immediately. At this time, a smoothing (fade-out) process for the playback output signal is carried out to suppress the occurrence of noise. (See Figure 22.4.1.2.)

Pause after Current Phrase command

The playback output is paused after ending the phrase that is being output when the command is issued. (See Figure 22.4.1.3.)

Pause release

The pause state can be released with the procedure shown below.

1. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0003 (sp_state_pause).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND_*n*.COMMAND[7:0] bits to 0x06. (Select Release Pause command)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)

   The HWP resumes playback output from this point.
6. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0002 (sp_state_play) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

When the playback output is resumed, a smoothing (fade-in) process for the playback output signal is carried out to suppress the occurrence of noise due to a sudden rise of the signal. (See Figure 22.4.1.4.)

Terminating playback

In the playback state (sp_state_play), pause state (sp_state_pause), and mute state (sp_state_mute), the playback can be terminated to return the state to the idle state (sp_state_idle) with the procedure shown below.

1. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0002 (sp_state_play), 0x0003 (sp_state_pause), or 0x0004 (sp_state_mute).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND_*n*.COMMAND[7:0] bits to 0x02 or 0x03.* (Select Sound Stop command)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)

   The HWP enters the idle state at this point.
6. Confirm that the STATE_*n*.STATE[15:0] bits = 0x0001 (sp_state_idle) as necessary.
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

* Two playback terminate commands are available. Setting the COMMAND_*n*.COMMAND[7:0] bits to 0x02 selects the Sound Stop Immediately command; setting to 0x03 selects the Sound Stop after Current Phrase command.

Sound Stop Immediately command

When this command is issued by the trigger bit, the playback output is terminated immediately. At this time, a smoothing (fade-out) process for the playback output signal is carried out to suppress the occurrence of noise. (See Figure 22.4.1.2.)

Sound Stop after Current Phrase command

The playback output is terminated after ending the phrase that is being output when the command is issued. (See Figure 22.4.1.3.)

In the pause or mute state, the playback is terminated by releasing the pause or mute state immediately regardless of which of the Sound Stop commands is issued.

Sound play error

When an error occurs during processing of the Sound Play function, the HWPINTF.HWP1IF bit is set to 1 (an interrupt can be generated). The error contents can be confirmed by reading the ERROR.ERROR[15:0] bits. As shown in Table 22.4.1.2, the ERROR.ERROR*x* bit corresponding to the error that has occurred is set to 1.

Table 22.4.1.2 List of Sound Play Errors

ERROR.ERROR[15:0] bits	Error	Meaning
0000 0000 0000 0000	error_no_error	No error has occurred.
Non-fatal error		
xxxx xxxx xxxx xxx1 (bit 0)	error_ch0_command	A command that is undefined or is ineffective in the current state has been specified in Ch.0.
xxxx xxxx xxxx xx1x (bit 1)	error_ch1_command	A command that is undefined or is ineffective in the current state has been specified in Ch.1.
xxxx xxxx xxxx x1xx (bit 2)	error_ch0_sentence_no	An invalid sentence number has been specified in Ch.0.
xxxx xxxx xxxx 1xxx (bit 3)	error_ch1_sentence_no	An invalid sentence number has been specified in Ch.1.
xxxx xxxx 1xxx xxxx (bit 7)	error_SDAC2_overflow	An overflow has occurred in the SDAC2 output signal.
Fatal error		
xxxx xxx1 xxxx xxxx (bit 8)	error_ch0_decode	Invalid sound data has been read in Ch.0.
xxxx xx1x xxxx xxxx (bit 9)	error_ch1_decode	Invalid sound data has been read in Ch.1.
xxx1 xxxx xxxx xxxx (bit 12)	error_rom_data_mount	The sound data ROM cannot be accessed.
x1xx xxxx xxxx xxxx (bit 14)	error_function_id	An undefined function ID has been specified.
1xxx xxxx xxxx xxxx (bit 15)	error_others	Another error has occurred.

When a non-fatal error has occurred, reissue a valid command.
When a fatal error has occurred, remove the cause of error and redo the processing from initialization.

Tone signal output

The SDAC2 provides a tone generation function to output a tone signal (square wave) with the frequency specified from the SDAC2 pins.
Before using this function, make sure that both Ch.0 and Ch.1 have terminated normal playback.
The following shows a procedure to start/stop tone signal output:

Starting output

1. Check if the STATE_*n*.STATE[15:0] bits = 0x0001(sp_state_idle).
2. Set the SDAC2TONE.TONEDIV[15:0] bits. (Set tone signal (square wave) frequency)
3. Set the SDAC2CTL.TONEON bit to 1. (Start tone signal (square wave) output)

   :

   Output is in progress.

   :

Terminating output

4. Set the SDAC2CTL.TONEON bit to 0. (Stop tone signal (square wave) output)

22.4.2 Memory Check Function

Initialization

Before using the Memory Check function, initialize the HWP as shown below.

1. Configure the HWP operating clock as necessary. (Refer to “Clock Settings.”)
2. Set the HWPCTL.HWPEN bit to 0. (Disable HWP)
3. Set the following HWP internal register bits (Memory Check function register bits):
   - Set the FUNCTION.ID[7:0] bits to 0x03. (Select Memory Check function)
   - INTMASK.TO_PROCESSING bit (Set check start interrupt mask)
   - INTMASK.TO_IDLE bit (Set check completed/idle state interrupt mask)
4. Set the following bits when using the interrupt:
   - Write 1 to the interrupt flags in the HWPINTF register. (Clear interrupt flags)
   - Set the HWPINTE.HWPIE bit to 1. (Enable interrupts)
5. Set the HWPCTL.HWPEN bit to 1. (Enable HWP)
6. Wait until the HWPINTF.HWP0IF bit is set to 1 and the STATE.STATE[15:0] bits are set to 0x0001 (mc_state_idle = Memory Check function idle state).

   Initialize the HWP again if the HWPINTF.HWP1IF bit = 1.

Once the Memory Check function register bits have been set in Step 3, it is not necessary to set again until they need to be altered. When altering these register bits, perform the above processing from Step 2.

Memory check state transition

Figure 22.4.2.1 shows the memory check state transition diagram.

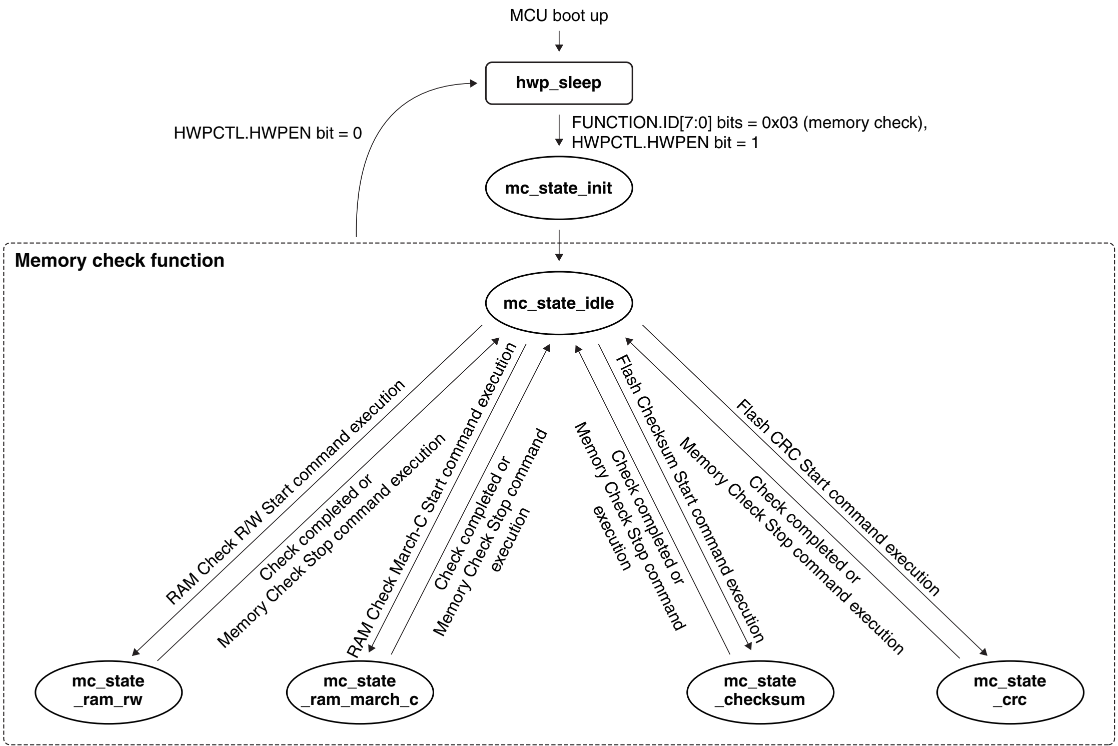


Figure 22.4.2.1 Memory check State Transition Diagram

As shown in the figure above, there are seven operating states in the Memory Check function.

1) hwp_sleep

After the MCU boots up, the HWP enters this state (HWPCTL.HWPEN bit = 0). In this state, the clock supply to the HWP stops. By setting the HWPCTL.HWPEN bit to 1 after configuring the Memory Check function registers as shown in "Initialization" above, the HWP transits to mc_state_init state.

2) mc_state_init

After the HWPCTL.HWPEN bit is set to 1, the HWP enters this state and initializes the internal circuit according to the settings of the Memory Check function registers. Upon completion of the initial processing, the HWP transits to mc_state_idle state.

3) mc_state_idle

This is the state in which the Memory Check function is idle. This state allows issuance of a memory check command. After a memory check command is issued, the HWP transits to a state from 4) to 7) to start memory check.

4) mc_state_ram_rw

This is the state in which the HWP is performing the RAM read/write check. When the RAM Check R/W Start command is issued in mc_state_idle state, the HWP transits to this state. When the check has completed or the Memory Check Stop command is issued, the HWP returns to mc_state_idle state.

5) mc_state_ram_march_c

This is the state in which the HWP is performing the RAM check using the March-C algorithm. When the RAM Check March-C Start command is issued in mc_state_idle state, the HWP transits to this state. When the check has completed or the Memory Check Stop command is issued, the HWP returns to mc_state_idle state.

6) mc_state_checksum

This is the state in which the HWP is performing the Flash memory check that calculates the checksum. When the Flash Checksum Start command is issued in mc_state_idle state, the HWP transits to this state. When the check has completed or the Memory Check Stop command is issued, the HWP returns to mc_state_idle state.

7) mc_state_crc

This is the state in which the HWP is performing the Flash memory check that calculates the CRC. When the Flash CRC Start command is issued in mc_state_idle state, the HWP transits to this state. When the check has completed or the Memory Check Stop command is issued, the HWP returns to mc_state_idle state.

The current operating state can be monitored by reading the STATE.STATE[15:0] bits (except hwp_sleep). Furthermore, an interrupt can be generated when a state transition to the designated state occurs.

Memory check commands

Table 22.4.2.1 lists the Memory Check function commands.

Table 22.4.2.1 List of Memory Check Commands

Command	Function	Issuable state	Transit destination state
RAM Check R/W Start	Start RAM check (read/write)	mc_state_idle	mc_state_ram_rw
RAM Check March-C Start	Start RAM check (March-C)	mc_state_idle	mc_state_ram_march_c
Flash Checksum Start	Start Flash check (checksum)	mc_state_idle	mc_state_checksum
Flash CRC Start	Start Flash check (CRC)	mc_state_idle	mc_state_crc
Memory Check Stop	Stop memory check	mc_state_ram_rw, mc_state_ram_march_c, mc_state_checksum, mc_state_crc	mc_state_idle

The memory check start command can be issued only in mc_state_idle state.
Follow the procedure below to issue a command.

1. Confirm that the STATE.STATE[15:0] bits = 0x0001 (mc_state_idle).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND.COMMAND[7:0] bits. (Select command)
4. Set the MEMADDR.ADDRESS[31:0] bits. (Specify check start address)
5. Set the MEMSIZE.SIZE[31:0] bits. (Specify check size (byte))

6. Set the INITVALUE.INITVALUE[31:0] bits to 0x00000000. (Specify Flash check initial value)
   * It is not necessary in the RAM check.
7. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
8. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
9. Confirm that the STATE.STATE[15:0] bits = transit destination state (if necessary).

RAM check

The following shows the procedure to execute a RAM check:

1. Confirm that the STATE.STATE[15:0] bits = 0x0001 (mc_state_idle).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND.COMMAND[7:0] bits or 0x02 or 0x03.* (Select command)
4. Set the MEMADDR.ADDRESS[31:0] bits. (Specify check start address)
5. Set the MEMSIZE.SIZE[31:0] bits. (Specify check size (byte))
6. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
7. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   The HWP starts the memory check from this point.
8. Confirm that the STATE.STATE[15:0] bits = 0x0002 (mc_state_ram_rw) or 0x0003 (mc_state_ram_march_c) as necessary.*
9. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)
   :
   The memory check is in progress.
   :
10. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
    The memory check is completed at this point.
11. Confirm that the STATE.STATE[15:0] bits = 0x0001 (mc_state_idle).
12. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)
13. Read the STATUS.PROCESSING[1:0] bits. (Confirm check result)

When the STATUS.PROCESSING[1:0] bits = 0x2, the check has completed without an error. So the processing can be terminated.
If the STATUS.PROCESSING[1:0] bits = 0x3, an error has occurred. In this case, confirm the address where the error has occurred as follows:

14. Read the RESULT.RESULT[31:0] bits. (Confirm error address)
    These bits hold the address where an error has occurred first.

* Two RAM check commands are available. Setting the COMMAND.COMMAND[7:0] bits to 0x02 selects the RAM Check R/W Start command; setting to 0x03 selects the RAM Check March-C Start command.

RAM Check R/W Start command

When this command is issued by the trigger bit, the HWP transits to mc_state_ram_rw state to execute the RAM read/write check. In this check, the HWP performs read-after-write verification for all addresses twice: first it writes 0x55aa, next 0xaa55.

RAM Check March-C Start command

When this command is issued by the trigger bit, the HWP transits to mc_state_ram_march_c state to execute the RAM marching test (March-C algorithm).

Note: When an error occurs during RAM check, the check is terminated at the address where the error has occurred.

Flash check

The following shows the procedure to execute a Flash check:

1. Confirm that the STATE.STATE[15:0] bits = 0x0001 (mc_state_idle).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND.COMMAND[7:0] bits to 0x04 or 0x05.* (Select command)
4. Set the MEMADDR.ADDRESS[31:0] bits. (Specify check start address)
5. Set the MEMSIZE.SIZE[31:0] bits. (Specify check size (byte))
6 Set the INITVALUE.INITVALUE[31:0] bits to 0x00000000. (Specify Flash check initial value)
7. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
8. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
   The HWP starts the memory check from this point.
9. Confirm that the STATE.STATE[15:0] bits = 0x0004 (mc_state_checksum) or 0x0005 (mc_state_crc) as necessary.*
10. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)
    :
    The memory check is in progress.
    :
11. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
    The memory check is completed at this point.
12. Confirm that the STATE.STATE[15:0] bits = 0x0001 (mc_state_idle).
13. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)
14. Confirm that the STATUS.PROCESSING[1:0] bits = 0x2 (check completed).
15. Read the RESULT.RESULT[31:0] bits. (Confirm check result)
    These bits hold the checksum or CRC calculation result.
16. Compare the read calculation result with the original value.

* Two Flash check commands are available. Setting the COMMAND.COMMAND[7:0] bits to 0x04 selects the Flash Checksum Start command; setting to 0x05 selects the Flash CRC Start command.

Flash Checksum Start command

When this command is issued by the trigger bit, the HWP transits to mc_state_checksum state to calculate the checksum from the specified Flash data.

Flash CRC Start command

When this command is issued by the trigger bit, the HWP transits to mc_state_crc state to calculate the CRC from the specified Flash data.

Note: The HWP uses memory mapped access mode (refer to the “Quad Synchronous Serial Interface” chapter) for the external QSPI-Flash check. Therefore, external Flash memories that do not support XIP (eXecute-In-Place) cannot be checked.

Terminating memory check

The following shows the procedure to terminate the memory check being executed.

1. Confirm that the STATE.STATE[15:0] bits = 0x0002 to 0x0005 (during memory check).
2. Confirm that the STATUS.READY bit = 1. (Command acceptable)
3. Set the COMMAND.COMMAND[7:0] bits to 0xff. (Select Memory Check Stop command)
4. Write 1 to the HWPCMDTRG.HWP0TRG bit. (Trigger to issue command)
5. Wait until the HWPINTF.HWP0IF bit is set to 1 (interrupt). (Occurrence of state transition)
6. Confirm that the STATE.STATE[15:0] bits = 0x0001 (mc_state_idle).
7. Write 0 to the HWPINTF.HWP0IF bit. (Clear interrupt flag)

Memory check error

When an error occurs during processing of the Memory Check function, the HWPINTF.HWP1IF bit is set to 1 (an interrupt can be generated). The error contents can be confirmed by reading the ERROR.ERROR[15:0] bits. As shown in Table 22.4.2.2, the ERROR.ERROR*x* bit corresponding to the error that has occurred is set to 1.

Table 22.4.2.2 List of Memory Check Errors

ERROR.ERROR[15:0] bits	Error	Meaning
0000 0000 0000 0000	error_no_error	No error has occurred.
Non-fatal error		
xxxx xxxx xxxx xxx1 (bit 0)	error_command	A command that is undefined or is ineffective in the current state has been specified.
Fatal error		
x1xx xxxx xxxx xxxx (bit 14)	error_function_id	An undefined function ID has been specified.
1xxx xxxx xxxx xxxx (bit 15)	error_others	Another error has occurred.

When a non-fatal error has occurred, reissue a valid command.
When a fatal error has occurred, remove the cause of error and redo the processing from initialization.

22.4.3 External QSPI Flash Memory Access

Initialization

When accessing an external QSPI Flash memory through the HWP function, set it into memory mapped access mode as in the procedure below before performing the initialization described in Section 22.4.1 or 22.4.2.

1. Perform Steps 1 to 6 described in Section 15.5.3.
   However, a part of Steps 3 and 6 must be changed as follows:
   - In Step 3, set the QSPI_*n*MB.XIPEXT[7:0] bits to the same value as the QSPI_*n*MB.XIPACT[7:0] bits.
   - In Step 6, set the interrupt enable bits in the QSPI_*n*INTE register to 0 (interrupt disabled).
2. Perform Steps 1 to 3 described in Section 15.5.6.

End processing

When switching the QSPI from memory mapped access mode to register access mode after the HPW function has been completed, follow the procedure below to terminate the memory mapped access.

1. Before disabling the HWP, set the mode byte for terminating the XIP session to the QSPI_*n*MB.XIPEXT[7:0] bits.
2. Disable the HWP.
3. Perform the procedure described in Section 15.5.7.

22.5 Interrupts

The HWP has a function to generate the interrupts shown in Table 22.5.1.

Table 22.5.1 HWP Interrupt Function

Interrupt	Interrupt flag	Set condition	Clear condition
Error occurrence	HWPINTF.HWP1IF	When a sound play error (see Table 22.4.1.2) or a memory check error (see Table 22.4.2.2) has occurred	Writing 0.
State transition	HWPINTF.HWP0IF	When a state transition of which the interrupt has not been masked occurs	Writing 0.

Interrupt enable

To enable HWP interrupts, the HWPINTE.HWPIE bit must be set to 1. An interrupt request is sent to the CPU only when the interrupt flag is set in this status. For more information on interrupt control, refer to the "Interrupt" chapter.

State transition interrupt mask

The HWP provides a HWP internal register that contains the interrupt mask bits used for setting whether to set the HWPINTF.HWP0IF bit (to generate an interrupt) or not when a state transition occurs. By setting the interrupt mask bits to 0, an interrupt can be generated when the corresponding state transition occurs.

Table 22.5.2 State Transition Interrupt Mask Bits

Function	Interrupt mask bit	State transition
Sound Play	INTMASK.TO_MUTE	sp_state_play state → sp_state_mute state
	INTMASK.TO_PAUSE	sp_state_play state → sp_state_pause state
	INTMASK.TO_PLAY	sp_state_idle, mute, pause state → sp_state_play state
	INTMASK.TO_IDLE	sp_state_init, mute, pause, play state → sp_state_idle state
Memory Check	INTMASK.TO_PROCESSING	mc_state_idle state → mc_state_ram_rw, ram_march_c, checksum, crc state
	INTMASK.TO_IDLE	mc_state_init, mc_state_ram_rw, ram_march_c, checksum, crc state → mc_state_idle state

22.6 HWP Internal Registers

The HWP internal registers are switched to the Sound Play function registers or the Memory Check function registers according to the set value of the FUNCTION.ID[7:0] bits when the HWPCTL.HWPEN bit is set to 1. Table 22.6.1 lists the HWP internal register map.

Table 22.6.1 HWP Internal Register Map

Address	Register name			
	Sound Play function		Memory Check function	
Base + 0x00	FUNCTION	Function ID Register	FUNCTION	Function ID Register
Base + 0x02	INTMASK	Interrupt Mask Register	INTMASK	Interrupt Mask Register
Base + 0x04	ROMADDR	ROM Address Register	MEMADDR	Memory Address Register
Base + 0x08	ROMSIZE	ROM Size Register	MEMSIZE	Memory Size Register
Base + 0x0c	KEYCODE	Key Code Register	INITVALUE	Initial Value Setting Register
Base + 0x10	COMMAND_0	Ch.0 Command Register	COMMAND	Command Register
Base + 0x12	COMMAND_1	Ch.1 Command Register	–	–
Base + 0x14	SENTENCE_0	Ch.0 Sentence Number Setting Register	–	–
Base + 0x16	SENTENCE_1	Ch.1 Sentence Number Setting Register	–	–
Base + 0x18	VOLUME_0	Ch.0 Volume Control Register	–	–
Base + 0x1a	VOLUME_1	Ch.1 Volume Control Register	–	–
Base + 0x1c	REPEAT_0	Ch.0 Repeat Control Register	–	–
Base + 0x1e	REPEAT_1	Ch.1 Repeat Control Register	–	–
Base + 0x20	SPEED_0	Ch.0 Playback Speed Conversion Register	–	–
Base + 0x24	PITCH_0	Ch.0 Playback Pitch Conversion Register	–	–
Base + 0x40	STATE_0	Ch.0 State Monitor Register	STATE	State Monitor Register
Base + 0x42	STATE_1	Ch.1 State Monitor Register	–	–
Base + 0x44	ERROR	Error Status Register	ERROR	Error Status Register
Base + 0x46	STATUS	Operating Status Register	STATUS	Operating Status Register
Base + 0x48	–	–	RESULT	Calculation Result Register
Base + 0x4c	VERSION	Version Number Register	VERSION	Version Number Register

Base = 0x00156700

22.6.1 Sound Play Function Registers

Function ID Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
FUNCTION	15–8	–	x	–	R/W	–
	7–0	ID[7:0]	x	–	R/W	

Bits 15–8 Reserved

Set to 0x00 when writing data to this register.

Bits 7–0 ID[7:0]

These bits select the function to be executed in the HWP.

Table 22.6.1.1 Function ID

FUNCTION.ID[7:0] bits	HWP function
0x03	Memory Check function
0x01	Sound Play function using SDAC2
Other	Setting prohibited (error)

Interrupt Mask Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
INTMASK (Sound Play)	15–8	–	x	–	R/W	–
	7–4	–	x	–	R/W	
	3	TO_MUTE	x	–	R/W	
	2	TO_PAUSE	x	–	R/W	
	1	TO_PLAY	x	–	R/W	
	0	TO_IDLE	x	–	R/W	

Bits 15–4 Reserved

Set to 0x000 when writing data to this register.

Bit 3 TO_MUTE
Bit 2 TO_PAUSE
Bit 1 TO_PLAY
Bit 0 TO_IDLE

These bits set whether the interrupt request when a state transition occurs during executing the Sound Play function is enabled or not.

1 (W): Mask interrupt (disabled)
0 (W): Enable interrupt

For more information on the state transition interrupts that can be masked with these bits, refer to Table 22.5.2.

ROM Address Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ROMADDR (Sound Play)	31–0	ADDRESS[31:0]	x	–	R/W	–

Bits 31–0 ADDRESS[31:0]

These bits specify the sound data ROM start address.
The address should be specified with a value shown below.

In case of internal Flash:
0x00 0000, ⋯, 0x02 fff0 (16-byte alignment)

In case of external QSPI-Flash:
0x00 0000 + OFFSET
0x10 0000 + OFFSET
0x20 0000 + OFFSET
...
0xe0 0000 + OFFSET
0xf0 0000 + OFFSET

* The OFFSET is 0x04 0000, the start address of the memory mapped access area for external Flash memory (refer to "Figure 4.1.1 Memory Map").

ROM Size Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ROMSIZE (Sound Play)	31–0	SIZE[31:0]	x	–	R/W	–

Bits 31–0 ADDRESS[31:0]

These bits specify the sound data ROM size in bytes.
The following shows the maximum size that can be specified.

In case of internal Flash:
0x03 0000 bytes (192K bytes) or less

In case of external QSPI-Flash:
0x100 0000 bytes (16M bytes) or less

Key Code Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
KEYCODE (Sound Play)	31–0	KEYCODE[31:0]	x	–	R/W	–

Bits 31–0 KEYCODE[31:0]

These bits specify the key code.
Write the key code provided by Seiko Epson.

Ch.*n* Command Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
COMMAND_*n* (Sound Play)	15–8	OPTION[7:0]	x	–	R/W	–
	7–0	COMMAND[7:0]	x	–	R/W	

Bits 15–8 OPTION[7:0]

These bits select a command option (gapless play/silent period insertion).

Table 22.6.1.2 Command Option Selection

COMMAND_*n*. OPTION[7:0] bits	Command Option
0xff–0x02	Setting prohibited (error)
0x01	Gapless play enable Sound will be played without a silent period inserted (gap = 0 ms) between phrases.
0x00	Gapless play disable Sound will be played with about 100 ms of a silent period (gap) inserted between phrases.

Notes:
- During 2-channel mix output with Ch.0 and Ch.1, gapless play cannot be performed (gapless play is disabled for both channels).
- Depending on the contents of the sentence to play, the played voice may be difficult to hear even if gapless play is enabled. Conduct sufficient evaluation when using the gapless play function.

Bits 7–0 COMMAND[7:0]

These bits select a sound play command to be executed.

Table 22.6.1.3 Sound Play Command Selection

COMMAND_*n*.COMMAND[7:0] bits	Sound play command
0xff–0x0a	Setting prohibited (error)
0x09	Release Mute
0x08	Mute after Current Phrase
0x07	Mute Immediately
0x06	Release Pause
0x05	Pause after Current Phrase
0x04	Pause Immediately
0x03	Sound Stop after Current Phrase
0x02	Sound Stop Immediately
0x01	Sound Start
0x00	No operation

Ch.*n* Sentence Number Setting Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SENTENCE_*n* (Sound Play)	15–0	SENTENCE_NO[15:0]	x	–	R/W	–

Bits 15–0 SENTENCE_NO[15:0]

These bits specify the sentence number of the sound to be played.
Set the sentence number that was listed in the "Voice Creation Tool (ESPER2)."

Ch.*n* Volume Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
VOLUME_*n* (Sound Play)	15–0	VOLUME[15:0]	x	–	R/W	–

Bits 15–0 VOLUME[15:0]

These bits specify the playback volume.

表22.6.1.4 Volume Setting

VOLUME_*n*.VOLUME[15:0] bits	Volume level
0xff–0x80	Setting prohibited (error)
0x7f	0 db
0x7e	-0.5 db
0x7d	-1.0 db
:	Can be specified in 0.5 db steps.
0x02	-62.5 db
0x01	-63.0 db
0x00	Silent

Ch.*n* Repeat Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
REPEAT_*n* (Sound Play)	15–0	REPEAT[15:0]	x	–	R/W	–

Bits 15–0 REPEAT[15:0]

These bits specify the number of repeat playbacks.

Table 22.6.1.5 Setting of Number of Repeat Playbacks

REPEAT_*n*.REPEAT[15:0] bits	Playback counts
0xff	Repeat until Sound Stop command execution
0xfe	254 times
0x7e	253 times
0x7d	252 times
:	:
0x3	3 times
0x02	2 times
0x01	1 time (no repetition)
0x00	Setting prohibited

Ch.0 Playback Speed Conversion Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SPEED_0 (Sound Play)	15–0	SPEED[15:0]	x	–	R/W	–

Bits 15–0 SPEED[15:0]

These bits specify the playback speed.

When converting the speed in combination with pitch conversion, the speed setting value should be specified within the range shown in Table 22.6.1.6.

When converting the speed alone without pitch conversion, the PITCH_0.PITCH[15:0] bits should be set to 0x00 and the speed setting value should be specified within the range shown in Table 22.6.1.7.

Table 22.6.1.6 Playback Speed Setting (0x5a ≤ PITCH_0.PITCH[15:0] bits ≤ 0x6e)

SPEED_0.SPEED[15:0] bits	Playback speed	
0x73	115%	Fast
0x6e	110%	↑
0x69	105%	
0x64	100%	Standard speed
0x5f	95%	
0x5a	90%	↓
0x55	85%	Slow
Other	Setting prohibited	

Table 22.6.1.7 Playback Speed Setting (PITCH_0.PITCH[15:0] bits = 0x00)

SPEED_0.SPEED[15:0] bits	Playback speed	
0x7d	125%	Fast
0x78	120%	↑
0x73	115%	
0x6e	110%	
0x69	105%	
0x64	100%	Standard speed
0x5f	95%	
0x5a	90%	
0x55	85%	
0x50	80%	↓
0x4b	75%	Slow
Other	Setting prohibited	

Ch.0 Playback Pitch Conversion Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
PITCH_0 (Sound Play)	15–0	PITCH[15:0]	x	–	R/W	–

Bits 15–0 PITCH[15:0]

These bits specify the playback pitch (or musical interval).

When converting the pitch in combination with speed conversion, the pitch setting value should be specified within the range shown in Table 22.6.1.8.

When converting the pitch alone without speed conversion, the SPEED_0.SPEED[15:0] bits should be set to 0x00 and the pitch setting value should be specified within the range shown in Table 22.6.1.9.

Table 22.6.1.8 Pitch Setting (0x55 ≤ SPEED_0.SPEED[15:0] bits ≤ 0x73)

PITCH_0.PITCH[15:0] bits	Pitch	
0x6e	110%	High
0x69	105%	↑
0x64	100%	Standard pitch
0x5f	95%	↓
0x5a	90%	Low
Other	Setting prohibited	

Table 22.6.1.9 Pitch Setting (SPEED_0.SPEED[15:0] bits = 0x00)

PITCH_0.PITCH[15:0] bits	Pitch	
0x7d	125%	High
0x78	120%	↑
0x73	115%	
0x6e	110%	
0x69	105%	
0x64	100%	Standard pitch
0x5f	95%	
0x5a	90%	
0x55	85%	
0x50	80%	↓
0x4b	75%	Low
Other	Setting prohibited	

Table 22.6.1.10 6.1.10 Setting Values to Convert Speed and Pitch Simultaneously

			PITCH_0.PITCH[15:0] bits											
			0x7d	0x78	0x73	0x6e	0x69	0x64	0x5f	0x5a	0x55	0x50	0x4b	0x00
			125%	120%	115%	110%	105%	100%	95%	90%	85%	80%	75%	–
SPEED_0. SPEED[15:0] bits	0x7d	125%	–	–	–	–	–	–	–	–	–	–	–	✓
	0x78	120%	–	–	–	–	–	–	–	–	–	–	–	✓
	0x73	115%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x6e	110%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x69	105%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x64	100%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x5f	95%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x5a	90%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x55	85%	–	–	–	✓	✓	✓	✓	✓	–	–	–	✓
	0x50	80%	–	–	–	–	–	–	–	–	–	–	–	✓
	0x4b	75%	–	–	–	–	–	–	–	–	–	–	–	✓
	0x00	–	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Ch.*n* State Monitor Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
STATE_*n* (Sound Play)	15–0	STATE[15:0]	x	–	R	–

Bits 15–0 STATE[15:0]

These bits indicate the current state of the Sound Play function.

Table 22.6.1.11 State Monitor

STATE_*n*.STATE[15:0] bits	State
0x0004	sp_state_mute
0x0003	sp_state_pause
0x0002	sp_state_play
0x0001	sp_state_idle
0x0000	sp_state_init

Error Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ERROR	15–0	ERROR[15:0]	x	–	R	–

Bits 15–0 ERROR[15:0]

These bits indicate the error that has occurred while the HWP is running. For the errors that may occur in the Sound Play function, refer to Table 22.4.1.2.

Operating Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
STATUS (Sound Play)	15–9	–	x	–	R	–
	8	SOUNDOUT	x	–	R	
	7–1	–	x	–	R	
	0	READY	x	–	R	

Bits 15–9 Reserved

Bit 8 SOUNDOUT

This bit indicates whether the HWP is performing playback output or not.

1 (R): During playback (sp_state_play, sp_state_mute)

0 (R): Stop (sp_state_init, sp_state_idle, sp_state_pause)

Bits 7–1 Reserved

Bit 0 READY

This bit indicates the HWP operating status (whether a command is acceptable or not).

1 (R): Ready (Command is acceptable.)

0 (R): Busy (Command is not acceptable.)

Version Number Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
VERSION	15–8	MAJOR[7:0]	x	–	R	–
	7–0	MINOR[7:0]	x	–	R	

Bits 15–8 MAJOR[7:0]

Bits 7–0 MINOR[7:0]

These bits indicate the HWP version number.

Version number = MAJOR[7:0] . MINOR[7:0]

22.6.2 Memory Check Function Register

Function ID Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
FUNCTION	15–8	–	x	–	R/W	–
	7–0	ID[7:0]	x	–	R/W	

Bits 15–8 Reserved

Set to 0x00 when writing data to this register.

Bits 7–0 ID[7:0]

These bits select the function to be executed in the HWP. (Refer to Table 22.6.1.1.)
Set to 0x03 when executing the Memory Check function.

Interrupt Mask Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
INTMASK (Memory Check)	15–8	–	x	–	R/W	–
	7–2	–	x	–	R/W	
	1	TO_PROCESSING	x	–	R/W	
	0	TO_IDLE	x	–	R/W	

Bits 15–2 Reserved

Set to 0x000(0) when writing data to this register.

Bit 1 TO_PROCESSING
Bit 0 TO_IDLE

These bits set whether the interrupt request when a state transition occurs during executing the Memory Check function is enabled or not.

1 (W): Mask interrupt (disabled)
0 (W): Enable interrupt

For more information on the state transition interrupts that can be masked with these bits, refer to Table 22.5.2.

Memory Address Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MEMADDR (Memory Check)	31–0	ADDRESS[31:0]	x	–	R/W	–

Bits 31–0 ADDRESS[31:0]

These bits specify the memory check start address.
The address should be specified within the range shown below.

In case of RAM:
0x15 0000, ⋯, 0x15 1fff
0x15 3000, ⋯, 0x15 67ff

In case of internal Flash:
0x00 0000, ⋯, 0x02 ffff

In case of external QSPI-Flash:
0x00 0000 + OFFSET, ⋯, 0x0f ffff + OFFSET

* The OFFSET is 0x04 0000, the start address of the memory mapped access area for external Flash memory (refer to “Figure 4.1.1 Memory Map”).

Memory Size Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
MEMSIZE (Memory Check)	31–0	SIZE[31:0]	x	–	R/W	–

Bits 31–0 SIZE[31:0]

These bits specify the size in bytes of the memory to be checked.

Initial Value Setting Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
INITVALUE (Memory Check)	31–0	INITVALUE[31:0]	x	–	R/W	–

Bits 31–0 INITVALUE[31:0]

These bits set the initial value used for the calculation of the Flash check (checksum, CRC). Normally, set to 0x0000 0000.

Command Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
COMMAND (Memory Check)	15–8	–	x	–	R/W	–
	7–0	COMMAND[7:0]	x	–	R/W	

Bits 15–8 Reserved

Set to 0x00 when writing data to this register.

Bits 7–0 COMMAND[7:0]

These bits select a memory check command to be executed.

Table 22.6.2.1 Memory Check Command Selection

COMMAND.COMMAND[7:0] bits	Memory check command
0xff	Memory Check Stop
0xfe–0x06	Setting prohibited (error)
0x05	Flash CRC Start
0x04	Flash Checksum Start
0x03	RAM Check March-C Start
0x02	RAM Check R/W Start
0x00, 0x01	No operation

State Monitor Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
STATE (Memory Check)	15–0	STATE[15:0]	x	–	R	–

Bits 15–0 STATE[15:0]

These bits indicate the current state of the Memory Check function.

Table 22.6.2.2 State Monitor

STATE.STATE[15:0] bits	State
0x0005	mc_state_crc
0x0004	mc_state_checksum
0x0003	mc_state_ram_march_c
0x0002	mc_state_ram_rw
0x0001	mc_state_idle
0x0000	mc_state_init

Error Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
ERROR	15–0	ERROR[15:0]	x	–	R	–

Bits 15–0 ERROR[15:0]

These bits indicate the error that has occurred while the HWP is running. For the errors that may occur in the Memory Check function, refer to Table 22.4.2.2.

Operating Status Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
STATUS (Memory Check)	15–10	–	x	–	R	–
	9–8	PROCESSING[1:0]	x	–	R	
	7–1	–	x	–	R	
	0	READY	x	–	R	

Bits 15–10 Reserved

Bits 9–8 PROCESSING[1:0]

These bits indicate the memory check processing status.

Table 22.6.2.3 Memory Check Processing Status

STATUS. PROCESSING[1:0] bits	Memory check processing status	
	RAM check	Flash check
0x0003	Terminated with an error	–
0x0002	Completed successfully	Completed
0x0001	During checking	During checking
0x0000	During standby	During standby

Bits 7–1 Reserved

Bit 0 READY

This bit indicates the HWP operating status (whether a command is acceptable or not).

1 (R): Ready (Command is acceptable.)

0 (R): Busy (Command is not acceptable.)

Calculation Result Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
RESULT (Memory Check)	31–0	RESULT[31:0]	x	–	R	–

Bits 31–0 RESULT[31:0]

These bits indicate the memory check result.

When the RAM check has completed

Indicates the address where a first error has occurred.

When the Flash check has completed

Indicates the checksum/CRC calculation result. Compare it with the original value to confirm whether there is an error or not.

Version Number Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
VERSION	15–8	MAJOR[7:0]	x	–	R	–
	7–0	MINOR[7:0]	x	–	R	

Bits 15–8 MAJOR[7:0]

Bits 7–0 MINOR[7:0]

These bits indicate the HWP version number.

Version number = MAJOR[7:0] . MINOR[7:0]

22.7 Control Registers

HWP Control Register

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

Bits 15–1 Reserved

Bit 0 HWPEN

This bit enables the HWP operations.

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

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

HWP Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
HWPINTF	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	1	HWP1IF	0	H0	R/W	Cleared by writing 0.
	0	HWP0IF	0	H0	R/W	

Bits 15–2 Reserved

Bit 1 HWP1IF
Bit 0 HWP0IF

These bits indicate the HWP interrupt cause occurrence status.

1 (R): Cause of interrupt occurred

0 (R): No cause of interrupt occurred

1 (W): Setting prohibited

0 (W): Clear flag

The following shows the correspondence between the bit and interrupt:

HWPINTF.HWP1IF bit: Error occurrence interrupt

HWPINTF.HWP0IF bit: State transition interrupt

Note: Be aware that the writing value to clear the flags is different from other peripheral circuits.

HWP Interrupt Enable Register

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

Bits 15–1 Reserved

Bit 0 HWPIE

This bit enables HWP interrupts.

1 (R/W): Enable interrupts

0 (R/W): Disable interrupts

HWP Command Trigger Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
HWPCMDTRG	15–8	–	0x00	–	R	–
	7–1	–	0x00	–	R	
	0	HWP0TRG	0	H0	R/W	

Bits 15–1 Reserved

Bit 0 HWP0TRG

This bit starts executing the command specified by the HWP internal register.

1 (W): Trigger to issue command
0 (W): Setting prohibited
1 (R): In command issuing process
0 (R): Command issuance completed/standby to issue command

SDAC2 Clock Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2CLK	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 SDAC2 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 SDAC2 operating clock.

Bits 3–2 Reserved

Bits 1–0 CLKSRC[1:0]

These bits select the clock source of SDAC2.

Table 22.7.1 Clock Source and Division Ratio Settings

SDAC2CLK. CLKDIV[1:0] bits	SDAC2CLK.CLKSRC[1:0] bits			
	0x0	0x1	0x2	0x3
	IOSC	OSC1	OSC3	EXOSC
0x3	Reserved	Reserved	Reserved	Reserved
0x2	1/4		1/4	1/4
0x1	1/2		1/2	1/2
0x0	1/1		1/1	1/1

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

Note: The SDAC2CLK register settings can be altered only when the SDAC2CTL.SDACEN bit = 0.

SDAC2 Control Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2CTL	15–8	–	0x00	–	R	–
	7–4	–	0x00	–	R	
	3	TONEON	0	H0	R/W	
	2	–	0	–	R	
	1	RESAMPEN	0	H0	R/W	
	0	SDACEN	0	H0	R/W	

Bits 15–4 Reserved

Bit 3 TONEON

This bit enables the square-wave tone generator.

1 (R/W): Turn on square-wave tone
0 (R/W): Turn off square-wave tone

Bit 2 **Reserved**

Bit 1 **RESAMPEN**

This bit enables the resampler.

1 (R/W): Enable resampler

0 (R/W): Disable resampler

Bit 0 **SDACEN**

This bit enables the SDAC operations.

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

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

SDAC2 Mode Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2MOD	15–9	–	0x00	–	R	–
	8	PWMOUTEN	0	H0	R/W	
	7–2	–	0x00	H0	R	
	1–0	PWMMODE[1:0]	0x00	H0	R/W	

Bits 15–9 **Reserved**

Bit 8 **PWMOUTEN**

This bit enables the SDAC2 output pins to output the PWM signals.

1 (R/W): Enable PWM signal output

0 (R/W): Disable PWM signal output

Bits 7–2 **Reserved**

Bits 1–0 **PWMMODE[1:0]**

These bits set th SDAC2 operating mode.

Table 22.7.2 SDAC2 Operating Mode

SDAC2MOD.PWMMODE[1:0] bits	Mode
0x3	CPLM mode 2
0x2	CPLM mode 1
0x1	Normal mode
0x0	Buzzer mode

SDAC2 Ch.*n* Data Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2_*n*DAT	15–10	–	0x00	–	R	–
	9–0	DAT[9:0]	0x000	H0	R/W	

Bits 15–10 **Reserved**

Bits 9–0 **DAT[9:0]**

These bits store sound data.

Note: This register is used by the HWP. Do not write any data to this register while the HWP operation is enabled (HWPCTL.HWPEN bit = 1).

SDAC2 Interrupt Flag Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2INTF	15–8	–	0x00	–	R	–
	7–4	–	0x0	–	R	
	3	ERR1IF	0	H0	R/W	Cleared by writing 1.
	2	DATREQ1IF	0	H0	R/W	
	1	ERR0IF	0	H0	R/W	
	0	DATREQ0IF	0	H0	R/W	

Bits 15–4 **Reserved**

Bit 3 ERR1IF
Bit 2 DATREQ1IF
Bit 1 ERR0IF
Bit 0 DATREQ0IF

These bits indicate the SDAC2 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:
SDAC2INTF.ERR1IF bit: Ch.1 error occurrence interrupt
SDAC2INTF.DATREQ1IF bit: Ch.1 data request interrupt
SDAC2INTF.ERR0IF bit: Ch.0 error occurrence interrupt
SDAC2INTF.DATREQ0IF bit: Ch.0 data request interrupt

Note: This register is used by the HWP. Do not write any data to this register while the HWP operation is enabled (HWPCTL.HWPEN bit = 1).

SDAC2 Interrupt Enable Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2INTE	15–8	–	0x00	–	R	–
	7–2	–	0x00	–	R	
	3	ERR1IE	0	H0	R/W	
	2	DATREQ1IE	0	H0	R/W	
	1	ERR0IE	0	H0	R/W	
	0	DATREQ0IE	0	H0	R/W	

Bits 15–4 Reserved

Bit 3 ERR1IE
Bit 2 DATREQ1IE
Bit 1 ERR0IE
Bit 0 DATREQ0IE

These bits enable SDAC2 interrupts.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

The following shows the correspondence between the bit and interrupt:
SDAC2INTE.ERR1IE bit: Ch.1 error occurrence interrupt
SDAC2INTE.DATREQ1IE bit: Ch.1 data request interrupt
SDAC2INTE.ERR0IE bit: Ch.0 error occurrence interrupt
SDAC2INTE.DATREQ0IE bit: Ch.0 data request interrupt

Note: This register is used by the HWP. Do not write any data to this register while the HWP operation is enabled (HWPCTL.HWPEN bit = 1).

SDAC2 Resampler Rate Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2RESAMP	15–11	–	0x00	–	R	–
	10–0	RESAMPRATE[10:0]	0x400	H0	R/W	

Bits 15–11 Reserved

Bits 10–0 RESAMPRATE[10:0]

These bits set the audio sampling frequency of the SDAC2.
The audio sampling frequency is calculated as follows:

$$\text{Sampling frequency} = f_{SDAC2CLK} \times (1{,}024 / \text{RESAMPRATE}) \qquad \text{(Eq. 22.1)}$$

where

$f_{SDAC2CLK}$: SDAC2 operating clock frequency set using the SDAC2CLK register [Hz]
RESAMPRATE: Value set in the SDAC2RESAMP.RESAMPRATE[10:0] bits

Note: This register is used by the HWP. Do not write any data to this register while the HWP operation is enabled (HWPCTL.HWPEN bit = 1).

SDAC2 Tone Divider Register

Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
SDAC2TONE	15–0	TONEDIV[15:0]	0x4000	H0	R/W	–

Bits 15–0 TONEDIV[15:0]

These bits set the frequency of the square-wave tone frequency when the SDAC2CTL.TONEON bit = 1 (square-wave tone generator enabled).
The tone frequency is calculated as follows:

$$\text{Tone frequency} = f_{SDAC2CLK} / [(4 \times \text{TONEDIV} + 4) \times 2] \qquad \text{(Eq. 22.2)}$$

where

$f_{SDAC2CLK}$: SDAC2 operating clock frequency set using the SDAC2CLK register [Hz]
TONEDIV: Value set in the SDAC2TONE.TONEDIV[15:0] bits

23 Electrical Characteristics

23.1 Absolute Maximum Ratings

(V_{SS} = 0 V)

Item	Symbol	Condition	Rated value	Unit
Power supply voltage	V_{DD}		-0.3 to 7.0	V
QSPI-Flash interface power supply voltage	V_{DDQSPI}		-0.3 to 7.0	V
Flash programming voltage	V_{PP}		-0.3 to 8.0	V
Input voltage	V_I	#RESET, TEST, P10–17, P40, P42–43, PD2–D3	-0.3 to V_{DD} + 0.5	V
		P00–07, P20–27, P30–37, P41, P44–45, P50–56, P60–65, PD0–D1	-0.3 to 7.0	V
Output voltage	V_O		-0.3 to V_{DD} + 0.5	V
High level output current	I_{OH}	1 pin	-10	mA
		Total of all pins	-20	mA
Low level output current	I_{OL}	1 pin	10	mA
		Total of all pins	20	mA
Operating temperature	Ta		-40 to 85	°C
Storage temperature	Tstg		-65 to 125	°C

23.2 Recommended Operating Conditions

(V_{SS} = 0 V) ∗1

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
Power supply voltage	V_{DD}	For normal operation		1.8	–	5.5	V
		V_{D1} voltage mode = mode1		1.8	–	3.6	V
		For Flash programming		2.2	–	5.5	V
QSPI-Flash interface power supply voltage	V_{DDQSPI}	For P60–65 and QSPI	When QSPI is used	3.0	–	3.6	V
			When QSPI is not used	1.8	–	5.5	V
OSC1 oscillator oscillation frequency	f_{OSC1}	Crystal oscillator		–	32.768	–	kHz
OSC3 oscillator oscillation frequency	f_{OSC3}	Crystal/ceramic oscillator		0.2	–	16.3	MHz
EXOSC external clock frequency	f_{EXOSC}	When supplied from an external oscillator		0.016	–	16.3	MHz
Bypass capacitor between V_{SS} and V_{DD}	C_{PW1}			–	3.3	–	µF
Capacitor between V_{SS} and V_{D1}	C_{PW2}			–	1.0	1.2	µF
Capacitor between V_{SS} and V_{DDQSPI}	$C_{VDDQSPI}$			–	3.3	–	µF
Gate capacitor for OSC1 oscillator	C_{G1}	When crystal oscillator is used ∗2		0	–	25	pF
Drain capacitor for OSC1 oscillator	C_{D1}	When crystal oscillator is used ∗2		–	0	–	pF
Gate capacitor for OSC3 oscillator	C_{G3}	When crystal/ceramic oscillator is used ∗2		0	–	100	pF
Drain capacitor for OSC3 oscillator	C_{D3}	When crystal/ceramic oscillator is used ∗2		0	–	100	pF
Debug pin pull-up resistors	R_{DBG1-2}	∗3		–	100	–	kΩ
Capacitor between V_{SS} and V_{PP}	C_{VPP}			–	0.1	–	µF
Capacitor between V_{SS} and V_{REFA}	C_{VREFA}			–	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 component values should be determined after performing matching evaluation of the resonator mounted on the printed circuit board actually used.

∗3 R_{DBG1-2} are not required when using the debug pins as general-purpose I/O ports.

∗4 The component values should be determined after evaluating operations using an actual mounting board.

23.3 Current Consumption

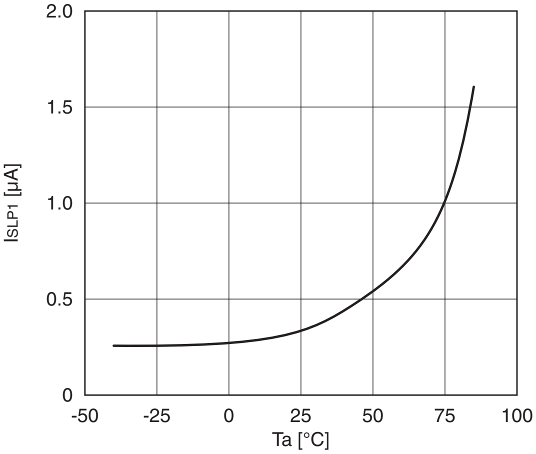
Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = 25°C, EXOSC = OFF, PWGACTL.REGMODE[1:0] bits = 0x0 (automatic mode), PWGACTL.REGSEL bit = 1 (mode0), FLASHCWAIT.RDWAIT[1:0] bits = 0x1 (2 cycles)

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Current consumption in SLEEP mode	I_{SLP1}	IOSC = OFF, OSC1 = OFF, OSC3 = OFF	–	0.34	4	μA
	I_{SLP2}	IOSC = OFF, OSC1 = OFF, OSC3 = OFF, PWGACTL.REGSEL bit = 0 (mode1)	–	0.27	3.5	μA
	I_{SLP3}	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = OFF, RTCA = ON	–	0.90	6	μA
	I_{SLP4}	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = OFF, RTCA = ON, PWGACTL.REGSEL bit = 0 (mode1)	–	0.80	5.5	μA
Current consumption in HALT mode	I_{HALT1}	IOSC = 8 MHz*1, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = IOSC	–	500	710	μA
	I_{HALT2}	IOSC = 2 MHz*2, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = IOSC	–	120	172	μA
	I_{HALT3}	IOSC = 2 MHz*2, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = IOSC, PWGACTL.REGSEL bit = 0 (mode1)	–	70	100	μA
	I_{HALT4}	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = OSC1	–	1.5	8	μA
		IOSC = OFF, OSC1 = 32 kHz*4, OSC3 = OFF, SYSCLK = OSC1	–	2.5	12	μA
	I_{HALT5}	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = OSC1, PWGACTL.REGSEL bit = 0 (mode1)	–	1.2	7.5	μA
		IOSC = OFF, OSC1 = 32 kHz*4, OSC3 = OFF, SYSCLK = OSC1, PWGACTL.REGSEL bit = 0 (mode1)	–	2.2	11.5	μA
	I_{HALT6}	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = 16 MHz (ceramic oscillator)*5, SYSCLK = OSC3	–	630	1,050	μA
	I_{HALT7}	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = 16 MHz (internal oscillator)*6, SYSCLK = OSC3	–	710	1,060	μA
	I_{HALT8}*7	IOSC = OFF, OSC1 = OFF, OSC3 = 16 MHz (internal oscillator)*6, SYSCLK = OSC3, HWP/SDAC = ON		4,500	5,500	μA
Current consumption in RUN mode	I_{RUN1}*8	IOSC = 8 MHz*1, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = IOSC	–	1,750	2,580	μA
	I_{RUN2}*8	IOSC = 2 MHz*2, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = IOSC	–	430	650	μA
	I_{RUN3}*8	IOSC = 2 MHz*2, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = IOSC, PWGACTL.REGSEL bit = 0 (mode1)	–	260	390	μA
	I_{RUN4}*8	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = OSC1	–	6.5	14	μA
		IOSC = OFF, OSC1 = 32 kHz*4, OSC3 = OFF, SYSCLK = OSC1		7.5	18	μA
	I_{RUN5}*8	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = OFF, SYSCLK = OSC1, PWGACTL.REGSEL bit = 0 (mode1)	–	5	12	μA
		IOSC = OFF, OSC1 = 32 kHz*4, OSC3 = OFF, SYSCLK = OSC1, PWGACTL.REGSEL bit = 0 (mode1)	–	6	14	μA
	I_{RUN6}*8	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = 16 MHz (ceramic oscillator)*5, SYSCLK = OSC3	–	3,150	4,780	μA
	I_{RUN7}*8	IOSC = OFF, OSC1 = 32.768 kHz*3, OSC3 = 16 MHz (internal oscillator)*6, SYSCLK = OSC3	–	3,200	4,800	μA
	I_{RUN8}*9	IOSC = OFF, OSC1 = OFF, OSC3 = 16 MHz (internal oscillator)*6, SYSCLK = OSC3, HWP/SDAC = ON	–	5,600	8,500	μA

*1 IOSC oscillator: CLGIOSC.IOSCFQ[1:0] bits = 0x2
*2 IOSC oscillator: CLGIOSC.IOSCFQ[1:0] bits = 0x1
*3 OSC1 oscillator: CLGOSC1.OSC1SELCR bit = 0, 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)
*4 OSC1 oscillator: CLGOSC1.OSC1SELCR bit = 1
*5 OSC3 oscillator: CLGOSC3.OSC3MD bit = 1, CLGOSC3.OSC3INV[1:0] bits = 0x3, C_{G3} = C_{D3} = 10 pF
*6 OSC3 oscillator: CLGOSC3.OSC3MD bit = 0, CLGOSC3.OSC3FQ[1:0] bits = 0x3
*7 The current consumption value was measured when the HWP and SDAC2 was performing playback of 2 channels, SPEED_0.SPEED[15:0] bits = 0x7d (playback speed: 125%), PITCH_0.PITCH[15:0] bits = 0x00, and CPU was in HALT mode.
*8 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.
*9 The current consumption value was measured when the HWP and SDAC2 was performing playback of 2 channels, SPEED_0.SPEED[15:0] bits = 0x7d (playback speed: 125%), PITCH_0.PITCH[15:0] bits = 0x00, and the CPU was executing a loop containing a NOP instruction.

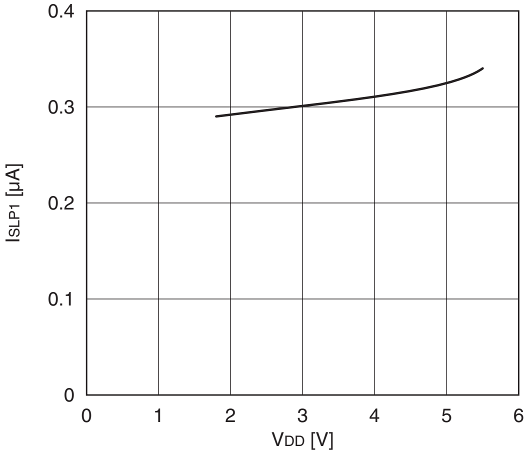
Current consumption-temperature characteristic in SLEEP mode

IOSC = OFF, OSC1 = OFF, OSC3 = OFF, Typ. value



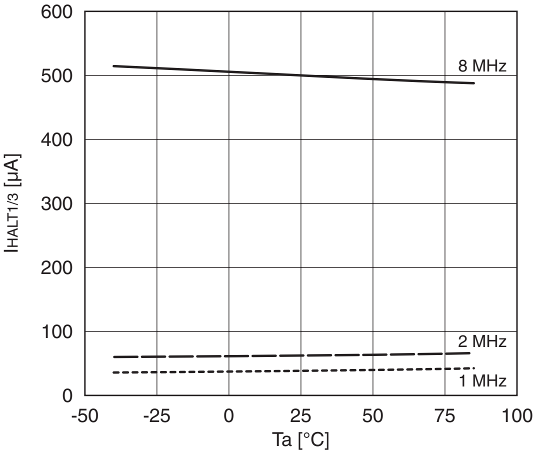
Current consumption-power supply voltage characteristic in SLEEP mode

IOSC = OFF, OSC1 = OFF, OSC3 = OFF, Typ. value



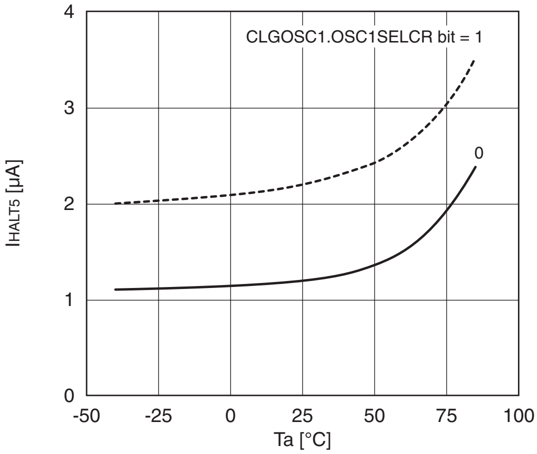
Current consumption-temperature characteristic in HALT mode (IOSC operation)

IOSC =ON, OSC1 = 32.768 kHz, OSC3 = OFF, Typ. value



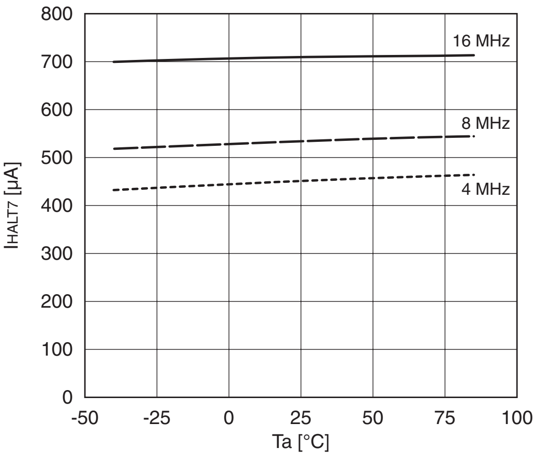
Current consumption-temperature characteristic in HALT mode (OSC1 operation)

IOSC = OFF, OSC1 = 32.768 kHz, OSC3 = OFF, Typ. value

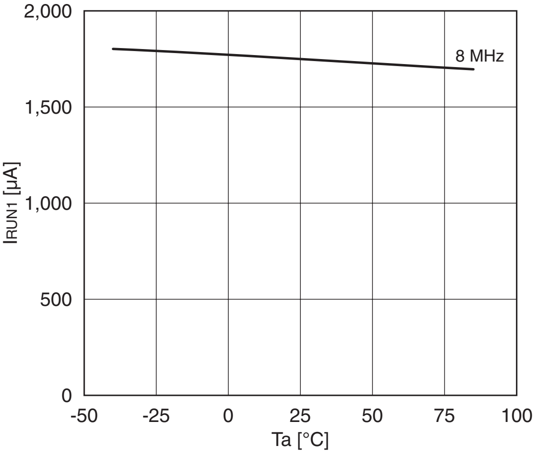


Current consumption-temperature characteristic in HALT mode (OSC3 operation)

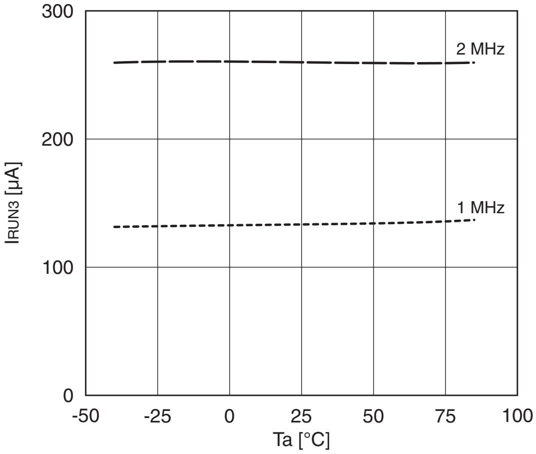
IOSC =OFF, OSC1 = 32.768 kHz, OSC3 = ON (internal oscillator), Typ. value



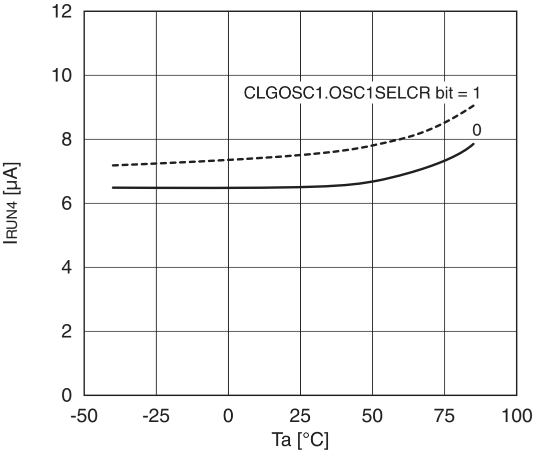
Current consumption-temperature characteristic in RUN mode (IOSC operation)
IOSC = ON, OSC1 = 32.768 kHz, OSC3 = OFF
PWGACTL.REGSEL bit = 1 (mode0), Typ. value



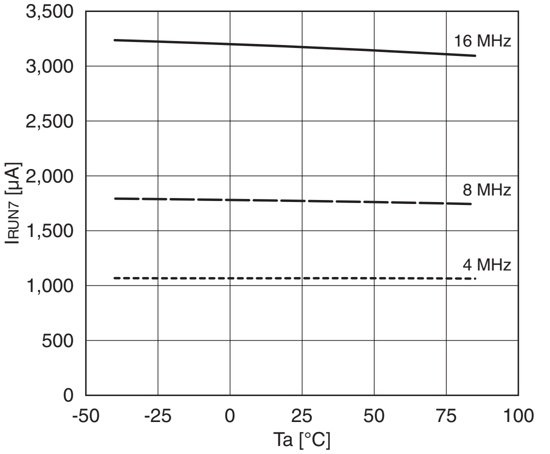
Current consumption-temperature characteristic in RUN mode (IOSC operation)
IOSC = ON, OSC1 = 32.768 kHz, OSC3 = OFF
PWGACTL.REGSEL bit = 0 (mode1), Typ. value



Current consumption-temperature characteristic in RUN mode (OSC1 operation)
IOSC = OFF, OSC1 = 32.768 kHz, OSC3 = OFF, Typ. value



Current consumption-temperature characteristic in RUN mode (OSC3 operation)
IOSC = OFF, OSC1 = 32.768 kHz, OSC3 = ON (internal oscillator), Typ. value

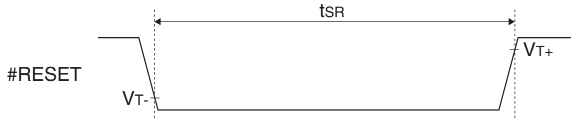


23.4 System Reset Controller (SRC) Characteristics

#RESET pin characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

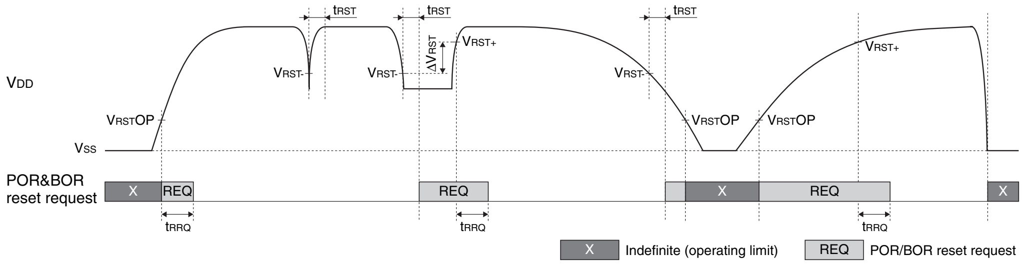
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
High level Schmitt input threshold voltage	V_{T+}		0.5 × V_{DD}	–	0.8 × V_{DD}	V
Low level Schmitt input threshold voltage	V_{T-}		0.2 × V_{DD}	–	0.5 × V_{DD}	V
Schmitt input hysteresis voltage	ΔV_T		180	–	–	mV
Input pull-up resistance	R_{IN}		100	200	500	kΩ
Pin capacitance	C_{IN}		–	–	15	pF
Reset Low pulse width	t_{SR}		25	–	–	µs



POR/BOR characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
POR/BOR canceling voltage	V_{RST+}		1.15	–	1.75	V
POR/BOR detection voltage	V_{RST-}		1.05	–	1.60	V
POR/BOR hysteresis voltage	ΔV_{RST}		40	60	–	mV
POR/BOR detection response time	t_{RST}		–	–	500	µs
POR/BOR operating limit voltage	$V_{RST}OP$		–	0.5	0.95	V
POR/BOR reset request hold time	t_{RRQ}		0.01	–	4	ms



Note: When performing a power-on-reset again after the power is turned off, decrease the V_{DD} voltage to $V_{RST}OP$ or less.

Reset hold circuit characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 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	–	1.8	ms

*1 Time until the internal reset signal is negated after the reset request is canceled.

23.5 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.

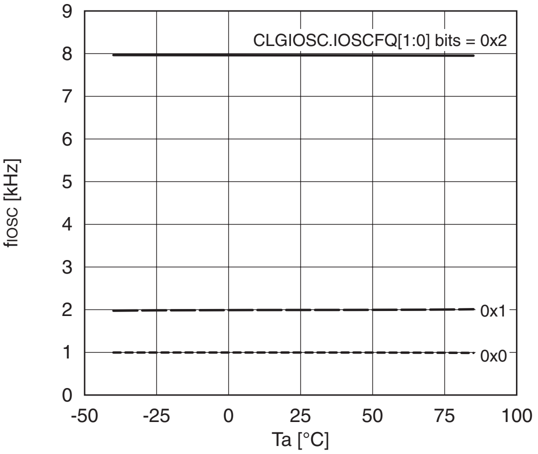
IOSC oscillator circuit characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

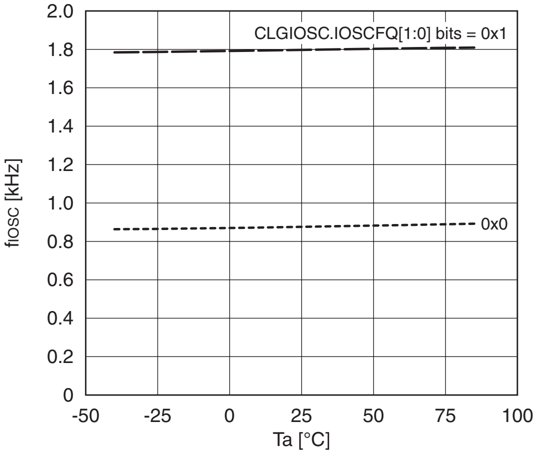
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{stal}		–	–	3	µs
Oscillation frequency	f_{IOSC}	CLGIOSC.IOSCFQ[1:0] bits = 0x2, PWGACTL.REGSEL bit = 1	7.2	8	8.4	MHz
		CLGIOSC.IOSCFQ[1:0] bits = 0x1, PWGACTL.REGSEL bit = 1	1.8	2	2.1	MHz
		CLGIOSC.IOSCFQ[1:0] bits = 0x0, PWGACTL.REGSEL bit = 1	0.9	1	1.05	MHz
		CLGIOSC.IOSCFQ[1:0] bits = 0x1, PWGACTL.REGSEL bit = 0	1.62	1.8	1.89	MHz
		CLGIOSC.IOSCFQ[1:0] bits = 0x0, PWGACTL.REGSEL bit = 0	0.78	0.9	1.02	MHz

IOSC oscillation frequency-temperature characteristic

V_{DD} = 1.8 to 5.5 V, PWGACTL.REGSEL bit = 1, Typ. value



V_{DD} = 1.8 to 5.5 V, PWGACTL.REGSEL bit = 0, Typ. value



OSC1 oscillator circuit characteristics

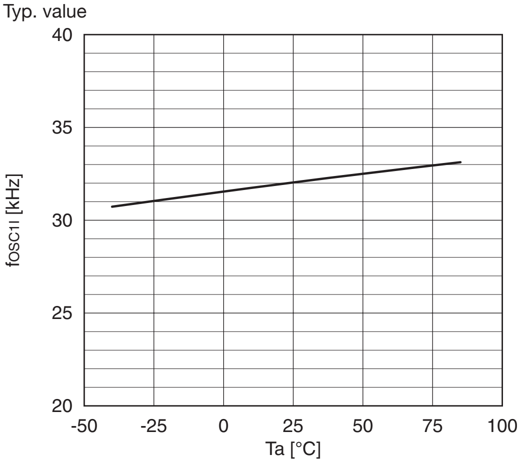
Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Crystal oscillator oscillation start time*1	t_{sta1C}	CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.INV1N[1:0] bits = 0x1, CLGOSC1.INV1B[1:0] bits = 0x2, CLGOSC1.OSC1BUP bit = 1	–	–	3	s
Crystal oscillator internal gate capacitance	C_{GI1C}	CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x0	–	12	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x1	–	14	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x2	–	16	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x3	–	18	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x4	–	19	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x5	–	21	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x6	–	23	–	pF
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.CGI1[2:0] bits = 0x7	–	24	–	pF
Crystal oscillator internal drain capacitance	C_{DI1C}	CLGOSC1.OSC1SELCR bit = 0,	–	6	–	pF
Crystal oscillator oscillator circuit current - oscillation inverter drivability ratio *1	I_{OSC1C}	CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.INV1N/INV1B[1:0] bits = 0x0	–	70	–	%
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.INV1N/INV1B[1:0] bits = 0x1 (reference)	–	100	–	%
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.INV1N/INV1B[1:0] bits = 0x2	–	130	–	%
		CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.INV1N/INV1B[1:0] bits = 0x3	–	300	–	%
Crystal oscillator oscillation stop detector current	I_{OSD1C}	CLGOSC1.OSC1SELCR bit = 0, CLGOSC1.OSDEN bit = 1	–	0.025	0.1	μA
Internal oscillator oscillation start time	t_{sta1I}	CLGOSC1.OSC1SELCR bit = 1	–	–	100	μs
Internal oscillator oscillation frequency	f_{OSC1I}	CLGOSC1.OSC1SELCR bit = 1	31.04	32	32.96	kHz

*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)

OSC1 internal oscillation frequency-temperature characteristic

Typ. value



OSC3 oscillator circuit characteristics

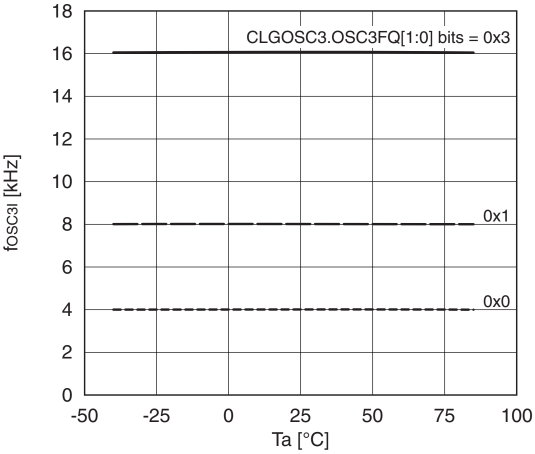
Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = 25°C

Item	Symbol	Condition	Ta	Min.	Typ.	Max.	Unit
Crystal/ceramic oscillator oscillation start time	t_{sta3C}	CLGOSC3.OSC3MD bit = 1, Crystal resonator		–	–	20	ms
		CLGOSC3.OSC3MD bit = 1, Ceramic resonator		–	–	1	ms
Crystal/ceramic oscillator internal gate capacitance	C_{GI3C}	CLGOSC3.OSC3MD bit = 1		–	5	–	pF
Crystal/ceramic oscillator internal drain capacitance	C_{DI3C}	CLGOSC3.OSC3MD bit = 1		–	5	–	pF
Internal oscillator oscillation start time	t_{sta3I}	CLGOSC3.OSC3MD bit = 0		–	–	200	µs
Internal oscillator oscillation frequency	f_{OSC3I}	CLGOSC3.OSC3MD bit = 0, CLGOSC3.OSC3FQ[1:0] bits = 0x3	0 to 85°C	15.84	16	16.16	MHz
			-40 to 0°C	15.76	16	16.24	MHz
		CLGOSC3.OSC3MD bit = 0, CLGOSC3.OSC3FQ[1:0] bits = 0x1	0 to 85°C	7.92	8	8.08	MHz
			-40 to 0°C	7.88	8	8.12	MHz
		CLGOSC3.OSC3MD bit = 0, CLGOSC3.OSC3FQ[1:0] bits = 0x0	0 to 85°C	3.96	4	4.04	MHz
			-40 to 0°C	3.94	4	4.06	MHz
		CLGOSC3.OSC3MD bit = 0, CLGOSC3.OSC3FQ[1:0] bits = 0x3 *1		15.84	16	16.16	MHz

*1 Corrected value immediately after the auto-trimming operation has completed.

OSC3 internal oscillation frequency-temperature characteristic

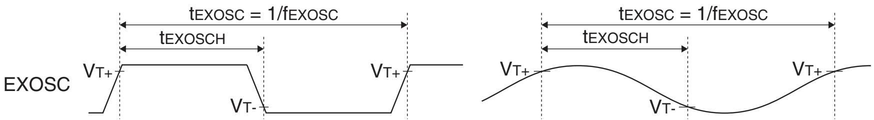
Typ. value



EXOSC external clock input characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
EXOSC external clock duty ratio	t_{EXOSCD}	$t_{EXOSCD} = t_{EXOSCH}/t_{EXOSC}$	46	–	54	%
High level Schmitt input threshold voltage	V_{T+}		0.5 × 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		180	–	–	mV



23.6 Flash Memory Characteristics

Unless otherwise specified: V_{DD} = 2.2 to 5.5 V *1, V_{SS} = 0 V, 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.	1,000	–	–	times

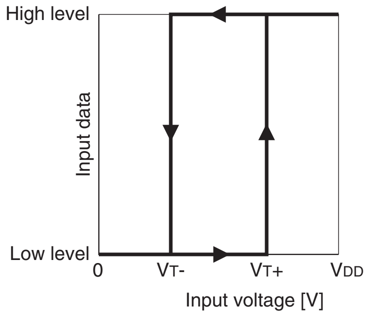
*1 The potential variation of the V_{SS} voltage should be suppressed to within ±0.3 V on the basis of the ground potential of the MCU mounting board while the Flash is being programmed, as it affects the Flash memory characteristics (programming count).

*2 Assumed that Erasing + Programming as count of 1. The count includes programming in the factory for shipment with ROM data programmed.

23.7 Input/Output Port (PPORT) Characteristics

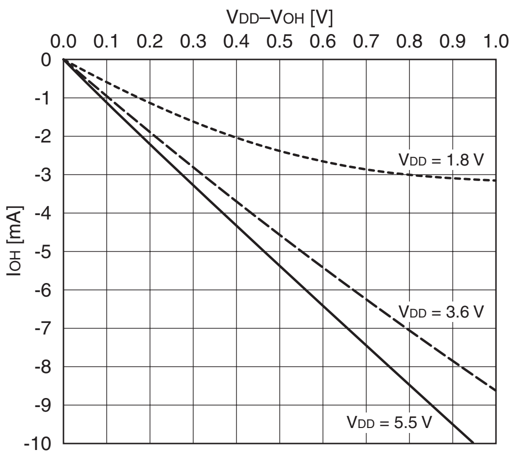
Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

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		180	–	–	mV
High level output current	I_{OH}	V_{OH} = 0.9 × V_{DD}	–	–	-0.5	mA
Low level output current	I_{OL}	V_{OL} = 0.1 × V_{DD}	0.5	–	–	mA
Leakage current	I_{LEAK}		-150	–	150	nA
Input pull-up resistance	R_{INU}		100	200	500	kΩ
Input pull-down resistance	R_{IND}		100	200	500	kΩ
Pin capacitance	C_{IN}		–	–	15	pF



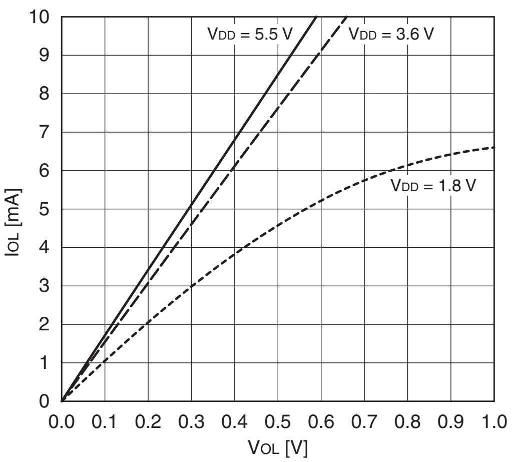
High-level output current characteristic

Ta = 85°C, Max. value



Low-level output current characteristic

Ta = 85°C, Min. value



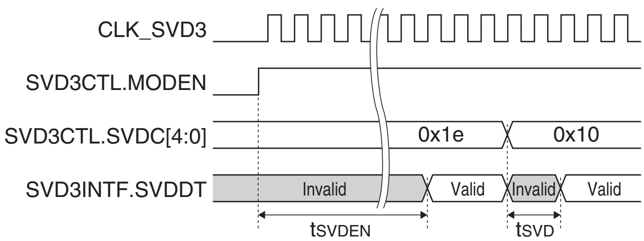
23.8 Supply Voltage Detector (SVD3) Characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
EXSVD*n* pin input voltage range	V_{EXSVD}		0	–	V_{DD}	V
EXSVD*n* input impedance	R_{EXSVD}	SVD3CTL.SVDC[4:0] bits = 0x00	253	279	305	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x01	274	302	330	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x02	317	348	380	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x03	338	371	405	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x04	380	418	456	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x05	421	464	507	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x06	443	487	531	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x07	464	511	557	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x08	486	534	581	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x09	507	557	607	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x0a	528	580	631	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x0b	551	603	655	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x0c	571	626	682	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x0d	593	649	705	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x0e	616	672	727	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x0f	635	695	754	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x10	658	718	777	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x11	679	741	804	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x12	698	765	833	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x13	739	812	885	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x14	761	834	908	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x15	804	880	955	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x16	842	929	1,016	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x17	878	948	1,019	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x18	893	972	1,052	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x19	922	993	1,064	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x1a	963	1,041	1,119	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x1b	982	1,063	1,145	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x1c	1,001	1,086	1,171	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x1d	1,022	1,110	1,198	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x1e	1,054	1,129	1,204	kΩ
		SVD3CTL.SVDC[4:0] bits = 0x1f	1,072	1,154	1,237	kΩ
EXSVD*n* detection voltage	V_{SVD_EXT}	SVD3CTL.SVDC[4:0] bits = 0x00	1.17	1.2	1.23	V
		SVD3CTL.SVDC[4:0] bits = 0x01	1.27	1.3	1.33	V
		SVD3CTL.SVDC[4:0] bits = 0x02	1.46	1.5	1.54	V
		SVD3CTL.SVDC[4:0] bits = 0x03	1.56	1.6	1.64	V
		SVD3CTL.SVDC[4:0] bits = 0x04	1.76	1.8	1.85	V
		SVD3CTL.SVDC[4:0] bits = 0x05	1.95	2.0	2.05	V
		SVD3CTL.SVDC[4:0] bits = 0x06	2.05	2.1	2.15	V
		SVD3CTL.SVDC[4:0] bits = 0x07	2.15	2.2	2.26	V
		SVD3CTL.SVDC[4:0] bits = 0x08	2.24	2.3	2.36	V
		SVD3CTL.SVDC[4:0] bits = 0x09	2.34	2.4	2.46	V
		SVD3CTL.SVDC[4:0] bits = 0x0a	2.44	2.5	2.56	V
		SVD3CTL.SVDC[4:0] bits = 0x0b	2.54	2.6	2.67	V
		SVD3CTL.SVDC[4:0] bits = 0x0c	2.63	2.7	2.77	V
		SVD3CTL.SVDC[4:0] bits = 0x0d	2.73	2.8	2.87	V
		SVD3CTL.SVDC[4:0] bits = 0x0e	2.83	2.9	2.97	V
		SVD3CTL.SVDC[4:0] bits = 0x0f	2.93	3.0	3.08	V
		SVD3CTL.SVDC[4:0] bits = 0x10	3.02	3.1	3.18	V
		SVD3CTL.SVDC[4:0] bits = 0x11	3.12	3.2	3.28	V
		SVD3CTL.SVDC[4:0] bits = 0x12	3.22	3.3	3.38	V
		SVD3CTL.SVDC[4:0] bits = 0x13	3.41	3.5	3.59	V
		SVD3CTL.SVDC[4:0] bits = 0x14	3.51	3.6	3.69	V
		SVD3CTL.SVDC[4:0] bits = 0x15	3.71	3.8	3.90	V
		SVD3CTL.SVDC[4:0] bits = 0x16	3.90	4.0	4.10	V
		SVD3CTL.SVDC[4:0] bits = 0x17	4.00	4.1	4.20	V
		SVD3CTL.SVDC[4:0] bits = 0x18	4.10	4.2	4.31	V
		SVD3CTL.SVDC[4:0] bits = 0x19	4.19	4.3	4.41	V
		SVD3CTL.SVDC[4:0] bits = 0x1a	4.39	4.5	4.61	V
		SVD3CTL.SVDC[4:0] bits = 0x1b	4.49	4.6	4.72	V
		SVD3CTL.SVDC[4:0] bits = 0x1c	4.58	4.7	4.82	V
		SVD3CTL.SVDC[4:0] bits = 0x1d	4.68	4.8	4.92	V
		SVD3CTL.SVDC[4:0] bits = 0x1e	4.78	4.9	5.02	V
		SVD3CTL.SVDC[4:0] bits = 0x1f	4.88	5.0	5.13	V

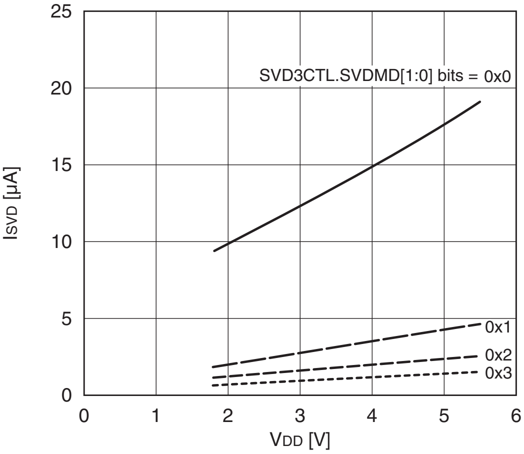
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
SVD detection voltage	V_{SVD}	SVD3CTL.SVDC[4:0] bits = 0x04	1.76	1.8	1.85	V
		SVD3CTL.SVDC[4:0] bits = 0x05	1.95	2.0	2.05	V
		SVD3CTL.SVDC[4:0] bits = 0x06	2.05	2.1	2.15	V
		SVD3CTL.SVDC[4:0] bits = 0x07	2.15	2.2	2.26	V
		SVD3CTL.SVDC[4:0] bits = 0x08	2.24	2.3	2.36	V
		SVD3CTL.SVDC[4:0] bits = 0x09	2.34	2.4	2.46	V
		SVD3CTL.SVDC[4:0] bits = 0x0a	2.44	2.5	2.56	V
		SVD3CTL.SVDC[4:0] bits = 0x0b	2.54	2.6	2.67	V
		SVD3CTL.SVDC[4:0] bits = 0x0c	2.63	2.7	2.77	V
		SVD3CTL.SVDC[4:0] bits = 0x0d	2.73	2.8	2.87	V
		SVD3CTL.SVDC[4:0] bits = 0x0e	2.83	2.9	2.97	V
		SVD3CTL.SVDC[4:0] bits = 0x0f	2.93	3.0	3.08	V
		SVD3CTL.SVDC[4:0] bits = 0x10	3.02	3.1	3.18	V
		SVD3CTL.SVDC[4:0] bits = 0x11	3.12	3.2	3.28	V
		SVD3CTL.SVDC[4:0] bits = 0x12	3.22	3.3	3.38	V
		SVD3CTL.SVDC[4:0] bits = 0x13	3.41	3.5	3.59	V
		SVD3CTL.SVDC[4:0] bits = 0x14	3.51	3.6	3.69	V
		SVD3CTL.SVDC[4:0] bits = 0x15	3.71	3.8	3.90	V
		SVD3CTL.SVDC[4:0] bits = 0x16	3.90	4.0	4.10	V
		SVD3CTL.SVDC[4:0] bits = 0x17	4.00	4.1	4.20	V
		SVD3CTL.SVDC[4:0] bits = 0x18	4.10	4.2	4.31	V
		SVD3CTL.SVDC[4:0] bits = 0x19	4.19	4.3	4.41	V
		SVD3CTL.SVDC[4:0] bits = 0x1a	4.39	4.5	4.61	V
		SVD3CTL.SVDC[4:0] bits = 0x1b	4.49	4.6	4.72	V
		SVD3CTL.SVDC[4:0] bits = 0x1c	4.58	4.7	4.82	V
		SVD3CTL.SVDC[4:0] bits = 0x1d	4.68	4.8	4.92	V
		SVD3CTL.SVDC[4:0] bits = 0x1e	4.78	4.9	5.02	V
		SVD3CTL.SVDC[4:0] bits = 0x1f	4.88	5.0	5.13	V
SVD circuit enable response time	t_{SVDEN}	*1	–	–	500	µs
SVD circuit response time	t_{SVD}		–	–	60	µs
SVD circuit current	I_{SVD}	SVD3CTL.SVDMD[1:0] bits = 0x0, SVD3CTL.SVDC[4:0] bits = 0x04 CLK_SVD3 = 32 kHz, Ta = 25°C	–	19	35	µA
		SVD3CTL.SVDMD[1:0] bits = 0x1, SVD3CTL.SVDC[4:0] bits = 0x04, CLK_SVD3 = 32 kHz, Ta = 25°C	–	4.7	7.7	µA
		SVD3CTL.SVDMD[1:0] bits = 0x2, SVD3CTL.SVDC[4:0] bits = 0x04, CLK_SVD3 = 32 kHz, Ta = 25°C	–	2.5	4.1	µA
		SVD3CTL.SVDMD[1:0] bits = 0x3, SVD3CTL.SVDC[4:0] bits = 0x04, CLK_SVD3 = 32 kHz, Ta = 25°C	–	1.5	2.4	µA

*1 If CLK_SVD3 is configured in the neighborhood of 32 kHz, the SVD3INTF.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, SVD3CTL.SVDC[4:0] bits = 0x04, CLK_SVD3 = 32 kHz, Typ. value



23.9 UART (UART3) Characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Transfer baud rate	U_{BRT1}	Normal mode	150	–	460,800	bps
	U_{BRT2}	IrDA mode	150	–	115,200	bps

23.10 Synchronous Serial Interface (SPIA) Characteristics

Master mode

Unless otherwise specified: V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	V_{DD}	V_{D1} output	Min.	Typ.	Max.	単位
SPICLK0 cycle time	t_{SCYC}		1.8 to 5.5 V	mode0	250	–	–	ns
			1.8 to 3.6 V	mode1	1,000	–	–	ns
SPICLK0 High pulse width	t_{SCKH}		1.8 to 5.5 V	mode0	100	–	–	ns
			1.8 to 3.6 V	mode1	400	–	–	ns
SPICLK0 Low pulse width	t_{SCKL}		1.8 to 5.5 V	mode0	100	–	–	ns
			1.8 to 3.6 V	mode1	400	–	–	ns
SDI0 setup time	t_{SDS}		1.8 to 5.5 V	mode0	90	–	–	ns
			1.8 to 3.6 V	mode1	275	–	–	ns
SDI0 hold time	t_{SDH}		1.8 to 5.5 V	mode0	10	–	–	ns
			1.8 to 3.6 V	mode1	40	–	–	ns
SDO0 output delay time	t_{SDO}	C_L = 15 pF *1	1.8 to 5.5 V	mode0	–	–	40	ns
			1.8 to 3.6 V	mode1	–	–	135	ns

*1 C_L = Pin load

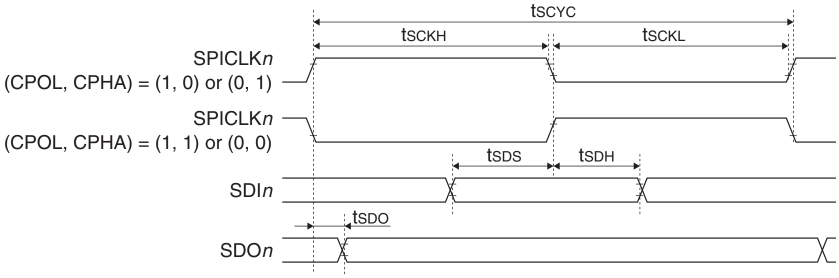
Slave mode

Unless otherwise specified: V_{SS} = 0 V, Ta = -40 to 85°C

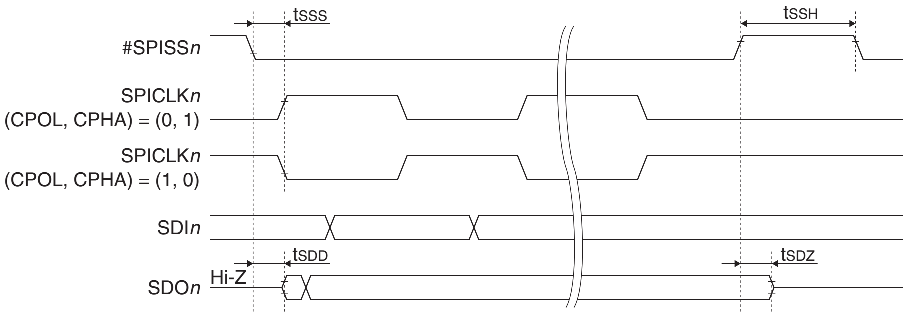
Item	Symbol	Condition	V_{DD}	V_{D1} output	Min.	Typ.	Max.	単位
SPICLK0 cycle time	t_{SCYC}		1.8 to 5.5 V	mode0	250	–	–	ns
			1.8 to 3.6 V	mode1	1,000	–	–	ns
SPICLK0 High pulse width	t_{SCKH}		1.8 to 5.5 V	mode0	100	–	–	ns
			1.8 to 3.6 V	mode1	400	–	–	ns
SPICLK0 Low pulse width	t_{SCKL}		1.8 to 5.5 V	mode0	100	–	–	ns
			1.8 to 3.6 V	mode1	400	–	–	ns
SDI0 setup time	t_{SDS}		1.8 to 5.5 V	mode0	20	–	–	ns
			1.8 to 3.6 V	mode1	60	–	–	ns
SDI0 hold time	t_{SDH}		1.8 to 5.5 V	mode0	25	–	–	ns
			1.8 to 3.6 V	mode1	120	–	–	ns
SDO0 output delay time	t_{SDO}	C_L = 15 pF *1	1.8 to 5.5 V	mode0	–	–	100	ns
			1.8 to 3.6 V	mode1	–	–	360	ns
#SPISS0 setup time	t_{SSS}		1.8 to 5.5 V	mode0	20	–	–	ns
			1.8 to 3.6 V	mode1	60	–	–	ns
#SPISS0 High pulse width	t_{SSH}		1.8 to 5.5 V	mode0	100	–	–	ns
			1.8 to 3.6 V	mode1	400	–	–	ns
SDO0 output start time	t_{SDD}	C_L = 15 pF *1	1.8 to 5.5 V	mode0	–	–	100	ns
			1.8 to 3.6 V	mode1	–	–	360	ns
SDO0 output stop time	t_{SDZ}	C_L = 15 pF *1	1.8 to 5.5 V	mode0	–	–	100	ns
			1.8 to 3.6 V	mode1	–	–	360	ns

*1 C_L = Pin load

Master and slave modes



Slave mode



23.11 Quad Synchronous Serial Interface (QSPI) Characteristics

Master mode

Unless otherwise specified: V_{DDQSPI} = 3.0 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	V_{D1} output	Min.	Typ.	Max.	単位
QSPICLK*n* cycle time	t_{SCYC}		mode0	125	–	–	ns
			mode1	500	–	–	ns
QSPICLK*n* High pulse width	t_{SCKH}		mode0	50	–	–	ns
			mode1	200	–	–	ns
QSPICLK*n* Low pulse width	t_{SCKL}		mode0	50	–	–	ns
			mode1	200	–	–	ns
QSDIO*n*[3:0] setup time	t_{SDS}		mode0	35	–	–	ns
			mode1	120	–	–	ns
QSDIO*n*[3:0] hold time	t_{SDH}		mode0	10	–	–	ns
			mode1	40	–	–	ns
QSDIO*n*[3:0] output delay time	t_{SDO}	C_L = 15 pF *1	mode0	–	–	35	ns
			mode1	–	–	120	ns

*1 C_L = Pin load

Slave mode

Unless otherwise specified: V_{DDQSPI} = 3.0 to 3.6 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	V_{D1} output	Min.	Typ.	Max.	単位
QSPICLK*n* cycle time	t_{SCYC}		mode0	150	–	–	ns
			mode1	500	–	–	ns
QSPICLK*n* High pulse width	t_{SCKH}		mode0	60	–	–	ns
			mode1	200	–	–	ns
QSPICLK*n* Low pulse width	t_{SCKL}		mode0	60	–	–	ns
			mode1	200	–	–	ns
QSDIO*n*[3:0] setup time	t_{SDS}		mode0	10	–	–	ns
			mode1	30	–	–	ns
QSDIO*n*[3:0] hold time	t_{SDH}		mode0	10	–	–	ns
			mode1	50	–	–	ns
QSDIO*n*[3:0] output delay time	t_{SDO}	C_L = 15 pF *1	mode0	–	–	60	ns
			mode1	–	–	220	ns
#QSPISS*n* setup time	t_{SSS}		mode0	10	–	–	ns
			mode1	30	–	–	ns
#QSPISS*n* High pulse width	t_{SSH}		mode0	60	–	–	ns
			mode1	200	–	–	ns
QSDIO*n*[3:0] output start time	t_{SDD}	C_L = 15 pF *1	mode0	–	–	60	ns
			mode1	–	–	220	ns
QSDIO*n*[3:0] output stop time	t_{SDZ}	C_L = 15 pF *1	mode0	–	–	60	ns
			mode1	–	–	220	ns

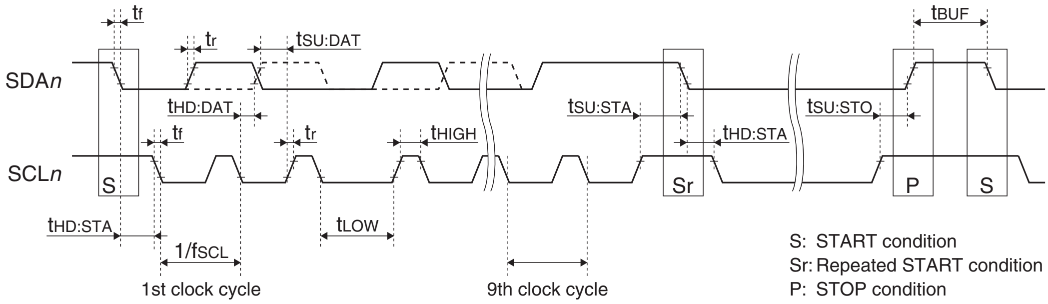
*1 C_L = Pin load

23.12 I^2C (I2C) Characteristics

Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C

Item	Symbol	Condition	Standard mode			Fast mode			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
SCL*n* frequency	f_{SCL}		0	–	100	0	–	400	kHz
Hold time (repeated) START condition *	$t_{HD:STA}$		4.0	–	–	0.6	–	–	µs
SCL*n* Low pulse width	t_{LOW}		4.7	–	–	1.3	–	–	µs
SCL*n* High pulse width	t_{HIGH}		4.0	–	–	0.6	–	–	µs
Repeated START condition setup time	$t_{SU:STA}$		4.7	–	–	0.6	–	–	µs
Data hold time	$t_{HD:DAT}$		0	–	–	0	–	–	µs
Data setup time	$t_{SU:DAT}$		250	–	–	100	–	–	ns
SDA*n*, SCL*n* rise time	t_r		–	–	1,000	–	–	300	ns
SDA*n*, SCL*n* fall time	t_f		–	–	300	–	–	300	ns
STOP condition setup time	$t_{SU:STO}$		4.0	–	–	0.6	–	–	µs
Bus free time	t_{BUF}		4.7	–	–	1.3	–	–	µs

* After this period, the first clock pulse is generated.



23.13 12-bit A/D Converter (ADC12A) Characteristics

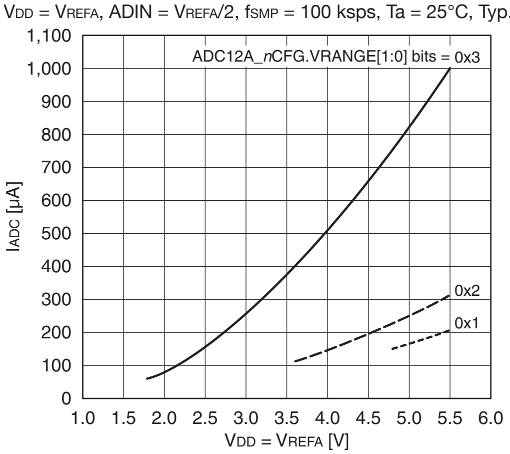
Unless otherwise specified: V_{DD} = 2.5 to 5.5 V, VREFA*n* = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 85°C, ADC12A_*n*TRG.SMPCLK[2:0] bits = 0x3 (7cycles)

Item	Symbol	Condition	V_{DD}	Min.	Typ.	Max.	Unit
VREFA*n* voltage range	V_{REFA}			1.8	–	V_{DD}	V
A/D conversion clock frequency	f_{CLK_ADC12A}			16	–	2,200	kHz
Sampling rate *1	f_{SMP}			–	–	100	ksps
Integral nonlinearity *2	INL	V_{DD} = VREFA*n* *3		–	–	±3	LSB
Differential nonlinearity	DNL	V_{DD} = VREFA*n* *3		–	–	±3	LSB
Zero-scale error	ZSE	V_{DD} = VREFA*n* *3		–	–	±5	LSB
Full-scale error	FSE	V_{DD} = VREFA*n* *3		–	–	±5	LSB
Analog input resistance	R_{ADIN}			–	–	4	kΩ
Analog input capacitance	C_{ADIN}			–	–	30	pF
A/D converter circuit current	I_{ADC}	ADC12A_*n*CFG.VRANGE[1:0] bits = 0x3, V_{DD} = V_{REFA}, ADIN = V_{REFA}/2, f_{SMP} = 100 ksps, Ta = 25°C	3.6 V	–	400	700	μA
		ADC12A_*n*CFG.VRANGE[1:0] bits = 0x2, V_{DD} = V_{REFA}, ADIN = V_{REFA}/2, f_{SMP} = 100 ksps, Ta = 25°C	4.8 V	–	230	470	μA
		ADC12A_*n*CFG.VRANGE[1:0] bits = 0x1, V_{DD} = V_{REFA}, ADIN = V_{REFA}/2, f_{SMP} = 100 ksps, Ta = 25°C	5.5 V	–	210	390	μA

*1 The Max. value is the value when the A/D conversion clock frequency f_{CLK_ADC12A} = 2,000 kHz.
*2 Integral nonlinearity is measured at the end point line.
*3 The error will be increased according to the potential difference between V_{DD} and VREFA*n*.

A/D converter current consumption-power supply voltage characteristic

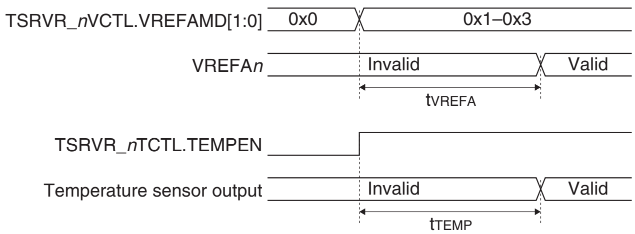
V_{DD} = V_{REFA}, ADIN = V_{REFA}/2, f_{SMP} = 100 ksps, Ta = 25°C, Typ. value



23.14 Temperature Sensor/Reference Voltage Generator (TSRVR) Characteristics

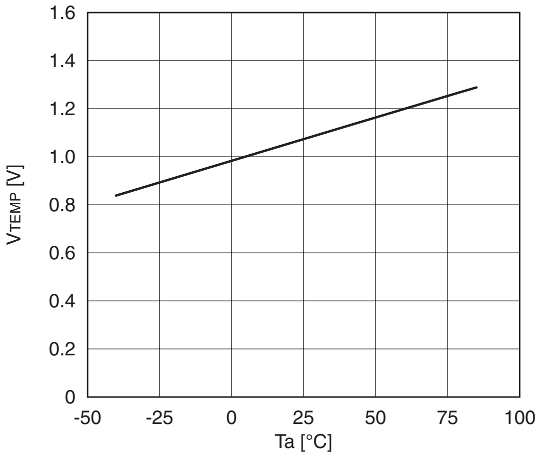
Unless otherwise specified: V_{DD} = 1.8 to 5.5 V, V_{SS} = 0 V, Ta = -40 to 105°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
V_{REFA} (2.5 V) output voltage	V_{VO25}	V_{DD} = 2.7 to 5.5 V	2.4	2.5	2.6	V
V_{REFA} (2.0 V) output voltage	V_{VO20}	V_{DD} = 2.2 to 5.5 V	1.9	2.0	2.1	V
V_{REFA} (V_{DD}) output voltage	V_{VODD}	V_{DD} = 1.8 to 5.5 V	V_{DD} - 0.1	V_{DD}	V_{DD} + 0.1	V
V_{REFA} (2.5/2.0 V) operating current	I_{VO1}	V_{DD} = 5.5 V, Ta = 25°C	25	40	60	µA
V_{REFA} (V_{DD}) operating current	I_{VO2}	V_{DD} = 5.5 V, Ta = 25°C	–	0	0.1	µA
V_{REFA} output voltage stabilization time	t_{VREFA}	C_{VREFA} = 0.1 µF	–	1.5	5	ms
Temperature sensor output voltage	V_{TEMP}	V_{DD} = 2.2 to 5.5 V, Ta = 25°C	1.04	1.07	1.10	V
Temperature sensor output voltage temperature coefficient	ΔV_{TEMP}	V_{DD} = 2.2 to 5.5 V	–	3.6 ± 3%	3.6 ± 7%	mV/°C
Temperature sensor operating current	I_{VTEMP}	V_{DD} = 5.5 V, Ta = 25°C	10	16	22	µA
Temperature sensor output stabilization time	t_{TEMP}		–	–	200	µs



Temperature sensor output voltage-temperature characteristic

V_{DD} = 2.2 to 5.5 V, Typ. value



23.15 R/F Converter (RFC) Characteristics

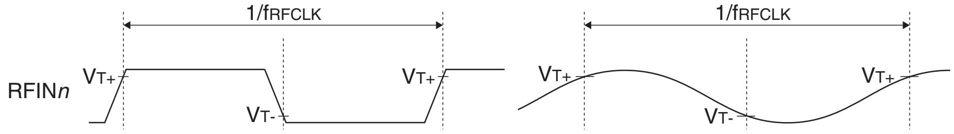
R/F converter characteristics change depending on conditions (board pattern, components used, etc.). Use these characteristic values as a reference and perform evaluation using the actual printed circuit board.

Unless otherwise specified: VDD = 1.8 to 5.5 V, VSS = 0 V, Ta = -40 to 85 °C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Reference/sensor oscillation frequency	fRFCLK		1	–	1,000	kHz
Reference/sensor oscillation frequency IC deviation	ΔfRFCLK/ΔIC	Ta = 25 °C *1	-40	–	40	%
Reference resistor/resistive sensor resistance	RREF, RSEN		10	–	-	kΩ
Reference capacitance	CREF		100	–	–	pF
Time base counter clock frequency	fTCCLK		–	–	4.2	MHz
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
R/F converter operating current	IRFC	CREF = 1,000 pF, RREF/RSEN = 100 kΩ, Ta = 25 °C, VDD = 3.6 V	–	200	350	μA

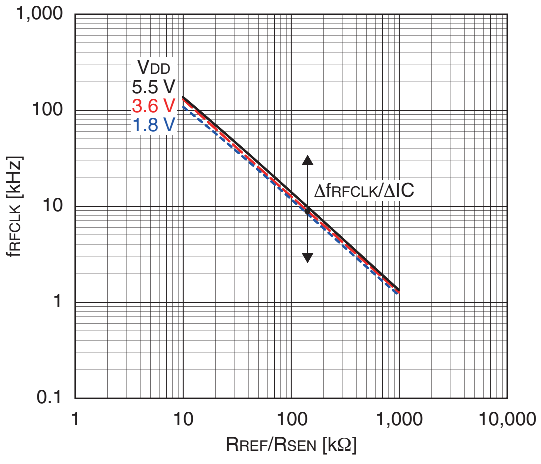
*1 In this characteristic, unevenness between production lots, and variations in measurement board, resistances and capacitances are taken into account.

Waveforms for external clock input mode



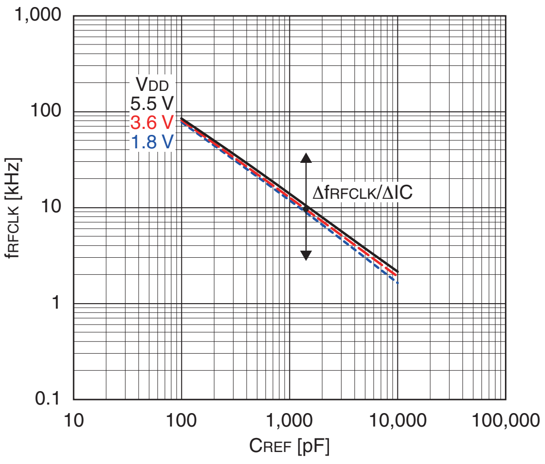
RFC reference/sensor oscillation frequency-resistance characteristic

CREF = 1,000 pF, Ta = 25 °C, Typ. value



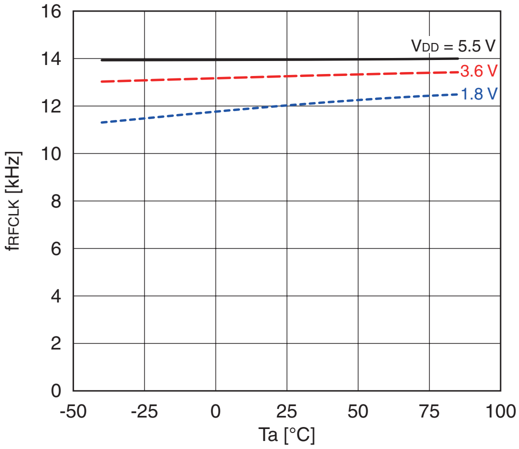
RFC reference/sensor oscillation frequency-capacitance characteristic

RREF/RSEN = 100 kΩ, Ta = 25 °C, Typ. value



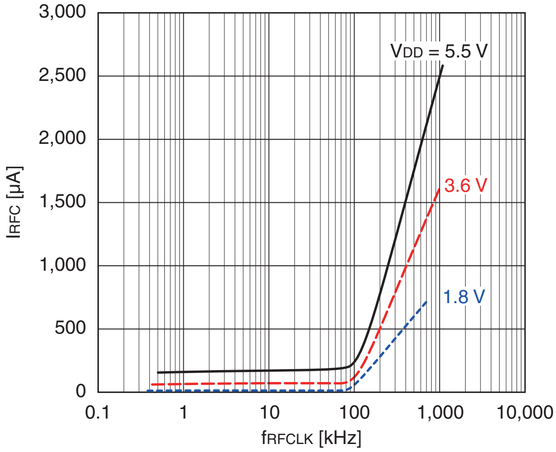
RFC reference/sensor oscillation frequency-temperature characteristic

R_{REF}/R_{SEN} = 100 kΩ, C_{REF} = 1,000 pF, Typ. value

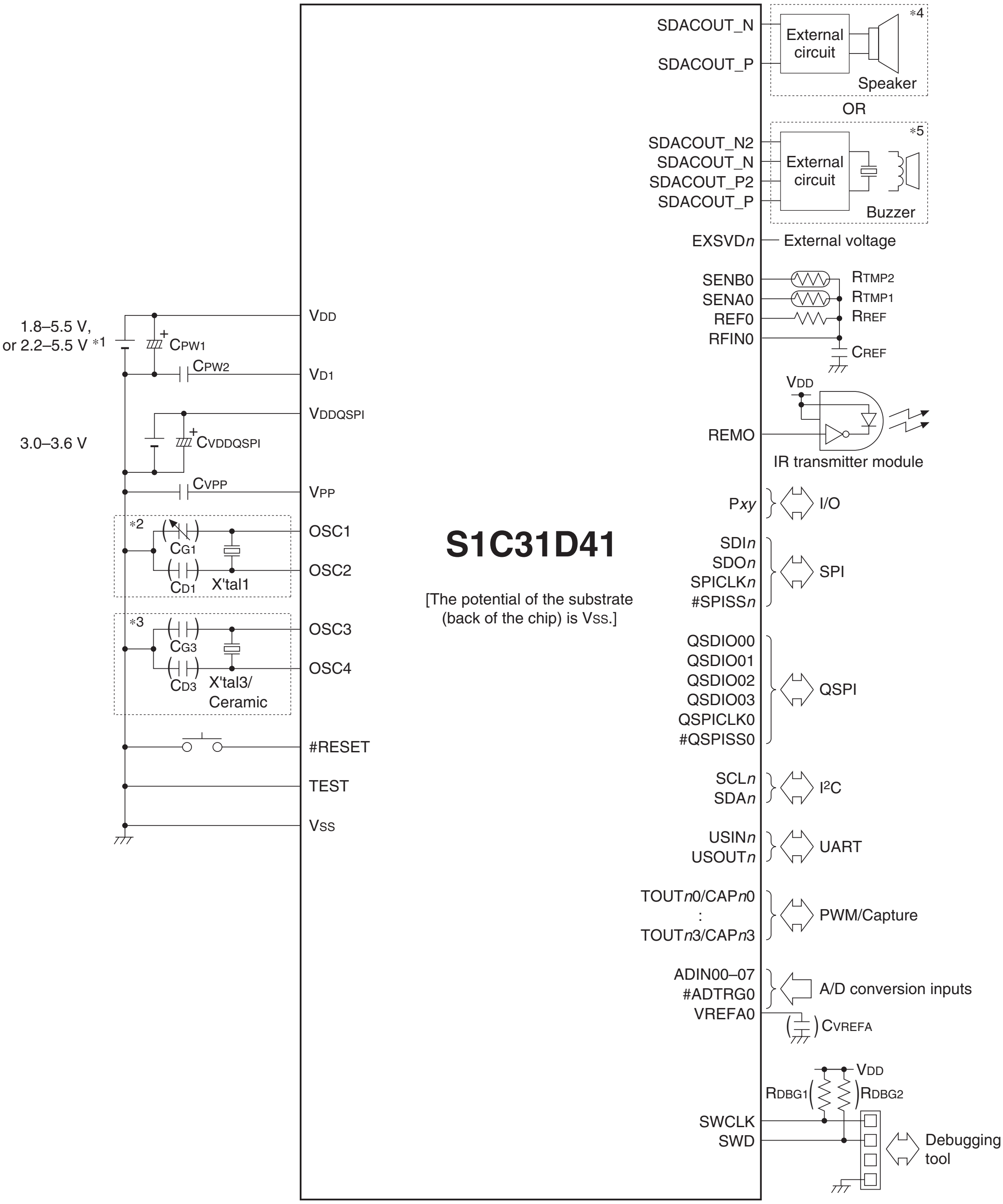


RFC reference/sensor oscillation current consumption-frequency characteristic

C_{REF} = 1,000 pF, Ta = 25 °C, Typ. value



24 Basic External Connection Diagram



*1: For Flash programming
*2: When OSC1 crystal oscillator is selected
*3: When OSC3 crystal/ceramic oscillator is selected
*4: Two-pin output mode
*5: Four-pin output mode
( ): 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	FA-238V (16 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
C_{PW1}	Bypass capacitor between V_{SS} and V_{DD}	Ceramic capacitor or electrolytic capacitor
C_{PW2}	Capacitor between V_{SS} and V_{D1}	Ceramic capacitor
$C_{VDDQSPI}$	Capacitor between V_{SS} and V_{DDQSPI}	Ceramic capacitor or electrolytic capacitor
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_{VREFA}	Capacitor between V_{SS} and VREFA	Ceramic capacitor
R_{DBG1-2}	Debug pin pull-up resistor	Thick film chip resistor
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. However, the final values should be determined after evaluating operations using an actual mounting board.

25 Package

TQFP12-32PIN (P-TQFP032-0707-0.80)

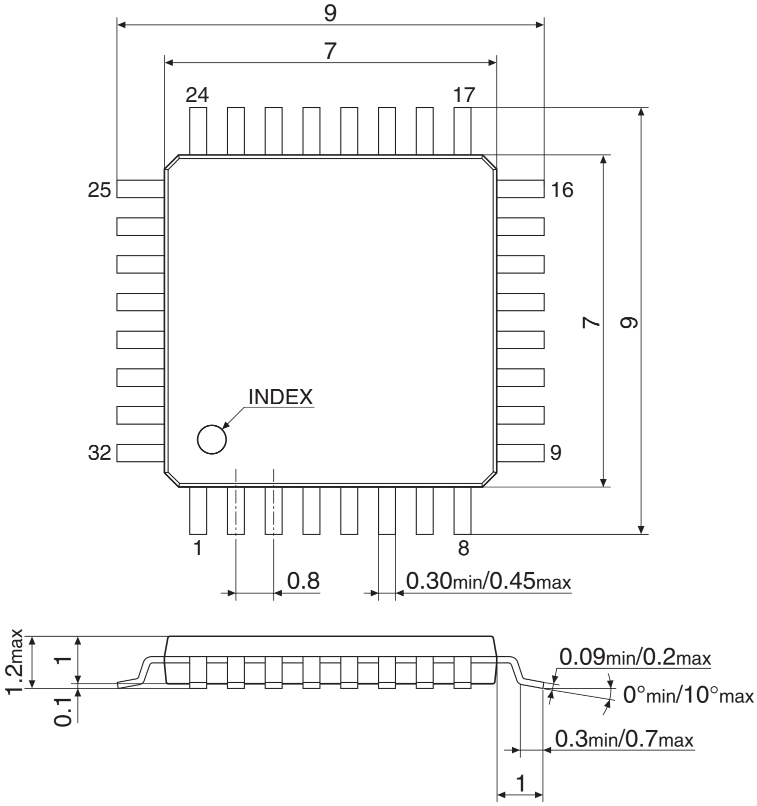


Figure 25.1 TQFP12-32PIN Package Dimensions

TQFP12-48PIN (P-TQFP048-0707-0.50)

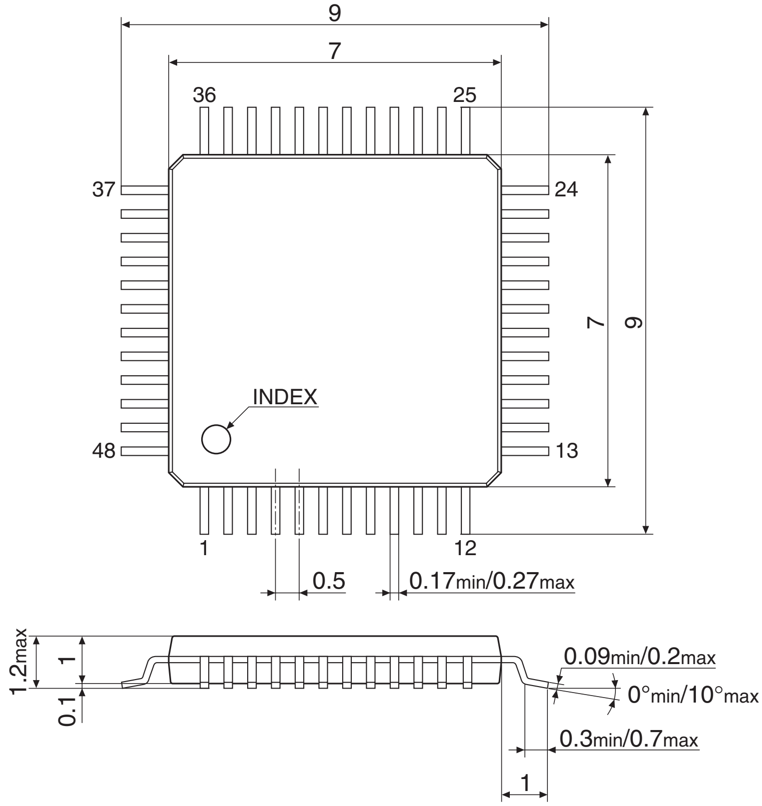


Figure 25.2 TQFP12-48PIN Package Dimensions

QFP13-64PIN (P-LQFP064-1010-0.50)

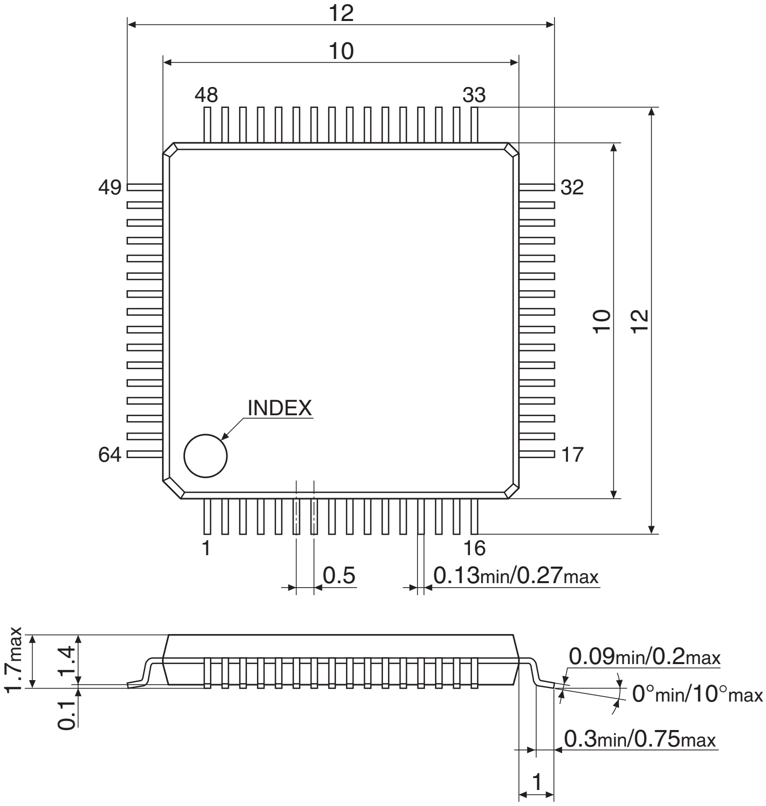


Figure 25.3 QFP13-64PIN Package Dimensions

Appendix A List of Peripheral Circuit Control Registers

0x0020 0000 System Register (SYS)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0000	SYSPROT (System Protect Register)	15–0	PROT[15:0]	0x0000	H0	R/W	–

0x0020 0020 Power Generator (PWGA)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0020	PWGACTL (PWGA Control Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5	REGDIS	0	H0	R/WP	
		4	REGSEL	1	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	REGMODE[1:0]	0x0	H0	R/WP	

0x0020 0040–0x0020 005a Clock Generator (CLG)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0040	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]	0x2	H0	R/WP	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/WP	
0x0020 0042	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	
0x0020 0044	CLGIOSC (CLG IOSC Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1–0	IOSCFQ[1:0]	0x2	H0	R/WP	
0x0020 0046	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	OSC1SELCR	0	H0	R/WP	
		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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0048	CLGOSC3 (CLG OSC3 Control Register)	15–12	–	0x0	–	R	–
		11–10	OSC3FQ[1:0]	0x1	H0	R/WP	
		9	OSC3MD	0	H0	R/WP	
		8	–	0	–	R	
		7–6	–	0x0	–	R	
		5–4	OSC3INV[1:0]	0x3	H0	R/WP	
		3	OSC3STM	0	H0	R/WP	
		2–0	OSC3WT[2:0]	0x6	H0	R/WP	
0x0020 004c	CLGINTF (CLG Interrupt Flag Register)	15–9	–	0x00	–	R	–
		8	OSC3TERIF	0	H0	R/W	Cleared by writing 1.
		7	–	0	–	R	–
		6	(reserved)	0	H0	R	
		5	OSC1STPIF	0	H0	R/W	Cleared by writing 1.
		4	OSC3TEDIF	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	
0x0020 004e	CLGINTE (CLG Interrupt Enable Register)	15–9	–	0x00	–	R	–
		8	OSC3TERIE	0	H0	R/W	
		7	–	0	–	R	
		6	(reserved)	0	H0	R/W	
		5	OSC1STPIE	0	H0	R/W	
		4	OSC3TEDIE	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	
0x0020 0050	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	
0x0020 0052	CLGTRIM1 (CLG Oscillation Frequency Trimming Register 1)	15–14	–	0x0	–	R	–
		13–8	IOSCLSAJ[5:0]	*	H0	R/WP	* Determined by factory adjustment.
		7–6	–	0x0	–	R	–
		5–0	IOSCHSAJ[5:0]	*	H0	R/WP	* Determined by factory adjustment.
0x0020 0054	CLGTRIM2 (CLG Oscillation Frequency Trimming Register 2)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	R	
		5–0	OSC1SAJ[5:0]	*	H0	R/WP	* Determined by factory adjustment.
0x0020 005a	CLGTRIM3 (CLG Oscillation Frequency Trimming Register 3)	15–9	–	0x00	–	R	–
		8–0	OSC3SAJ[8:0]	*	H0	R/WP	* Determined by factory adjustment.

0x0020 0080 Cache Controller (CACHE)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0080	CACHECTL (CACHE Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	–	1	–	R	
		0	CACHEEN	0	H0	R/W	

0x0020 00a0–0x0020 00a4 Watchdog Timer (WDT2)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 00a0	WDT2CLK (WDT2 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	
0x0020 00a2	WDT2CTL (WDT2 Control Register)	15–11	–	0x00	–	R	–
		10–9	MOD[1:0]	0x0	H0	R/WP	
		8	STATNMI	0	H0	R	
		7–5	–	0x0	–	R	
		4	WDTCNTRST	0	H0	WP	Always read as 0.
		3–0	WDTRUN[3:0]	0xa	H0	R/WP	–
0x0020 00a4	WDT2CMP (WDT2 Counter Compare Match Register)	15–10	–	0x00	–	R	–
		9–0	CMP[9:0]	0x3ff	H0	R/WP	

0x0020 00c0–0x0020 00d2 Real-time Clock (RTCA)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 00c0	RTCACTLL (RTCA Control Register (Low Byte))	7	–	0	–	R	–
		6	RTCBSY	0	H0	R	
		5	RTCHLD	0	H0	R/W	Cleared by setting the RTCACTLL.RTCRST bit to 1.
		4	RTC24H	0	H0	R/W	–
		3	–	0	–	R	
		2	RTCADJ	0	H0	R/W	Cleared by setting the RTCACTLL.RTCRST bit to 1.
		1	RTCRST	0	H0	R/W	–
		0	RTCRUN	0	H0	R/W	
0x0020 00c1	RTCACTLH (RTCA Control Register (High Byte))	7	RTCTRMBSY	0	H0	R	–
		6–0	RTCTRM[6:0]	0x00	H0	W	Read as 0x00.
0x0020 00c2	RTCAALM1 (RTCA 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	
0x0020 00c4	RTCAALM2 (RTCA 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	
0x0020 00c6	RTCASWCTL (RTCA 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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 00c8	RTCASEC (RTCA 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 RTCACTLL.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	
0x0020 00ca	RTCAHUR (RTCA 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	
0x0020 00cc	RTCAMON (RTCA 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	
0x0020 00ce	RTCAYAR (RTCA 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	
0x0020 00d0	RTCAINTF (RTCA 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	T1DAYIF	0	H0	R/W	
		6	T1HURIF	0	H0	R/W	
		5	T1MINIF	0	H0	R/W	
		4	T1SECIF	0	H0	R/W	
		3	T1_2SECIF	0	H0	R/W	
		2	T1_4SECIF	0	H0	R/W	
		1	T1_8SECIF	0	H0	R/W	
		0	T1_32SECIF	0	H0	R/W	
0x0020 00d2	RTCAINTE (RTCA 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	T1DAYIE	0	H0	R/W	
		6	T1HURIE	0	H0	R/W	
		5	T1MINIE	0	H0	R/W	
		4	T1SECIE	0	H0	R/W	
		3	T1_2SECIE	0	H0	R/W	
		2	T1_4SECIE	0	H0	R/W	
		1	T1_8SECIE	0	H0	R/W	
		0	T1_32SECIE	0	H0	R/W	

0x0020 0100–0x0020 0106 Supply Voltage Detector (SVD3)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0100	SVD3CLK (SVD3 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	
0x0020 0102	SVD3CTL (SVD3 Control Register)	15	VDSEL	0	H1	R/WP	–
		14–13	SVDSC[1:0]	0x0	H0	R/WP	Writing takes effect when the SVD3CTL.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	EXSEL	0	H1	R/WP	
		2–1	SVDMD[1:0]	0x0	H0	R/WP	
		0	MODEN	0	H1	R/WP	
0x0020 0104	SVD3INTF (SVD3 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.
0x0020 0106	SVD3INTE (SVD3 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	SVDIE	0	H0	R/W	

0x0020 0160–0x0020 016c 16-bit Timer (T16) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0160	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	
0x0020 0162	T16_0MOD (T16 Ch.0 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 0164	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	
0x0020 0166	T16_0TR (T16 Ch.0 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 0168	T16_0TC (T16 Ch.0 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 016a	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.
0x0020 016c	T16_0INTE (T16 Ch.0 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 01b0 Flash Controller (FLASHC)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 01b0	FLASHCWAIT (FLASHC Flash Read Cycle Register)	15–9	–	0x00	–	R	–
		8	(reserved)	0	H0	R/WP	
		7–2	–	0x00	–	R	
		1–0	RDWAIT[1:0]	0x1	H0	R/WP	

0x0020 0200–0x0020 02e2 I/O Ports (PPORT)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0200	P0DAT (P0 Port Data Register)	15	P0OUT7	0	H0	R/W	–	–	–	✓
		14	P0OUT6	0	H0	R/W		–	✓	✓
		13	P0OUT5	0	H0	R/W		–	✓	✓
		12	P0OUT4	0	H0	R/W		–	✓	✓
		11	P0OUT3	0	H0	R/W		–	✓	✓
		10	P0OUT2	0	H0	R/W		–	✓	✓
		9	P0OUT1	0	H0	R/W		–	–	✓
		8	P0OUT0	0	H0	R/W		–	–	✓
		7	P0IN7	0	H0	R	–	–	–	✓
		6	P0IN6	0	H0	R		–	✓	✓
		5	P0IN5	0	H0	R		–	✓	✓
		4	P0IN4	0	H0	R		–	✓	✓
		3	P0IN3	0	H0	R		–	✓	✓
		2	P0IN2	0	H0	R		–	✓	✓
		1	P0IN1	0	H0	R		–	–	✓
		0	P0IN0	0	H0	R		–	–	✓
0x0020 0202	P0IOEN (P0 Port Enable Register)	15	P0IEN7	0	H0	R/W	–	–	–	✓
		14	P0IEN6	0	H0	R/W		–	✓	✓
		13	P0IEN5	0	H0	R/W		–	✓	✓
		12	P0IEN4	0	H0	R/W		–	✓	✓
		11	P0IEN3	0	H0	R/W		–	✓	✓
		10	P0IEN2	0	H0	R/W		–	✓	✓
		9	P0IEN1	0	H0	R/W		–	–	✓
		8	P0IEN0	0	H0	R/W		–	–	✓
		7	P0OEN7	0	H0	R/W	–	–	–	✓
		6	P0OEN6	0	H0	R/W		–	✓	✓
		5	P0OEN5	0	H0	R/W		–	✓	✓
		4	P0OEN4	0	H0	R/W		–	✓	✓
		3	P0OEN3	0	H0	R/W		–	✓	✓
		2	P0OEN2	0	H0	R/W		–	✓	✓
		1	P0OEN1	0	H0	R/W		–	–	✓
		0	P0OEN0	0	H0	R/W		–	–	✓
0x0020 0204	P0RCTL (P0 Port Pull-up/down Control Register)	15	P0PDPU7	0	H0	R/W	–	–	–	✓
		14	P0PDPU6	0	H0	R/W		–	✓	✓
		13	P0PDPU5	0	H0	R/W		–	✓	✓
		12	P0PDPU4	0	H0	R/W		–	✓	✓
		11	P0PDPU3	0	H0	R/W		–	✓	✓
		10	P0PDPU2	0	H0	R/W		–	✓	✓
		9	P0PDPU1	0	H0	R/W		–	–	✓
		8	P0PDPU0	0	H0	R/W		–	–	✓
		7	P0REN7	0	H0	R/W	–	–	–	✓
		6	P0REN6	0	H0	R/W		–	✓	✓
		5	P0REN5	0	H0	R/W		–	✓	✓
		4	P0REN4	0	H0	R/W		–	✓	✓
		3	P0REN3	0	H0	R/W		–	✓	✓
		2	P0REN2	0	H0	R/W		–	✓	✓
		1	P0REN1	0	H0	R/W		–	–	✓
		0	P0REN0	0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0206	P0INTF (P0 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P0IF7	0	H0	R/W	Cleared by writing 1.	–	–	✓
		6	P0IF6	0	H0	R/W		–	✓	✓
		5	P0IF5	0	H0	R/W		–	✓	✓
		4	P0IF4	0	H0	R/W		–	✓	✓
		3	P0IF3	0	H0	R/W		–	✓	✓
		2	P0IF2	0	H0	R/W		–	✓	✓
		1	P0IF1	0	H0	R/W		–	–	✓
		0	P0IF0	0	H0	R/W		–	–	✓
0x0020 0208	P0INTCTL (P0 Port Interrupt Control Register)	15	P0EDGE7	0	H0	R/W	–	–	–	✓
		14	P0EDGE6	0	H0	R/W		–	✓	✓
		13	P0EDGE5	0	H0	R/W		–	✓	✓
		12	P0EDGE4	0	H0	R/W		–	✓	✓
		11	P0EDGE3	0	H0	R/W		–	✓	✓
		10	P0EDGE2	0	H0	R/W		–	✓	✓
		9	P0EDGE1	0	H0	R/W		–	–	✓
		8	P0EDGE0	0	H0	R/W		–	–	✓
		7	P0IE7	0	H0	R/W	–	–	–	✓
		6	P0IE6	0	H0	R/W		–	✓	✓
		5	P0IE5	0	H0	R/W		–	✓	✓
		4	P0IE4	0	H0	R/W		–	✓	✓
		3	P0IE3	0	H0	R/W		–	✓	✓
		2	P0IE2	0	H0	R/W		–	✓	✓
		1	P0IE1	0	H0	R/W		–	–	✓
		0	P0IE0	0	H0	R/W		–	–	✓
0x0020 020a	P0CHATEN (P0 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P0CHATEN7	0	H0	R/W	–	–	–	✓
		6	P0CHATEN6	0	H0	R/W		–	✓	✓
		5	P0CHATEN5	0	H0	R/W		–	✓	✓
		4	P0CHATEN4	0	H0	R/W		–	✓	✓
		3	P0CHATEN3	0	H0	R/W		–	✓	✓
		2	P0CHATEN2	0	H0	R/W		–	✓	✓
		1	P0CHATEN1	0	H0	R/W		–	–	✓
		0	P0CHATEN0	0	H0	R/W		–	–	✓
0x0020 020c	P0MODSEL (P0 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P0SEL7	0	H0	R/W	–	–	–	✓
		6	P0SEL6	0	H0	R/W		–	✓	✓
		5	P0SEL5	0	H0	R/W		–	✓	✓
		4	P0SEL4	0	H0	R/W		–	✓	✓
		3	P0SEL3	0	H0	R/W		–	✓	✓
		2	P0SEL2	0	H0	R/W		–	✓	✓
		1	P0SEL1	0	H0	R/W		–	–	✓
		0	P0SEL0	0	H0	R/W		–	–	✓
0x0020 020e	P0FNCSEL (P0 Port Function Select Register)	15–14	P07MUX[1:0]	0x0	H0	R/W	–	–	–	✓
		13–12	P06MUX[1:0]	0x0	H0	R/W		–	✓	✓
		11–10	P05MUX[1:0]	0x0	H0	R/W		–	✓	✓
		9–8	P04MUX[1:0]	0x0	H0	R/W		–	✓	✓
		7–6	P03MUX[1:0]	0x0	H0	R/W		–	✓	✓
		5–4	P02MUX[1:0]	0x0	H0	R/W		–	✓	✓
		3–2	P01MUX[1:0]	0x0	H0	R/W		–	–	✓
		1–0	P00MUX[1:0]	0x0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0210	P1DAT (P1 Port Data Register)	15	P1OUT7	0	H0	R/W	–	✓	✓	✓
		14	P1OUT6	0	H0	R/W		✓	✓	✓
		13	P1OUT5	0	H0	R/W		✓	✓	✓
		12	P1OUT4	0	H0	R/W		✓	✓	✓
		11	P1OUT3	0	H0	R/W		✓	✓	✓
		10	P1OUT2	0	H0	R/W		✓	✓	✓
		9	P1OUT1	0	H0	R/W		–	–	✓
		8	P1OUT0	0	H0	R/W		–	–	✓
		7	P1IN7	0	H0	R	–	✓	✓	✓
		6	P1IN6	0	H0	R		✓	✓	✓
		5	P1IN5	0	H0	R		✓	✓	✓
		4	P1IN4	0	H0	R		✓	✓	✓
		3	P1IN3	0	H0	R		✓	✓	✓
		2	P1IN2	0	H0	R		✓	✓	✓
		1	P1IN1	0	H0	R		–	–	✓
		0	P1IN0	0	H0	R		–	–	✓
0x0020 0212	P1IOEN (P1 Port Enable Register)	15	P1IEN7	0	H0	R/W	–	✓	✓	✓
		14	P1IEN6	0	H0	R/W		✓	✓	✓
		13	P1IEN5	0	H0	R/W		✓	✓	✓
		12	P1IEN4	0	H0	R/W		✓	✓	✓
		11	P1IEN3	0	H0	R/W		✓	✓	✓
		10	P1IEN2	0	H0	R/W		✓	✓	✓
		9	P1IEN1	0	H0	R/W		–	–	✓
		8	P1IEN0	0	H0	R/W		–	–	✓
		7	P1OEN7	0	H0	R/W	–	✓	✓	✓
		6	P1OEN6	0	H0	R/W		✓	✓	✓
		5	P1OEN5	0	H0	R/W		✓	✓	✓
		4	P1OEN4	0	H0	R/W		✓	✓	✓
		3	P1OEN3	0	H0	R/W		✓	✓	✓
		2	P1OEN2	0	H0	R/W		✓	✓	✓
		1	P1OEN1	0	H0	R/W		–	–	✓
		0	P1OEN0	0	H0	R/W		–	–	✓
0x0020 0214	P1RCTL (P1 Port Pull-up/down Control Register)	15	P1PDPU7	0	H0	R/W	–	✓	✓	✓
		14	P1PDPU6	0	H0	R/W		✓	✓	✓
		13	P1PDPU5	0	H0	R/W		✓	✓	✓
		12	P1PDPU4	0	H0	R/W		✓	✓	✓
		11	P1PDPU3	0	H0	R/W		✓	✓	✓
		10	P1PDPU2	0	H0	R/W		✓	✓	✓
		9	P1PDPU1	0	H0	R/W		–	–	✓
		8	P1PDPU0	0	H0	R/W		–	–	✓
		7	P1REN7	0	H0	R/W	–	✓	✓	✓
		6	P1REN6	0	H0	R/W		✓	✓	✓
		5	P1REN5	0	H0	R/W		✓	✓	✓
		4	P1REN4	0	H0	R/W		✓	✓	✓
		3	P1REN3	0	H0	R/W		✓	✓	✓
		2	P1REN2	0	H0	R/W		✓	✓	✓
		1	P1REN1	0	H0	R/W		–	–	✓
		0	P1REN0	0	H0	R/W		–	–	✓
0x0020 0216	P1INTF (P1 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P1IF7	0	H0	R/W	Cleared by writing 1.	✓	✓	✓
		6	P1IF6	0	H0	R/W		✓	✓	✓
		5	P1IF5	0	H0	R/W		✓	✓	✓
		4	P1IF4	0	H0	R/W		✓	✓	✓
		3	P1IF3	0	H0	R/W		✓	✓	✓
		2	P1IF2	0	H0	R/W		✓	✓	✓
		1	P1IF1	0	H0	R/W		–	–	✓
		0	P1IF0	0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0218	P1INTCTL (P1 Port Interrupt Control Register)	15	P1EDGE7	0	H0	R/W	–	✓	✓	✓
		14	P1EDGE6	0	H0	R/W		✓	✓	✓
		13	P1EDGE5	0	H0	R/W		✓	✓	✓
		12	P1EDGE4	0	H0	R/W		✓	✓	✓
		11	P1EDGE3	0	H0	R/W		✓	✓	✓
		10	P1EDGE2	0	H0	R/W		✓	✓	✓
		9	P1EDGE1	0	H0	R/W		–	–	✓
		8	P1EDGE0	0	H0	R/W		–	–	✓
		7	P1IE7	0	H0	R/W	–	✓	✓	✓
		6	P1IE6	0	H0	R/W		✓	✓	✓
		5	P1IE5	0	H0	R/W		✓	✓	✓
		4	P1IE4	0	H0	R/W		✓	✓	✓
		3	P1IE3	0	H0	R/W		✓	✓	✓
		2	P1IE2	0	H0	R/W		✓	✓	✓
		1	P1IE1	0	H0	R/W		–	–	✓
		0	P1IE0	0	H0	R/W		–	–	✓
0x0020 021a	P1CHATEN (P1 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P1CHATEN7	0	H0	R/W	–	✓	✓	✓
		6	P1CHATEN6	0	H0	R/W		✓	✓	✓
		5	P1CHATEN5	0	H0	R/W		✓	✓	✓
		4	P1CHATEN4	0	H0	R/W		✓	✓	✓
		3	P1CHATEN3	0	H0	R/W		✓	✓	✓
		2	P1CHATEN2	0	H0	R/W		✓	✓	✓
		1	P1CHATEN1	0	H0	R/W		–	–	✓
		0	P1CHATEN0	0	H0	R/W		–	–	✓
0x0020 021c	P1MODSEL (P1 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P1SEL7	0	H0	R/W	–	✓	✓	✓
		6	P1SEL6	0	H0	R/W		✓	✓	✓
		5	P1SEL5	0	H0	R/W		✓	✓	✓
		4	P1SEL4	0	H0	R/W		✓	✓	✓
		3	P1SEL3	0	H0	R/W		✓	✓	✓
		2	P1SEL2	0	H0	R/W		✓	✓	✓
		1	P1SEL1	0	H0	R/W		–	–	✓
		0	P1SEL0	0	H0	R/W		–	–	✓
0x0020 021e	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		–	–	✓
0x0020 0220	P2DAT (P2 Port Data Register)	15	P2OUT7	0	H0	R/W	–	–	–	✓
		14	P2OUT6	0	H0	R/W		–	–	✓
		13	P2OUT5	0	H0	R/W		–	–	✓
		12	P2OUT4	0	H0	R/W		–	–	✓
		11	P2OUT3	0	H0	R/W		–	✓	✓
		10	P2OUT2	0	H0	R/W		–	✓	✓
		9	P2OUT1	0	H0	R/W		–	✓	✓
		8	P2OUT0	0	H0	R/W		✓	✓	✓
		7	P2IN7	0	H0	R	–	–	–	✓
		6	P2IN6	0	H0	R		–	–	✓
		5	P2IN5	0	H0	R		–	–	✓
		4	P2IN4	0	H0	R		–	–	✓
		3	P2IN3	0	H0	R		–	✓	✓
		2	P2IN2	0	H0	R		–	✓	✓
		1	P2IN1	0	H0	R		–	✓	✓
		0	P2IN0	0	H0	R		✓	✓	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0222	P2IOEN (P2 Port Enable Register)	15	P2IEN7	0	H0	R/W	–	–	–	✓
		14	P2IEN6	0	H0	R/W		–	–	✓
		13	P2IEN5	0	H0	R/W		–	–	✓
		12	P2IEN4	0	H0	R/W		–	–	✓
		11	P2IEN3	0	H0	R/W		–	✓	✓
		10	P2IEN2	0	H0	R/W		–	✓	✓
		9	P2IEN1	0	H0	R/W		–	✓	✓
		8	P2IEN0	0	H0	R/W		✓	✓	✓
		7	P2OEN7	0	H0	R/W	–	–	–	✓
		6	P2OEN6	0	H0	R/W		–	–	✓
		5	P2OEN5	0	H0	R/W		–	–	✓
		4	P2OEN4	0	H0	R/W		–	–	✓
		3	P2OEN3	0	H0	R/W		–	✓	✓
		2	P2OEN2	0	H0	R/W		–	✓	✓
		1	P2OEN1	0	H0	R/W		–	✓	✓
		0	P2OEN0	0	H0	R/W		✓	✓	✓
0x0020 0224	P2RCTL (P2 Port Pull-up/down Control Register)	15	P2PDPU7	0	H0	R/W	–	–	–	✓
		14	P2PDPU6	0	H0	R/W		–	–	✓
		13	P2PDPU5	0	H0	R/W		–	–	✓
		12	P2PDPU4	0	H0	R/W		–	–	✓
		11	P2PDPU3	0	H0	R/W		–	✓	✓
		10	P2PDPU2	0	H0	R/W		–	✓	✓
		9	P2PDPU1	0	H0	R/W		–	✓	✓
		8	P2PDPU0	0	H0	R/W		✓	✓	✓
		7	P2REN7	0	H0	R/W	–	–	–	✓
		6	P2REN6	0	H0	R/W		–	–	✓
		5	P2REN5	0	H0	R/W		–	–	✓
		4	P2REN4	0	H0	R/W		–	–	✓
		3	P2REN3	0	H0	R/W		–	✓	✓
		2	P2REN2	0	H0	R/W		–	✓	✓
		1	P2REN1	0	H0	R/W		–	✓	✓
		0	P2REN0	0	H0	R/W		✓	✓	✓
0x0020 0226	P2INTF (P2 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P2IF7	0	H0	R/W	Cleared by writing 1.	–	–	✓
		6	P2IF6	0	H0	R/W		–	–	✓
		5	P2IF5	0	H0	R/W		–	–	✓
		4	P2IF4	0	H0	R/W		–	–	✓
		3	P2IF3	0	H0	R/W		–	✓	✓
		2	P2IF2	0	H0	R/W		–	✓	✓
		1	P2IF1	0	H0	R/W		–	✓	✓
		0	P2IF0	0	H0	R/W		✓	✓	✓
0x0020 0228	P2INTCTL (P2 Port Interrupt Control Register)	15	P2EDGE7	0	H0	R/W	–	–	–	✓
		14	P2EDGE6	0	H0	R/W		–	–	✓
		13	P2EDGE5	0	H0	R/W		–	–	✓
		12	P2EDGE4	0	H0	R/W		–	–	✓
		11	P2EDGE3	0	H0	R/W		–	✓	✓
		10	P2EDGE2	0	H0	R/W		–	✓	✓
		9	P2EDGE1	0	H0	R/W		–	✓	✓
		8	P2EDGE0	0	H0	R/W		✓	✓	✓
		7	P2IE7	0	H0	R/W	–	–	–	✓
		6	P2IE6	0	H0	R/W		–	–	✓
		5	P2IE5	0	H0	R/W		–	–	✓
		4	P2IE4	0	H0	R/W		–	–	✓
		3	P2IE3	0	H0	R/W		–	✓	✓
		2	P2IE2	0	H0	R/W		–	✓	✓
		1	P2IE1	0	H0	R/W		–	✓	✓
		0	P2IE0	0	H0	R/W		✓	✓	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 022a	P2CHATEN (P2 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P2CHATEN7	0	H0	R/W	–	–	–	✓
		6	P2CHATEN6	0	H0	R/W		–	–	✓
		5	P2CHATEN5	0	H0	R/W		–	–	✓
		4	P2CHATEN4	0	H0	R/W		–	–	✓
		3	P2CHATEN3	0	H0	R/W		–	✓	✓
		2	P2CHATEN2	0	H0	R/W		–	✓	✓
		1	P2CHATEN1	0	H0	R/W		–	✓	✓
		0	P2CHATEN0	0	H0	R/W		✓	✓	✓
0x0020 022c	P2MODSEL (P2 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P2SEL7	0	H0	R/W	–	–	–	✓
		6	P2SEL6	0	H0	R/W		–	–	✓
		5	P2SEL5	0	H0	R/W		–	–	✓
		4	P2SEL4	0	H0	R/W		–	–	✓
		3	P2SEL3	0	H0	R/W		–	✓	✓
		2	P2SEL2	0	H0	R/W		–	✓	✓
		1	P2SEL1	0	H0	R/W		–	✓	✓
		0	P2SEL0	0	H0	R/W		✓	✓	✓
0x0020 022e	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		✓	✓	✓
0x0020 0230	P3DAT (P3 Port Data Register)	15	P3OUT7	0	H0	R/W	–	–	–	✓
		14	P3OUT6	0	H0	R/W		–	–	✓
		13	P3OUT5	0	H0	R/W		–	✓	✓
		12	P3OUT4	0	H0	R/W		–	✓	✓
		11	P3OUT3	0	H0	R/W		–	✓	✓
		10	P3OUT2	0	H0	R/W		✓	✓	✓
		9	P3OUT1	0	H0	R/W		✓	✓	✓
		8	P3OUT0	0	H0	R/W		–	–	✓
		7	P3IN7	0	H0	R	–	–	–	✓
		6	P3IN6	0	H0	R		–	–	✓
		5	P3IN5	0	H0	R		–	✓	✓
		4	P3IN4	0	H0	R		–	✓	✓
		3	P3IN3	0	H0	R		–	✓	✓
		2	P3IN2	0	H0	R		✓	✓	✓
		1	P3IN1	0	H0	R		✓	✓	✓
		0	P3IN0	0	H0	R		–	–	✓
0x0020 0232	P3IOEN (P3 Port Enable Register)	15	P3IEN7	0	H0	R/W	–	–	–	✓
		14	P3IEN6	0	H0	R/W		–	–	✓
		13	P3IEN5	0	H0	R/W		–	✓	✓
		12	P3IEN4	0	H0	R/W		–	✓	✓
		11	P3IEN3	0	H0	R/W		–	✓	✓
		10	P3IEN2	0	H0	R/W		✓	✓	✓
		9	P3IEN1	0	H0	R/W		✓	✓	✓
		8	P3IEN0	0	H0	R/W		–	–	✓
		7	P3OEN7	0	H0	R/W	–	–	–	✓
		6	P3OEN6	0	H0	R/W		–	–	✓
		5	P3OEN5	0	H0	R/W		–	✓	✓
		4	P3OEN4	0	H0	R/W		–	✓	✓
		3	P3OEN3	0	H0	R/W		–	✓	✓
		2	P3OEN2	0	H0	R/W		✓	✓	✓
		1	P3OEN1	0	H0	R/W		✓	✓	✓
		0	P3OEN0	0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0234	P3RCTL (P3 Port Pull-up/down Control Register)	15	P3PDPU7	0	H0	R/W	–	–	–	✓
		14	P3PDPU6	0	H0	R/W		–	–	✓
		13	P3PDPU5	0	H0	R/W		–	✓	✓
		12	P3PDPU4	0	H0	R/W		–	✓	✓
		11	P3PDPU3	0	H0	R/W		–	✓	✓
		10	P3PDPU2	0	H0	R/W		✓	✓	✓
		9	P3PDPU1	0	H0	R/W		✓	✓	✓
		8	P3PDPU0	0	H0	R/W		–	–	✓
		7	P3REN7	0	H0	R/W	–	–	–	✓
		6	P3REN6	0	H0	R/W		–	–	✓
		5	P3REN5	0	H0	R/W		–	✓	✓
		4	P3REN4	0	H0	R/W		–	✓	✓
		3	P3REN3	0	H0	R/W		–	✓	✓
		2	P3REN2	0	H0	R/W		✓	✓	✓
		1	P3REN1	0	H0	R/W		✓	✓	✓
		0	P3REN0	0	H0	R/W		–	–	✓
0x0020 0236	P3INTF (P3 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P3IF7	0	H0	R/W	Cleared by writing 1.	–	–	✓
		6	P3IF6	0	H0	R/W		–	–	✓
		5	P3IF5	0	H0	R/W		–	✓	✓
		4	P3IF4	0	H0	R/W		–	✓	✓
		3	P3IF3	0	H0	R/W		–	✓	✓
		2	P3IF2	0	H0	R/W		✓	✓	✓
		1	P3IF1	0	H0	R/W		✓	✓	✓
		0	P3IF0	0	H0	R/W		–	–	✓
0x0020 0238	P3INTCTL (P3 Port Interrupt Control Register)	15	P3EDGE7	0	H0	R/W	–	–	–	✓
		14	P3EDGE6	0	H0	R/W		–	–	✓
		13	P3EDGE5	0	H0	R/W		–	✓	✓
		12	P3EDGE4	0	H0	R/W		–	✓	✓
		11	P3EDGE3	0	H0	R/W		–	✓	✓
		10	P3EDGE2	0	H0	R/W		✓	✓	✓
		9	P3EDGE1	0	H0	R/W		✓	✓	✓
		8	P3EDGE0	0	H0	R/W		–	–	✓
		7	P3IE7	0	H0	R/W	–	–	–	✓
		6	P3IE6	0	H0	R/W		–	–	✓
		5	P3IE5	0	H0	R/W		–	✓	✓
		4	P3IE4	0	H0	R/W		–	✓	✓
		3	P3IE3	0	H0	R/W		–	✓	✓
		2	P3IE2	0	H0	R/W		✓	✓	✓
		1	P3IE1	0	H0	R/W		✓	✓	✓
		0	P3IE0	0	H0	R/W		–	–	✓
0x0020 023a	P3CHATEN (P3 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P3CHATEN7	0	H0	R/W	–	–	–	✓
		6	P3CHATEN6	0	H0	R/W		–	–	✓
		5	P3CHATEN5	0	H0	R/W		–	✓	✓
		4	P3CHATEN4	0	H0	R/W		–	✓	✓
		3	P3CHATEN3	0	H0	R/W		–	✓	✓
		2	P3CHATEN2	0	H0	R/W		✓	✓	✓
		1	P3CHATEN1	0	H0	R/W		✓	✓	✓
		0	P3CHATEN0	0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 023c	P3MODSEL (P3 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7	P3SEL7	0	H0	R/W	–	–	–	✓
		6	P3SEL6	0	H0	R/W		–	–	✓
		5	P3SEL5	0	H0	R/W		–	✓	✓
		4	P3SEL4	0	H0	R/W		–	✓	✓
		3	P3SEL3	0	H0	R/W		–	✓	✓
		2	P3SEL2	0	H0	R/W		✓	✓	✓
		1	P3SEL1	0	H0	R/W		✓	✓	✓
		0	P3SEL0	0	H0	R/W		–	–	✓
0x0020 023e	P3FNCSEL (P3 Port Function Select Register)	15–14	P37MUX[1:0]	0x0	H0	R/W	–	–	–	✓
		13–12	P36MUX[1:0]	0x0	H0	R/W		–	–	✓
		11–10	P35MUX[1:0]	0x0	H0	R/W		–	✓	✓
		9–8	P34MUX[1:0]	0x0	H0	R/W		–	✓	✓
		7–6	P33MUX[1:0]	0x0	H0	R/W		–	✓	✓
		5–4	P32MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		3–2	P31MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		1–0	P30MUX[1:0]	0x0	H0	R/W		–	–	✓
0x0020 0240	P4DAT (P4 Port Data Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P4OUT5	0	H0	R/W	–	–	✓	✓
		12	P4OUT4	0	H0	R/W		✓	✓	✓
		11	P4OUT3	0	H0	R/W		✓	✓	✓
		10	P4OUT2	0	H0	R/W		✓	✓	✓
		9	P4OUT1	0	H0	R/W		–	–	✓
		8	P4OUT0	0	H0	R/W		–	–	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P4IN5	0	H0	R	–	–	✓	✓
		4	P4IN4	0	H0	R		✓	✓	✓
		3	P4IN3	0	H0	R		✓	✓	✓
		2	P4IN2	0	H0	R		✓	✓	✓
		1	P4IN1	0	H0	R		–	–	✓
		0	P4IN0	0	H0	R		–	–	✓
0x0020 0242	P4IOEN (P4 Port Enable Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P4IEN5	0	H0	R/W	–	–	✓	✓
		12	P4IEN4	0	H0	R/W		✓	✓	✓
		11	P4IEN3	0	H0	R/W		✓	✓	✓
		10	P4IEN2	0	H0	R/W		✓	✓	✓
		9	P4IEN1	0	H0	R/W		–	–	✓
		8	P4IEN0	0	H0	R/W		–	–	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P4OEN5	0	H0	R/W	–	–	✓	✓
		4	P4OEN4	0	H0	R/W		✓	✓	✓
		3	P4OEN3	0	H0	R/W		✓	✓	✓
		2	P4OEN2	0	H0	R/W		✓	✓	✓
		1	P4OEN1	0	H0	R/W		–	–	✓
		0	P4OEN0	0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0244	P4RCTL (P4 Port Pull-up/down Control Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P4PDPU5	0	H0	R/W	–	–	✓	✓
		12	P4PDPU4	0	H0	R/W		✓	✓	✓
		11	P4PDPU3	0	H0	R/W		✓	✓	✓
		10	P4PDPU2	0	H0	R/W		✓	✓	✓
		9	P4PDPU1	0	H0	R/W		–	–	✓
		8	P4PDPU0	0	H0	R/W		–	–	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P4REN5	0	H0	R/W	–	–	✓	✓
		4	P4REN4	0	H0	R/W		✓	✓	✓
		3	P4REN3	0	H0	R/W		✓	✓	✓
		2	P4REN2	0	H0	R/W		✓	✓	✓
		1	P4REN1	0	H0	R/W		–	–	✓
		0	P4REN0	0	H0	R/W		–	–	✓
0x0020 0246	P4INTF (P4 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7–6	–	0x0	–	R		–	–	–
		5	P4IF5	0	H0	R/W	Cleared by writing 1.	–	✓	✓
		4	P4IF4	0	H0	R/W		✓	✓	✓
		3	P4IF3	0	H0	R/W		✓	✓	✓
		2	P4IF2	0	H0	R/W		✓	✓	✓
		1	P4IF1	0	H0	R/W		–	–	✓
		0	P4IF0	0	H0	R/W		–	–	✓
0x0020 0248	P4INTCTL (P4 Port Interrupt Control Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P4EDGE5	0	H0	R/W	–	–	✓	✓
		12	P4EDGE4	0	H0	R/W		✓	✓	✓
		11	P4EDGE3	0	H0	R/W		✓	✓	✓
		10	P4EDGE2	0	H0	R/W		✓	✓	✓
		9	P4EDGE1	0	H0	R/W		–	–	✓
		8	P4EDGE0	0	H0	R/W		–	–	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P4IE5	0	H0	R/W	–	–	✓	✓
		4	P4IE4	0	H0	R/W		✓	✓	✓
		3	P4IE3	0	H0	R/W		✓	✓	✓
		2	P4IE2	0	H0	R/W		✓	✓	✓
		1	P4IE1	0	H0	R/W		–	–	✓
		0	P4IE0	0	H0	R/W		–	–	✓
0x0020 024a	P4CHATEN (P4 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7–6	–	0x0	–	R		–	–	–
		5	P4CHATEN5	0	H0	R/W	–	–	✓	✓
		4	P4CHATEN4	0	H0	R/W		✓	✓	✓
		3	P4CHATEN3	0	H0	R/W		✓	✓	✓
		2	P4CHATEN2	0	H0	R/W		✓	✓	✓
		1	P4CHATEN1	0	H0	R/W		–	–	✓
		0	P4CHATEN0	0	H0	R/W		–	–	✓
0x0020 024c	P4MODSEL (P4 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7–6	–	0x0	–	R		–	–	–
		5	P4SEL5	0	H0	R/W	–	–	✓	✓
		4	P4SEL4	0	H0	R/W		✓	✓	✓
		3	P4SEL3	0	H0	R/W		✓	✓	✓
		2	P4SEL2	0	H0	R/W		✓	✓	✓
		1	P4SEL1	0	H0	R/W		–	–	✓
		0	P4SEL0	0	H0	R/W		–	–	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 024e	P4FNCSEL (P4 Port Function Select Register)	15–12	–	0x0	H0	R/W	–	–	–	–
		11–10	P45MUX[1:0]	0x0	H0	R/W	–	–	✓	✓
		9–8	P44MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		7–6	P43MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		5–4	P42MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		3–2	P41MUX[1:0]	0x0	H0	R/W		–	–	✓
		1–0	P40MUX[1:0]	0x0	H0	R/W		–	–	✓
0x0020 0250	P5DAT (P5 Port Data Register)	15	–	0	–	R	–	–	–	–
		14	P5OUT6	0	H0	R/W	–	–	✓	✓
		13	P5OUT5	0	H0	R/W		–	✓	✓
		12	P5OUT4	0	H0	R/W		✓	✓	✓
		11	P5OUT3	0	H0	R/W		–	–	✓
		10	P5OUT2	0	H0	R/W		–	–	✓
		9	P5OUT1	0	H0	R/W		✓	✓	✓
		8	P5OUT0	0	H0	R/W		✓	✓	✓
		7	–	0	–	R	–	–	–	–
		6	P5IN6	0	H0	R	–	–	✓	✓
		5	P5IN5	0	H0	R		–	✓	✓
		4	P5IN4	0	H0	R		✓	✓	✓
		3	P5IN3	0	H0	R		–	–	✓
		2	P5IN2	0	H0	R		–	–	✓
		1	P5IN1	0	H0	R		✓	✓	✓
		0	P5IN0	0	H0	R		✓	✓	✓
0x0020 0252	P5IOEN (P5 Port Enable Register)	15	–	0	–	R	–	–	–	–
		14	P5IEN6	0	H0	R/W	–	–	✓	✓
		13	P5IEN5	0	H0	R/W		–	✓	✓
		12	P5IEN4	0	H0	R/W		✓	✓	✓
		11	P5IEN3	0	H0	R/W		–	–	✓
		10	P5IEN2	0	H0	R/W		–	–	✓
		9	P5IEN1	0	H0	R/W		✓	✓	✓
		8	P5IEN0	0	H0	R/W		✓	✓	✓
		7	–	0	–	R	–	–	–	–
		6	P5OEN6	0	H0	R/W	–	–	✓	✓
		5	P5OEN5	0	H0	R/W		–	✓	✓
		4	P5OEN4	0	H0	R/W		✓	✓	✓
		3	P5OEN3	0	H0	R/W		–	–	✓
		2	P5OEN2	0	H0	R/W		–	–	✓
		1	P5OEN1	0	H0	R/W		✓	✓	✓
		0	P5OEN0	0	H0	R/W		✓	✓	✓
0x0020 0254	P5RCTL (P5 Port Pull-up/down Control Register)	15	–	0	–	R	–	–	–	–
		14	P5PDPU6	0	H0	R/W	–	–	✓	✓
		13	P5PDPU5	0	H0	R/W		–	✓	✓
		12	P5PDPU4	0	H0	R/W		✓	✓	✓
		11	P5PDPU3	0	H0	R/W		–	–	✓
		10	P5PDPU2	0	H0	R/W		–	–	✓
		9	P5PDPU1	0	H0	R/W		✓	✓	✓
		8	P5PDPU0	0	H0	R/W		✓	✓	✓
		7	–	0	–	R	–	–	–	–
		6	P5REN6	0	H0	R/W	–	–	✓	✓
		5	P5REN5	0	H0	R/W		–	✓	✓
		4	P5REN4	0	H0	R/W		✓	✓	✓
		3	P5REN3	0	H0	R/W		–	–	✓
		2	P5REN2	0	H0	R/W		–	–	✓
		1	P5REN1	0	H0	R/W		✓	✓	✓
		0	P5REN0	0	H0	R/W		✓	✓	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0256	P5INTF (P5 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7	–	0	–	R	–	–	–	–
		6	P5IF6	0	H0	R/W	Cleared by writing 1.	–	✓	✓
		5	P5IF5	0	H0	R/W		–	✓	✓
		4	P5IF4	0	H0	R/W		✓	✓	✓
		3	P5IF3	0	H0	R/W		–	–	✓
		2	P5IF2	0	H0	R/W		–	–	✓
		1	P5IF1	0	H0	R/W		✓	✓	✓
		0	P5IF0	0	H0	R/W		✓	✓	✓
0x0020 0258	P5INTCTL (P5 Port Interrupt Control Register)	15	–	0	–	R	–	–	–	–
		14	P5EDGE6	0	H0	R/W	–	–	✓	✓
		13	P5EDGE5	0	H0	R/W		–	✓	✓
		12	P5EDGE4	0	H0	R/W		✓	✓	✓
		11	P5EDGE3	0	H0	R/W		–	–	✓
		10	P5EDGE2	0	H0	R/W		–	–	✓
		9	P5EDGE1	0	H0	R/W		✓	✓	✓
		8	P5EDGE0	0	H0	R/W		✓	✓	✓
		7	–	0	–	R	–	–	–	–
		6	P5IE6	0	H0	R/W	–	–	✓	✓
		5	P5IE5	0	H0	R/W		–	✓	✓
		4	P5IE4	0	H0	R/W		✓	✓	✓
		3	P5IE3	0	H0	R/W		–	–	✓
		2	P5IE2	0	H0	R/W		–	–	✓
		1	P5IE1	0	H0	R/W		✓	✓	✓
		0	P5IE0	0	H0	R/W		✓	✓	✓
0x0020 025a	P5CHATEN (P5 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7	–	0	–	R	–	–	–	–
		6	P5CHATEN6	0	H0	R/W	–	–	✓	✓
		5	P5CHATEN5	0	H0	R/W		–	✓	✓
		4	P5CHATEN4	0	H0	R/W		✓	✓	✓
		3	P5CHATEN3	0	H0	R/W		–	–	✓
		2	P5CHATEN2	0	H0	R/W		–	–	✓
		1	P5CHATEN1	0	H0	R/W		✓	✓	✓
		0	P5CHATEN0	0	H0	R/W		✓	✓	✓
0x0020 025c	P5MODSEL (P5 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7	–	0	–	R	–	–	–	–
		6	P5SEL6	0	H0	R/W	–	–	✓	✓
		5	P5SEL5	0	H0	R/W		–	✓	✓
		4	P5SEL4	0	H0	R/W		✓	✓	✓
		3	P5SEL3	0	H0	R/W		–	–	✓
		2	P5SEL2	0	H0	R/W		–	–	✓
		1	P5SEL1	1	H0	R/W		✓	✓	✓
		0	P5SEL0	1	H0	R/W		✓	✓	✓
0x0020 025e	P5FNCSEL (P5 Port Function Select Register)	15–14	–	0x0	–	R	–	–	–	–
		13–12	P56MUX[1:0]	0x0	H0	R/W	–	–	✓	✓
		11–10	P55MUX[1:0]	0x0	H0	R/W		–	✓	✓
		9–8	P54MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		7–6	P53MUX[1:0]	0x0	H0	R/W		–	–	✓
		5–4	P52MUX[1:0]	0x0	H0	R/W		–	–	✓
		3–2	P51MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		1–0	P50MUX[1:0]	0x0	H0	R/W		✓	✓	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0260	P6DAT (P6 Port Data Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P6OUT5	0	H0	R/W	–	✓	✓	✓
		12	P6OUT4	0	H0	R/W		✓	✓	✓
		11	P6OUT3	0	H0	R/W		✓	✓	✓
		10	P6OUT2	0	H0	R/W		✓	✓	✓
		9	P6OUT1	0	H0	R/W		✓	✓	✓
		8	P6OUT0	0	H0	R/W		✓	✓	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P6IN5	0	H0	R	–	✓	✓	✓
		4	P6IN4	0	H0	R		✓	✓	✓
		3	P6IN3	0	H0	R		✓	✓	✓
		2	P6IN2	0	H0	R		✓	✓	✓
		1	P6IN1	0	H0	R		✓	✓	✓
		0	P6IN0	0	H0	R		✓	✓	✓
0x0020 0262	P6IOEN (P6 Port Enable Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P6IEN5	0	H0	R/W	–	✓	✓	✓
		12	P6IEN4	0	H0	R/W		✓	✓	✓
		11	P6IEN3	0	H0	R/W		✓	✓	✓
		10	P6IEN2	0	H0	R/W		✓	✓	✓
		9	P6IEN1	0	H0	R/W		✓	✓	✓
		8	P6IEN0	0	H0	R/W		✓	✓	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P6OEN5	0	H0	R/W	–	✓	✓	✓
		4	P6OEN4	0	H0	R/W		✓	✓	✓
		3	P6OEN3	0	H0	R/W		✓	✓	✓
		2	P6OEN2	0	H0	R/W		✓	✓	✓
		1	P6OEN1	0	H0	R/W		✓	✓	✓
		0	P6OEN0	0	H0	R/W		✓	✓	✓
0x0020 0264	P6RCTL (P6 Port Pull-up/down Control Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P6PDPU5	0	H0	R/W	–	✓	✓	✓
		12	P6PDPU4	0	H0	R/W		✓	✓	✓
		11	P6PDPU3	0	H0	R/W		✓	✓	✓
		10	P6PDPU2	0	H0	R/W		✓	✓	✓
		9	P6PDPU1	0	H0	R/W		✓	✓	✓
		8	P6PDPU0	0	H0	R/W		✓	✓	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P6REN5	0	H0	R/W	–	✓	✓	✓
		4	P6REN4	0	H0	R/W		✓	✓	✓
		3	P6REN3	0	H0	R/W		✓	✓	✓
		2	P6REN2	0	H0	R/W		✓	✓	✓
		1	P6REN1	0	H0	R/W		✓	✓	✓
		0	P6REN0	0	H0	R/W		✓	✓	✓
0x0020 0266	P6INTF (P6 Port Interrupt Flag Register)	15–8	–	0x00	–	R	–	–	–	–
		7–6	–	0x0	–	R		–	–	–
		5	P6IF5	0	H0	R/W	Cleared by writing 1.	✓	✓	✓
		4	P6IF4	0	H0	R/W		✓	✓	✓
		3	P6IF3	0	H0	R/W		✓	✓	✓
		2	P6IF2	0	H0	R/W		✓	✓	✓
		1	P6IF1	0	H0	R/W		✓	✓	✓
		0	P6IF0	0	H0	R/W		✓	✓	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0268	P6INTCTL (P6 Port Interrupt Control Register)	15–14	–	0x0	–	R	–	–	–	–
		13	P6EDGE5	0	H0	R/W	–	✓	✓	✓
		12	P6EDGE4	0	H0	R/W		✓	✓	✓
		11	P6EDGE3	0	H0	R/W		✓	✓	✓
		10	P6EDGE2	0	H0	R/W		✓	✓	✓
		9	P6EDGE1	0	H0	R/W		✓	✓	✓
		8	P6EDGE0	0	H0	R/W		✓	✓	✓
		7–6	–	0x0	–	R	–	–	–	–
		5	P6IE5	0	H0	R/W	–	✓	✓	✓
		4	P6IE4	0	H0	R/W		✓	✓	✓
		3	P6IE3	0	H0	R/W		✓	✓	✓
		2	P6IE2	0	H0	R/W		✓	✓	✓
		1	P6IE1	0	H0	R/W		✓	✓	✓
		0	P6IE0	0	H0	R/W		✓	✓	✓
0x0020 026a	P6CHATEN (P6 Port Chattering Filter Enable Register)	15–8	–	0x00	–	R	–	–	–	–
		7–6	–	0x0	–	R		–	–	–
		5	P6CHATEN5	0	H0	R/W	–	✓	✓	✓
		4	P6CHATEN4	0	H0	R/W		✓	✓	✓
		3	P6CHATEN3	0	H0	R/W		✓	✓	✓
		2	P6CHATEN2	0	H0	R/W		✓	✓	✓
		1	P6CHATEN1	0	H0	R/W		✓	✓	✓
		0	P6CHATEN0	0	H0	R/W		✓	✓	✓
0x0020 026c	P6MODSEL (P6 Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7–6	–	0x0	–	R		–	–	–
		5	P6SEL5	0	H0	R/W	–	✓	✓	✓
		4	P6SEL4	0	H0	R/W		✓	✓	✓
		3	P6SEL3	0	H0	R/W		✓	✓	✓
		2	P6SEL2	0	H0	R/W		✓	✓	✓
		1	P6SEL1	0	H0	R/W		✓	✓	✓
		0	P6SEL0	0	H0	R/W		✓	✓	✓
0x0020 026e	P6FNCSEL (P6 Port Function Select Register)	15–12	–	0x0	H0	R/W	–	–	–	–
		11–10	P65MUX[1:0]	0x0	H0	R/W	–	✓	✓	✓
		9–8	P64MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		7–6	P63MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		5–4	P62MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		3–2	P61MUX[1:0]	0x0	H0	R/W		✓	✓	✓
		1–0	P60MUX[1:0]	0x0	H0	R/W		✓	✓	✓
0x0020 02d0	PDDAT (Pd Port Data Register)	15–12	–	0x0	–	R	–	–	–	–
		11	PDOUT3	0	H0	R/W	–	✓	✓	✓
		10	PDOUT2	0	H0	R/W		✓	✓	✓
		9	PDOUT1	0	H0	R/W		✓	✓	✓
		8	PDOUT0	0	H0	R/W		✓	✓	✓
		7–4	–	0x0	–	R	–	–	–	–
		3	PDIN3	0	H0	R	–	✓	✓	✓
		2	PDIN2	0	H0	R		✓	✓	✓
		1	PDIN1	0	H0	R		✓	✓	✓
		0	PDIN0	0	H0	R		✓	✓	✓
0x0020 02d2	PDIOEN (Pd Port Enable Register)	15–12	–	0x0	–	R	–	–	–	–
		11	PDIEN3	0	H0	R/W	–	✓	✓	✓
		10	PDIEN2	0	H0	R/W		✓	✓	✓
		9	PDIEN1	0	H0	R/W		✓	✓	✓
		8	PDIEN0	0	H0	R/W		✓	✓	✓
		7–4	–	0x0	–	R	–	–	–	–
		3	PDOEN3	0	H0	R/W	–	✓	✓	✓
		2	PDOEN2	0	H0	R/W		✓	✓	✓
		1	PDOEN1	0	H0	R/W		✓	✓	✓
		0	PDOEN0	0	H0	R/W		✓	✓	✓

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 02d4	PDRCTL (Pd Port Pull-up/down Control Register)	15–12	–	0x0	–	R	–	–	–	–
		11	PDPDPU3	0	H0	R/W	–	✓	✓	✓
		10	PDPDPU2	0	H0	R/W		✓	✓	✓
		9	PDPDPU1	0	H0	R/W		✓	✓	✓
		8	PDPDPU0	0	H0	R/W		✓	✓	✓
		7–4	–	0x0	–	R	–	–	–	–
		3	PDREN3	0	H0	R/W	–	✓	✓	✓
		2	PDREN2	0	H0	R/W		✓	✓	✓
		1	PDREN1	0	H0	R/W		✓	✓	✓
		0	PDREN0	0	H0	R/W		✓	✓	✓
0x0020 02dc	PDMODSEL (Pd Port Mode Select Register)	15–8	–	0x00	–	R	–	–	–	–
		7–4	–	0x0	–	R		–	–	–
		3	PDSEL3	0	H0	R/W	–	✓	✓	✓
		2	PDSEL2	0	H0	R/W		✓	✓	✓
		1	PDSEL1	1	H0	R/W		✓	✓	✓
		0	PDSEL0	1	H0	R/W		✓	✓	✓
0x0020 02de	PDFNCSEL (Pd Port Function Select Register)	15–8	–	0x00	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		✓	✓	✓
0x0020 02e0	PPORTCLK (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	–	0x0	–	R	–	–	–	–
		1–0	CLKSRC[1:0]	0x0	H0	R/WP	–	✓	✓	✓
0x0020 02e2	PPORTINTFGRP (P Port Interrupt Flag Group Register)	15–8	–	0x00	–	R	–	–	–	–
		7	–	0	–	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		–	✓	✓

0x0020 0300–0x0020 031e Universal Port Multiplexer (UPMUX)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0300	UPMUXP0MUX0 (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				
0x0020 0302	UPMUXP0MUX1 (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				
0x0020 0304	UPMUXP0MUX2 (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				

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0306	UPMUXP0MUX3 (P06–07 Universal Port Multiplexer Setting Register)	15–13	P07PPFNC[2:0]	0x0	H0	R/W	–	–	–	✓
		12–11	P07PERICH[1:0]	0x0	H0	R/W				
		10–8	P07PERISEL[2:0]	0x0	H0	R/W				
		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				
0x0020 0308	UPMUXP1MUX0 (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				
0x0020 030a	UPMUXP1MUX1 (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				
0x0020 030c	UPMUXP1MUX2 (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				
0x0020 030e	UPMUXP1MUX3 (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				
0x0020 0310	UPMUXP2MUX0 (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				
0x0020 0312	UPMUXP2MUX1 (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				
0x0020 0314	UPMUXP2MUX2 (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				
0x0020 0316	UPMUXP2MUX3 (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				

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks	32 pin	48 pin	64 pin
0x0020 0318	UPMUXP3MUX0 (P30–31 Universal Port Multiplexer Setting Register)	15–13	P31PPFNC[2:0]	0x0	H0	R/W	–	✓	✓	✓
		12–11	P31PERICH[1:0]	0x0	H0	R/W				
		10–8	P31PERISEL[2:0]	0x0	H0	R/W				
		7–5	P30PPFNC[2:0]	0x0	H0	R/W		–	–	✓
		4–3	P30PERICH[1:0]	0x0	H0	R/W				
		2–0	P30PERISEL[2:0]	0x0	H0	R/W				
0x0020 031a	UPMUXP3MUX1 (P32–33 Universal Port Multiplexer Setting Register)	15–13	P33PPFNC[2:0]	0x0	H0	R/W	–	–	✓	✓
		12–11	P33PERICH[1:0]	0x0	H0	R/W				
		10–8	P33PERISEL[2:0]	0x0	H0	R/W				
		7–5	P32PPFNC[2:0]	0x0	H0	R/W		✓	✓	✓
		4–3	P32PERICH[1:0]	0x0	H0	R/W				
		2–0	P32PERISEL[2:0]	0x0	H0	R/W				
0x0020 031c	UPMUXP3MUX2 (P34–35 Universal Port Multiplexer Setting Register)	15–13	P35PPFNC[2:0]	0x0	H0	R/W	–	–	✓	✓
		12–11	P35PERICH[1:0]	0x0	H0	R/W				
		10–8	P35PERISEL[2:0]	0x0	H0	R/W				
		7–5	P34PPFNC[2:0]	0x0	H0	R/W		–	✓	✓
		4–3	P34PERICH[1:0]	0x0	H0	R/W				
		2–0	P34PERISEL[2:0]	0x0	H0	R/W				
0x0020 031e	UPMUXP3MUX3 (P36 Universal Port Multiplexer Setting Register)	15–13	P37PPFNC[2:0]	0x0	H0	R/W	–	–	✓	✓
		12–11	P37PERICH[1:0]	0x0	H0	R/W				
		10–8	P37PERISEL[2:0]	0x0	H0	R/W				
		7–5	P36PPFNC[2:0]	0x0	H0	R/W		–	✓	✓
		4–3	P36PERICH[1:0]	0x0	H0	R/W				
		2–0	P36PERISEL[2:0]	0x0	H0	R/W				

0x0020 0380–0x0020 0394 UART (UART3) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0380	UART3_0CLK (UART3 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	
0x0020 0382	UART3_0MOD (UART3 Ch.0 Mode Register)	15–13	–	0x00	–	R	–
		12	PECAR	0	H0	R/W	
		11	CAREN	0	H0	R/W	
		10	BRDIV	0	H0	R/W	
		9	INVRX	0	H0	R/W	
		8	INVTX	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	
0x0020 0384	UART3_0BR (UART3 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	
0x0020 0386	UART3_0CTL (UART3 Ch.0 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 0388	UART3_0TXD (UART3 Ch.0 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 038a	UART3_0RXD (UART3 Ch.0 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x0020 038c	UART3_0INTF (UART3 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 UART3_0RXD 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 UART3_0RXD register.
		1	RB1FIF	0	H0/S0	R	
		0	TBEIF	1	H0/S0	R	Cleared by writing to the UART3_0TXD register.
0x0020 038e	UART3_0INTE (UART3 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	
0x0020 0390	UART3_0 TBEDMAEN (UART3 Ch.0 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 0392	UART3_0 RB1FDMAEN (UART3 Ch.0 Receive Buffer One Byte Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RB1FDMAEN[3:0]	0x0	H0	R/W	
0x0020 0394	UART3_0CAWF (UART3 Ch.0 Carrier Waveform Register)	15–8	–	0x00	–	R	–
		7–0	CRPER[7:0]	0x00	H0	R/W	

0x0020 03a0–0x0020 03ac 16-bit Timer (T16) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 03a0	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	
0x0020 03a2	T16_1MOD (T16 Ch.1 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 03a4	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	
0x0020 03a6	T16_1TR (T16 Ch.1 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 03a8	T16_1TC (T16 Ch.1 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 03aa	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.
0x0020 03ac	T16_1INTE (T16 Ch.1 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 03b0–0x0020 03be Synchronous Serial Interface (SPIA) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 03b0	SPIA_0MOD (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	
0x0020 03b2	SPIA_0CTL (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	
0x0020 03b4	SPIA_0TXD (SPIA Ch.0 Transmit Data Register)	15–0	TXD[15:0]	0x0000	H0	R/W	–
0x0020 03b6	SPIA_0RXD (SPIA Ch.0 Receive Data Register)	15–0	RXD[15:0]	0x0000	H0	R	–
0x0020 03b8	SPIA_0INTF (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 SPIA_0RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the SPIA_0TXD register.
0x0020 03ba	SPIA_0INTE (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	
0x0020 03bc	SPIA_0TBEDMAEN (SPIA Ch.0 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 03be	SPIA_0RBFDMAEN (SPIA Ch.0 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	

0x0020 03c0–0x0020 03d6 I²C (I2C) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 03c0	I2C_0CLK (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	
0x0020 03c2	I2C_0MOD (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	
0x0020 03c4	I2C_0BR (I2C Ch.0 Baud-Rate Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	BRT[6:0]	0x7f	H0	R/W	
0x0020 03c8	I2C_0OADR (I2C Ch.0 Own Address Register)	15–10	–	0x00	–	R	–
		9–0	OADR[9:0]	0x000	H0	R/W	
0x0020 03ca	I2C_0CTL (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	
0x0020 03cc	I2C_0TXD (I2C Ch.0 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x0020 03ce	I2C_0RXD (I2C Ch.0 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x0020 03d0	I2C_0INTF (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 I2C_0RXD register.
		0	TBEIF	0	H0/S0	R	Cleared by writing to the I2C_0TXD register.
0x0020 03d2	I2C_0INTE (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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 03d4	I2C_0TBEDMAEN (I2C Ch.0 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 03d6	I2C_0RBFDMAEN (I2C Ch.0 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	

0x0020 0400–0x0020 042c 16-bit PWM Timer (T16B) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0400	T16B_0CLK (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	
0x0020 0402	T16B_0CTL (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	
0x0020 0404	T16B_0MC (T16B Ch.0 Max Counter Data Register)	15–0	MC[15:0]	0xffff	H0	R/W	–
0x0020 0406	T16B_0TC (T16B Ch.0 Timer Counter Data Register)	15–0	TC[15:0]	0x0000	H0	R	–
0x0020 0408	T16B_0CS (T16B Ch.0 Counter Status Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	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	
0x0020 040a	T16B_0INTF (T16B Ch.0 Interrupt Flag Register)	15–10	–	0x00	–	R	–
		9	CAPOW3IF	0	H0	R/W	Cleared by writing 1.
		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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 040c	T16B_0INTE (T16B Ch.0 Interrupt Enable Register)	15–10	–	0x00	–	R	–
		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	
0x0020 040e	T16B_0MZDMAEN (T16B Ch.0 Counter Max/Zero DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	MZDMAEN[3:0]	0x0	H0	R/W	
0x0020 0410	T16B_0CCCTL0 (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	
0x0020 0412	T16B_0CCR0 (T16B Ch.0 Compare/ Capture 0 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 0414	T16B_0CC0DMAEN (T16B Ch.0 Compare/ Capture 0 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC0DMAEN[3:0]	0x0	H0	R/W	
0x0020 0418	T16B_0CCCTL1 (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	
0x0020 041a	T16B_0CCR1 (T16B Ch.0 Compare/ Capture 1 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 041c	T16B_0CC1DMAEN (T16B Ch.0 Compare/ Capture 1 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC1DMAEN[3:0]	0x0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0420	T16B_0CCCTL2 (T16B Ch.0 Compare/ Capture 2 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	
0x0020 0422	T16B_0CCR2 (T16B Ch.0 Compare/ Capture 2 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 0424	T16B_0CC2DMAEN (T16B Ch.0 Compare/ Capture 2 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC2DMAEN[3:0]	0x0	H0	R/W	
0x0020 0428	T16B_0CCCTL3 (T16B Ch.0 Compare/ Capture 3 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	
0x0020 042a	T16B_0CCR3 (T16B Ch.0 Compare/ Capture 3 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 042c	T16B_0CC3DMAEN (T16B Ch.0 Compare/ Capture 3 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC3DMAEN[3:0]	0x0	H0	R/W	

0x0020 0440–0x0020 046c 16-bit PWM Timer (T16B) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0440	T16B_1CLK (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	
0x0020 0442	T16B_1CTL (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	
0x0020 0444	T16B_1MC (T16B Ch.1 Max Counter Data Register)	15–0	MC[15:0]	0xffff	H0	R/W	–
0x0020 0446	T16B_1TC (T16B Ch.1 Timer Counter Data Register)	15–0	TC[15:0]	0x0000	H0	R	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0448	T16B_1CS (T16B Ch.1 Counter Status Register)	15–8	–	0x00	–	R	–
		7–6	–	0x0	–	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	
0x0020 044a	T16B_1INTF (T16B Ch.1 Interrupt Flag Register)	15–10	–	0x00	–	R	–
		9	CAPOW3IF	0	H0	R/W	Cleared by writing 1.
		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	
0x0020 044c	T16B_1INTE (T16B Ch.1 Interrupt Enable Register)	15–10	–	0x00	–	R	–
		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	
0x0020 044e	T16B_1MZDMAEN (T16B Ch.1 Counter Max/Zero DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	MZDMAEN[3:0]	0x0	H0	R/W	
0x0020 0450	T16B_1CCCTL0 (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	
0x0020 0452	T16B_1CCR0 (T16B Ch.1 Compare/ Capture 0 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 0454	T16B_1CC0DMAEN (T16B Ch.1 Compare/ Capture 0 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC0DMAEN[3:0]	0x0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0458	T16B_1CCCTL1 (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	
0x0020 045a	T16B_1CCR1 (T16B Ch.1 Compare/ Capture 1 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 045c	T16B_1CC1DMAEN (T16B Ch.1 Compare/ Capture 1 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC1DMAEN[3:0]	0x0	H0	R/W	
0x0020 0460	T16B_1CCCTL2 (T16B Ch.1 Compare/ Capture 2 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	
0x0020 0462	T16B_1CCR2 (T16B Ch.1 Compare/ Capture 2 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 0464	T16B_1CC2DMAEN (T16B Ch.1 Compare/ Capture 2 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC2DMAEN[3:0]	0x0	H0	R/W	
0x0020 0468	T16B_1CCCTL3 (T16B Ch.1 Compare/ Capture 3 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	
0x0020 046a	T16B_1CCR3 (T16B Ch.1 Compare/ Capture 3 Data Register)	15–0	CC[15:0]	0x0000	H0	R/W	–
0x0020 046c	T16B_1CC3DMAEN (T16B Ch.1 Compare/ Capture 3 DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	CC3DMAEN[3:0]	0x0	H0	R/W	

0x0020 0480–0x0020 048c 16-bit Timer (T16) Ch.3

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0480	T16_3CLK (T16 Ch.3 Clock Control Register)	15–9	–	0x00	–	R	–
		8	DBRUN	0	H0	R/W	
		7–4	CLKDIV[3:0]	0x0	H0	R/W	
		3–2	–	0x0	–	R	
		1–0	CLKSRC[1:0]	0x0	H0	R/W	
0x0020 0482	T16_3MOD (T16 Ch.3 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 0484	T16_3CTL (T16 Ch.3 Control Register)	15–9	–	0x00	–	R	–
		8	PRUN	0	H0	R/W	
		7–2	–	0x00	–	R	
		1	PRESET	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 0486	T16_3TR (T16 Ch.3 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 0488	T16_3TC (T16 Ch.3 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 048a	T16_3INTF (T16 Ch.3 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x0020 048c	T16_3INTE (T16 Ch.3 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 04a0–0x0020 04ac 16-bit Timer (T16) Ch.4

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 04a0	T16_4CLK (T16 Ch.4 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	
0x0020 04a2	T16_4MOD (T16 Ch.4 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 04a4	T16_4CTL (T16 Ch.4 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	
0x0020 04a6	T16_4TR (T16 Ch.4 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 04a8	T16_4TC (T16 Ch.4 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 04aa	T16_4INTF (T16 Ch.4 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x0020 04ac	T16_4INTE (T16 Ch.4 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 04c0–0x0020 04cc 16-bit Timer (T16) Ch.5

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 04c0	T16_5CLK (T16 Ch.5 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	
0x0020 04c2	T16_5MOD (T16 Ch.5 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 04c4	T16_5CTL (T16 Ch.5 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	
0x0020 04c6	T16_5TR (T16 Ch.5 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 04c8	T16_5TC (T16 Ch.5 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 04ca	T16_5INTF (T16 Ch.5 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x0020 04cc	T16_5INTE (T16 Ch.5 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 04d0–0x0020 04de Synchronous Serial Interface (SPIA) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 04d0	SPIA_2MOD (SPIA Ch.2 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	
0x0020 04d2	SPIA_2CTL (SPIA Ch.2 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 04d4	SPIA_2TXD (SPIA Ch.2 Transmit Data Register)	15–0	TXD[15:0]	0x0000	H0	R/W	–
0x0020 04d6	SPIA_2RXD (SPIA Ch.2 Receive Data Register)	15–0	RXD[15:0]	0x0000	H0	R	–
0x0020 04d8	SPIA_2INTF (SPIA Ch.2 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 SPIA_2RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the SPIA_2TXD register.

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 04da	SPIA_2INTE (SPIA Ch.2 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	
0x0020 04dc	SPIA_2TBEDMAEN (SPIA Ch.2 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 04de	SPIA_2RBFDMAEN (SPIA Ch.2 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	

0x0020 0600–0x0020 0614 UART (UART3) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0600	UART3_1CLK (UART3 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	
0x0020 0602	UART3_1MOD (UART3 Ch.1 Mode Register)	15–13	–	0x00	–	R	–
		12	PECAR	0	H0	R/W	
		11	CAREN	0	H0	R/W	
		10	BRDIV	0	H0	R/W	
		9	INVRX	0	H0	R/W	
		8	INVTX	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	
0x0020 0604	UART3_1BR (UART3 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	
0x0020 0606	UART3_1CTL (UART3 Ch.1 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 0608	UART3_1TXD (UART3 Ch.1 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x0020 060a	UART3_1RXD (UART3 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
0x0020 060c	UART3_1INTF (UART3 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 UART3_1RXD 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 UART3_1RXD register.
		1	RB1FIF	0	H0/S0	R	
		0	TBEIF	1	H0/S0	R	Cleared by writing to the UART3_1TXD register.
0x0020 060e	UART3_1INTE (UART3 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	
0x0020 0610	UART3_1 TBEDMAEN (UART3 Ch.1 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 0612	UART3_1 RB1FDMAEN (UART3 Ch.1 Receive Buffer One Byte Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RB1FDMAEN[3:0]	0x0	H0	R/W	
0x0020 0614	UART3_1CAWF (UART3 Ch.1 Carrier Waveform Register)	15–8	–	0x00	–	R	–
		7–0	CRPER[7:0]	0x00	H0	R/W	

0x0020 0620–0x0020 0634 UART (UART3) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0620	UART3_2CLK (UART3 Ch.2 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	
0x0020 0622	UART3_2MOD (UART3 Ch.2 Mode Register)	15–13	–	0x00	–	R	–
		12	PECAR	0	H0	R/W	
		11	CAREN	0	H0	R/W	
		10	BRDIV	0	H0	R/W	
		9	INVRX	0	H0	R/W	
		8	INVTX	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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0624	UART3_2BR (UART3 Ch.2 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	
0x0020 0626	UART3_2CTL (UART3 Ch.2 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 0628	UART3_2TXD (UART3 Ch.2 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x0020 062a	UART3_2RXD (UART3 Ch.2 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x0020 062c	UART3_2INTF (UART3 Ch.2 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 UART3_2RXD 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 UART3_2RXD register.
		1	RB1FIF	0	H0/S0	R	
		0	TBEIF	1	H0/S0	R	Cleared by writing to the UART3_2TXD register.
0x0020 062e	UART3_2INTE (UART3 Ch.2 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	
0x0020 0630	UART3_2 TBEDMAEN (UART3 Ch.2 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 0632	UART3_2 RB1FDMAEN (UART3 Ch.2 Receive Buffer One Byte Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RB1FDMAEN[3:0]	0x0	H0	R/W	
0x0020 0634	UART3_2CAWF (UART3 Ch.2 Carrier Waveform Register)	15–8	–	0x00	–	R	–
		7–0	CRPER[7:0]	0x00	H0	R/W	

0x0020 0660–0x0020 066c 16-bit Timer (T16) Ch.6

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0660	T16_6CLK (T16 Ch.6 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	
0x0020 0662	T16_6MOD (T16 Ch.6 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0664	T16_6CTL (T16 Ch.6 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	
0x0020 0666	T16_6TR (T16 Ch.6 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 0668	T16_6TC (T16 Ch.6 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 066a	T16_6INTF (T16 Ch.6 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x0020 066c	T16_6INTE (T16 Ch.6 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 0670–0x0020 067e Synchronous Serial Interface (SPIA) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0670	SPIA_1MOD (SPIA Ch.1 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	
0x0020 0672	SPIA_1CTL (SPIA Ch.1 Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 0674	SPIA_1TXD (SPIA Ch.1 Transmit Data Register)	15–0	TXD[15:0]	0x0000	H0	R/W	–
0x0020 0676	SPIA_1RXD (SPIA Ch.1 Receive Data Register)	15–0	RXD[15:0]	0x0000	H0	R	–
0x0020 0678	SPIA_1INTF (SPIA Ch.1 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 SPIA_1RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the SPIA_1TXD register.
0x0020 067a	SPIA_1INTE (SPIA Ch.1 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	
0x0020 067c	SPIA_1TBEDMAEN (SPIA Ch.1 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 067e	SPIA_1RBFDMAEN (SPIA Ch.1 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	

0x0020 0680–0x0020 068c 16-bit Timer (T16) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0680	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	
0x0020 0682	T16_2MOD (T16 Ch.2 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 0684	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	
0x0020 0686	T16_2TR (T16 Ch.2 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 0688	T16_2TC (T16 Ch.2 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 068a	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.
0x0020 068c	T16_2INTE (T16 Ch.2 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 0690–0x0020 06a8 Quad Synchronous Serial Interface (QSPI) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0690	QSPI_0MOD (QSPI Ch.0 Mode Register)	15–12	CHDL[3:0]	0x7	H0	R/W	–
		11–8	CHLN[3:0]	0x7	H0	R/W	
		7–6	TMOD[1:0]	0x0	H0	R/W	
		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	
0x0020 0692	QSPI_0CTL (QSPI Ch.0 Control Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	DIR	0	H0	R/W	
		2	MSTSSO	1	H0	R/W	
		1	SFTRST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 0694	QSPI_0TXD (QSPI Ch.0 Transmit Data Register)	15–0	TXD[15:0]	0x0000	H0	R/W	–
0x0020 0696	QSPI_0RXD (QSPI Ch.0 Receive Data Register)	15–0	RXD[15:0]	0x0000	H0	R	–

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0698	QSPI_0INTF (QSPI Ch.0 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7	BSY	0	H0	R	
		6	MMABSY	0	H0	R	
		5–4	–	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 QSPI_0RXD register.
		0	TBEIF	1	H0/S0	R	Cleared by writing to the QSPI_0TXD register.
0x0020 069a	QSPI_0INTE (QSPI 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	
0x0020 069c	QSPI_0TBEDMAEN (QSPI Ch.0 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 069e	QSPI_0RBFDMAEN (QSPI Ch.0 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	
0x0020 06a0	QSPI_0FRLDMAEN (QSPI Ch.0 FIFO Data Ready DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	FRLDMAEN[3:0]	0x0	H0	R/W	
0x0020 06a2	QSPI_0MMACFG1 (QSPI Ch.0 Memory Mapped Access Configuration Register 1)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TCSH[3:0]	0x0	H0	R/W	
0x0020 06a4	QSPI_0RMADRH (QSPI Ch.0 Remapping Start Address High Register)	15–4	RMADR[31:20]	0x000	H0	R/W	–
		3-0	–	0x0	–	R	
0x0020 06a6	QSPI_0MMACFG2 (QSPI Ch.0 Memory Mapped Access Configuration Register 2)	15–12	DUMDL[3:0]	0x7	H0	R/W	–
		11–8	DUMLN[3:0]	0x7	H0	R/W	
		7–6	DATTMOD[1:0]	0x0	H0	R/W	
		5–4	DUMTMOD[1:0]	0x0	H0	R/W	
		3–2	ADRTMOD[1:0]	0x0	H0	R/W	
		1	ADRCYC	0	H0	R/W	
		0	MMAEN	0	H0	R/W	
0x0020 06a8	QSPI_0MB (QSPI Ch.0 Mode Byte Register)	15–8	XIPACT[7:0]	0x00	H0	R/W	–
		7–0	XIPEXT[7:0]	0x00	H0	R/W	

0x0020 06c0–0x0020 06d6 I²C (I2C) Ch.1

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 06c0	I2C_1CLK (I2C 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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 06c2	I2C_1MOD (I2C Ch.1 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	
0x0020 06c4	I2C_1BR (I2C Ch.1 Baud-Rate Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	BRT[6:0]	0x7f	H0	R/W	
0x0020 06c8	I2C_1OADR (I2C Ch.1 Own Address Register)	15–10	–	0x00	–	R	–
		9–0	OADR[9:0]	0x000	H0	R/W	
0x0020 06ca	I2C_1CTL (I2C Ch.1 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	
0x0020 06cc	I2C_1TXD (I2C Ch.1 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x0020 06ce	I2C_1RXD (I2C Ch.1 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x0020 06d0	I2C_1INTF (I2C Ch.1 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 I2C_1RXD register.
		0	TBEIF	0	H0/S0	R	Cleared by writing to the I2C_1TXD register.
0x0020 06d2	I2C_1INTE (I2C Ch.1 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	
0x0020 06d4	I2C_1TBEDMAEN (I2C Ch.1 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 06d6	I2C_1RBFDMAEN (I2C Ch.1 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	

0x0020 06e0–0x0020 06f6 I²C (I2C) Ch.2

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 06e0	I2C_2CLK (I2C Ch.2 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	
0x0020 06e2	I2C_2MOD (I2C Ch.2 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	
0x0020 06e4	I2C_2BR (I2C Ch.2 Baud-Rate Register)	15–8	–	0x00	–	R	–
		7	–	0	–	R	
		6–0	BRT[6:0]	0x7f	H0	R/W	
0x0020 06e8	I2C_2OADR (I2C Ch.2 Own Address Register)	15–10	–	0x00	–	R	–
		9–0	OADR[9:0]	0x000	H0	R/W	
0x0020 06ea	I2C_2CTL (I2C Ch.2 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	
0x0020 06ec	I2C_2TXD (I2C Ch.2 Transmit Data Register)	15–8	–	0x00	–	R	–
		7–0	TXD[7:0]	0x00	H0	R/W	
0x0020 06ee	I2C_2RXD (I2C Ch.2 Receive Data Register)	15–8	–	0x00	–	R	–
		7–0	RXD[7:0]	0x00	H0	R	
0x0020 06f0	I2C_2INTF (I2C Ch.2 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 I2C_2RXD register.
		0	TBEIF	0	H0/S0	R	Cleared by writing to the I2C_2TXD register.
0x0020 06f2	I2C_2INTE (I2C Ch.2 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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 06f4	I2C_2TBEDMAEN (I2C Ch.2 Transmit Buffer Empty DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	TBEDMAEN[3:0]	0x0	H0	R/W	
0x0020 06f6	I2C_2RBFDMAEN (I2C Ch.2 Receive Buffer Full DMA Request Enable Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	RBFDMAEN[3:0]	0x0	H0	R/W	

0x0020 0720–0x0020 0732 IR Remote Controller (REMC3)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0720	REMC3CLK (REMC3 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	
0x0020 0722	REMC3DBCTL (REMC3 Data Bit Counter Control Register)	15–10	–	0x00	–	R	–
		9	PRESET	0	H0/S0	R/W	Cleared by writing 1 to the REMC3DBCTL.REMCRST bit.
		8	PRUN	0	H0/S0	R/W	
		7–5	–	0x0	–	R	–
		4	REMOINV	0	H0	R/W	
		3	BUFEN	0	H0	R/W	
		2	TRMD	0	H0	R/W	
		1	REMCRST	0	H0	W	
		0	MODEN	0	H0	R/W	
0x0020 0724	REMC3DBCNT (REMC3 Data Bit Counter Register)	15–0	DBCNT[15:0]	0x0000	H0/S0	R	Cleared by writing 1 to the REMC3DBCTL.REMCRST bit.
0x0020 0726	REMC3APLEN (REMC3 Data Bit Active Pulse Length Register)	15–0	APLEN[15:0]	0x0000	H0	R/W	Writing enabled when REMC3DBCTL.MODEN bit = 1.
0x0020 0728	REMC3DBLEN (REMC3 Data Bit Length Register)	15–0	DBLEN[15:0]	0x0000	H0	R/W	Writing enabled when REMC3DBCTL.MODEN bit = 1.
0x0020 072a	REMC3INTF (REMC3 Status and Interrupt Flag Register)	15–11	–	0x00	–	R	–
		10	DBCNTRUN	0	H0/S0	R	Cleared by writing 1 to the REMC3DBCTL.REMCRST bit.
		9	DBLENBSY	0	H0	R	Effective when the REMC3DBCTL.BUFEN bit = 1.
		8	APLENBSY	0	H0	R	
		7–2	–	0x00	–	R	–
		1	DBIF	0	H0/S0	R/W	Cleared by writing 1 to this bit or the REMC3DBCTL. REMCRST bit.
		0	APIF	0	H0/S0	R/W	
0x0020 072c	REMC3INTE (REMC3 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	DBIE	0	H0	R/W	
		0	APIE	0	H0	R/W	
0x0020 0730	REMC3CARR (REMC3 Carrier Waveform Register)	15–8	CRDTY[7:0]	0x00	H0	R/W	–
		7–0	CRPER[7:0]	0x00	H0	R/W	
0x0020 0732	REMC3CCTL (REMC3 Carrier Modulation Control Register)	15–9	–	0x00	–	R	–
		8	OUTINVEN	0	H0	R/W	
		7–1	–	0x00	–	R	
		0	CARREN	0	H0	R/W	

0x0020 0780–0x0020 078c 16-bit Timer (T16) Ch.7

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0780	T16_7CLK (T16 Ch.7 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	
0x0020 0782	T16_7MOD (T16 Ch.7 Mode Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	TRMD	0	H0	R/W	
0x0020 0784	T16_7CTL (T16 Ch.7 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	
0x0020 0786	T16_7TR (T16 Ch.7 Reload Data Register)	15–0	TR[15:0]	0xffff	H0	R/W	–
0x0020 0788	T16_7TC (T16 Ch.7 Counter Data Register)	15–0	TC[15:0]	0xffff	H0	R	–
0x0020 078a	T16_7INTF (T16 Ch.7 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIF	0	H0	R/W	Cleared by writing 1.
0x0020 078c	T16_7INTE (T16 Ch.7 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	UFIE	0	H0	R/W	

0x0020 07a0–0x0020 07bc 12-bit A/D Converter (ADC12A) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 07a2	ADC12A_0CTL (ADC12A Ch.0 Control Register)	15	–	0	–	R	–
		14–12	ADSTAT[2:0]	0x0	H0	R	
		11	–	0	–	R	
		10	BSYSTAT	0	H0	R	
		9–8	–	0x0	–	R	
		7–2	–	0x00	–	R	
		1	ADST	0	H0	R/W	
		0	MODEN	0	H0	R/W	
0x0020 07a4	ADC12A_0TRG (ADC12A Ch.0 Trigger/Analog Input Select Register)	15–14	–	0x0	–	R	–
		13–11	ENDAIN[2:0]	0x0	H0	R/W	
		10–8	STAAIN[2:0]	0x0	H0	R/W	
		7	STMD	0	H0	R/W	
		6	CNVMD	0	H0	R/W	
		5–4	CNVTRG[1:0]	0x0	H0	R/W	
		3	–	0	–	R	
		2–0	SMPCLK[2:0]	0x7	H0	R/W	
0x0020 07a6	ADC12A_0CFG (ADC12A Ch.0 Configuration Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1–0	VRANGE[1:0]	0x0	H0	R/W	
0x0020 07a8	ADC12A_0INTF (ADC12A Ch.0 Interrupt Flag Register)	15–9	–	0x00	–	R	–
		8	OVIF	0	H0	R/W	Cleared by writing 1.
		7	AD7CIF	0	H0	R/W	
		6	AD6CIF	0	H0	R/W	
		5	AD5CIF	0	H0	R/W	
		4	AD4CIF	0	H0	R/W	
		3	AD3CIF	0	H0	R/W	
		2	AD2CIF	0	H0	R/W	
		1	AD1CIF	0	H0	R/W	
		0	AD0CIF	0	H0	R/W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 07aa	ADC12A_0INTE (ADC12A Ch.0 Interrupt Enable Register)	15–9	–	0x00	–	R	–
		8	OVIE	0	H0	R/W	
		7	AD7CIE	0	H0	R/W	
		6	AD6CIE	0	H0	R/W	
		5	AD5CIE	0	H0	R/W	
		4	AD4CIE	0	H0	R/W	
		3	AD3CIE	0	H0	R/W	
		2	AD2CIE	0	H0	R/W	
		1	AD1CIE	0	H0	R/W	
		0	AD0CIE	0	H0	R/W	
0x0020 07ac	ADC12A_0DMAEN0 (ADC12A Ch.0 DMA Request Enable Register 0)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07ae	ADC12A_0DMAEN1 (ADC12A Ch.0 DMA Request Enable Register 1)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07b0	ADC12A_0DMAEN2 (ADC12A Ch.0 DMA Request Enable Register 2)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07b2	ADC12A_0DMAEN3 (ADC12A Ch.0 DMA Request Enable Register 3)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07b4	ADC12A_0DMAEN4 (ADC12A Ch.0 DMA Request Enable Register 4)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07b6	ADC12A_0DMAEN5 (ADC12A Ch.0 DMA Request Enable Register 5)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07b8	ADC12A_0DMAEN6 (ADC12A Ch.0 DMA Request Enable Register 6)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07ba	ADC12A_0DMAEN7 (ADC12A Ch.0 DMA Request Enable Register 7)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3–0	ADCDMAEN[3:0]	0x0	H0	R/W	
0x0020 07bc	ADC12A_0ADD (ADC12A Ch.0 Result Register)	15–0	ADD[15:0]	0x0000	H0	R	–

0x0020 07c0–0x0020 07c2 Temperature Sensor/Reference Voltage Generator (TSRVR) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 07c0	TSRVR_0TCTL (TSRVR Ch.0 Temperature Sensor Control Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	H0	R	
		0	TEMPEN	0	H0	R/W	
0x0020 07c2	TSRVR_0VCTL (TSRVR Ch.0 Reference Voltage Generator Control Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	H0	R	
		1–0	VREFAMD[1:0]	0x0	H0	R/W	

0x0020 0840–0x0020 0850 R/F Converter (RFC) Ch.0

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0840	RFC_0CLK (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	
0x0020 0842	RFC_0CTL (RFC Ch.0 Control Register)	15–9	–	0x00	–	R	–
		8	RFCLKMD	0	H0	R/W	
		7	CONEN	0	H0	R/W	
		6	EVTEN	0	H0	R/W	
		5–4	SMODE[1:0]	0x0	H0	R/W	
		3–1	–	0x0	–	R	
		0	MODEN	0	H0	R/W	
0x0020 0844	RFC_0TRG (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	
0x0020 0846	RFC_0MCL (RFC Ch.0 Measurement Counter Low Register)	15–0	MC[15:0]	0x0000	H0	R/W	–
0x0020 0848	RFC_0MCH (RFC Ch.0 Measurement Counter High Register)	15–8	–	0x00	–	R	–
		7–0	MC[23:16]	0x00	H0	R/W	
0x0020 084a	RFC_0TCL (RFC Ch.0 Time Base Counter Low Register)	15–0	TC[15:0]	0x0000	H0	R/W	–
0x0020 084c	RFC_0TCH (RFC Ch.0 Time Base Counter High Register)	15–8	–	0x00	–	R	–
		7–0	TC[23:16]	0x00	H0	R/W	
0x0020 084e	RFC_0INTF (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	
0x0020 0850	RFC_0INTE (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	

0x0020 0860–0x0020 087e Sound DAC (SDAC2)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0860	SDAC2CLK (SDAC2 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	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 0862	SDAC2CTL (SDAC2 Control Register)	15–8	–	0x00	–	R	–
		7–4	–	0x00	–	R	
		3	TONEON	0	H0	R/W	
		2	–	0	–	R	
		1	RESAMPEN	0	H0	R/W	
		0	SDACEN	0	H0	R/W	
0x0020 0864	SDAC2MOD (SDAC2 Mode Register)	15–9	–	0x00	–	R	–
		8	PWMOUTEN	0	H0	R/W	
		7–2	–	0x00	H0	R	
		1–0	PWMMODE[1:0]	0x00	H0	R/W	
0x0020 0866	SDAC2_0DAT (SDAC2 Ch.0 Data Register)	15–10	–	0x00	–	R	–
		9–0	DAT[9:0]	0x000	H0	R/W	
0x0020 0868	SDAC2INTF (SDAC2 Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–4	–	0x0	–	R	
		3	ERR1IF	0	H0	R/W	Cleared by writing 1.
		2	DATREQ1IF	0	H0	R/W	
		1	ERR0IF	0	H0	R/W	
		0	DATREQ0IF	0	H0	R/W	
0x0020 086a	SDAC2INTE (SDAC2 Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		3	ERR1IE	0	H0	R/W	
		2	DATREQ1IE	0	H0	R/W	
		1	ERR0IE	0	H0	R/W	
		0	DATREQ0IE	0	H0	R/W	
0x0020 0870	SDAC2RESAMP (SDAC2 Resampler Rate Register)	15–11	–	0x00	–	R	–
		10–0	RESAMPRATE[10:0]	0x400	H0	R/W	
0x0020 0878	SDAC2TONE (SDAC2 Tone Divider Register)	15–0	TONEDIV[15:0]	0x4000	H0	R/W	–
0x0020 087e	SDAC2_1DAT (SDAC2 Ch.1 Data Register)	15–10	–	0x00	–	R	–
		9–0	DAT[9:0]	0x000	H0	R/W	

0x0020 08a0–0x0020 08a8 HW Processor (HWP)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 08a2	HWPCTL (HWP Control Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	HWPEN	0	H0	R/W	
0x0020 08a4	HWPINTF (HWP Interrupt Flag Register)	15–8	–	0x00	–	R	–
		7–2	–	0x00	–	R	
		1	HWP1IF	0	H0	R/W	Cleared by writing 0.
		0	HWP0IF	0	H0	R/W	
0x0020 08a6	HWPINTE (HWP Interrupt Enable Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	HWPIE	0	H0	R/W	
0x0020 08a8	HWPCMDTRG (HWP Command Trigger Register)	15–8	–	0x00	–	R	–
		7–1	–	0x00	–	R	
		0	HWP0TRG	0	H0	R/W	

0x0020 1000–0x0020 2014 DMA Controller (DMAC)

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 1000	DMACSTAT (DMAC Status Register)	31–24	–	0x00	–	R	–
		23–21	–	0x0	–	R	
		20–16	CHNLS[4:0]	*	H0	R	* Number of channels implemented - 1
		15–8	–	0x00	–	R	–
		7–4	STATE[3:0]	0x0	H0	R	
		3–1	–	0x0	–	R	
		0	MSTENSTAT	0	H0	R	
0x0020 1004	DMACCFG (DMAC Configuration Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–1	–	0x00	–	R	
		0	MSTEN	–	–	W	
0x0020 1008	DMACCPTR (DMAC Control Data Base Pointer Register)	31–7	CPTR[31:7]	0x000 0000	H0	R/W	–
		6–0	CPTR[6:0]	0x00	H0	R	
0x0020 100c	DMACACPTR (DMAC Alternate Control Data Base Pointer Register)	31–0	ACPTR[31:0]	–	H0	R	–
0x0020 1014	DMACSWREQ (DMAC Software Request Register)	31–24	–	–	–	R	–
		23–16	–	–	–	R	
		15–8	–	–	–	R	
		7–4	–	–	–	R	
		3–0	SWREQ[3:0]	–	–	W	
0x0020 1020	DMACRMSET (DMAC Request Mask Set Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–4	–	0x0	–	R	
		3–0	RMSET[3:0]	0x0	H0	R/W	
0x0020 1024	DMACRMCLR (DMAC Request Mask Clear Register)	31–24	–	–	–	R	–
		23–16	–	–	–	R	
		15–8	–	–	–	R	
		7–4	–	–	–	R	
		3–0	RMCLR[3:0]	–	–	W	
0x0020 1028	DMACENSET (DMAC Enable Set Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–4	–	0x0	–	R	
		3–0	ENSET[3:0]	0x0	H0	R/W	
0x0020 102c	DMACENCLR (DMAC Enable Clear Register)	31–24	–	–	–	R	–
		23–16	–	–	–	R	
		15–8	–	–	–	R	
		7–4	–	–	–	R	
		3–0	ENCLR[3:0]	–	–	W	
0x0020 1030	DMACPASET (DMAC Primary-Alternate Set Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–4	–	0x0	–	R	
		3–0	PASET[3:0]	0x0	H0	R/W	
0x0020 1034	DMACPACLR (DMAC Primary-Alternate Clear Register)	31–24	–	–	–	R	–
		23–16	–	–	–	R	
		15–8	–	–	–	R	
		7–4	–	–	–	R	
		3–0	PACLR[3:0]	–	–	W	

Address	Register name	Bit	Bit name	Initial	Reset	R/W	Remarks
0x0020 1038	DMACPRSET (DMAC Priority Set Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–4	–	0x0	–	R	
		3–0	PRSET[3:0]	0x0	H0	R/W	
0x0020 103c	DMACPRCLR (DMAC Priority Clear Register)	31–24	–	–	–	R	–
		23–16	–	–	–	R	
		15–8	–	–	–	R	
		7–4	–	–	–	R	
		3–0	PRCLR[3:0]	–	–	W	
0x0020 104c	DMACERRIF (DMAC Error Interrupt Flag Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–1	–	0x00	–	R	
		0	ERRIF	0	H0	R/W	Cleared by writing 1.
0x0020 2000	DMACENDIF (DMAC Transfer Completion Interrupt Flag Register)	31–24	–	0x00	–	R	
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–4	–	0x0	–	R	
		3–0	ENDIF[3:0]	0x0	H0	R/W	Cleared by writing 1.
0x0020 2008	DMACENDIESET (DMAC Transfer Completion Interrupt Enable Set Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–4	–	0x0	–	R	
		3–0	ENDIESET[3:0]	0x0	H0	R/W	
0x0020 200c	DMACENDIECLR (DMAC Transfer Completion Interrupt Enable Clear Register)	31–24	–	–	–	R	–
		23–16	–	–	–	R	
		15–8	–	–	–	R	
		7–4	–	–	–	R	
		3–0	ENDIECLR[3:0]	–	–	W	
0x0020 2010	DMACERRIESET (DMAC Error Interrupt Enable Set Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–1	–	0x00	–	R	
		0	ERRIESET	0	H0	R/W	
0x0020 2014	DMACERRIECLR (DMAC Error Interrupt Enable Clear Register)	31–24	–	0x00	–	R	–
		23–16	–	0x00	–	R	
		15–8	–	0x00	–	R	
		7–1	–	0x00	–	R	
		0	ERRIECLR	–	–	W	

Appendix B Power Saving

Current consumption will vary dramatically, depending on CPU operating mode, operation clock frequency, peripheral circuits being operated, and V_{D1} regulator operating mode. Listed below are the control methods for saving power.

B.1 Operating Status Configuration Examples for Power Saving

Table B.1.1 lists typical examples of operating status configuration with consideration given to power saving.

Table B.1.1 Typical Operating Status Configuration Examples

<table>
<tr><th>Operating status configuration</th><th>Current consumption</th><th>V_{D1}</th><th>OSC1</th><th>IOSC/ OSC3/ EXOSC</th><th>RTCA</th><th>CPU</th><th>Current consumption listed in electrical characteristics</th></tr>
<tr><td>Standby</td><td rowspan="3">↑
Low</td><td rowspan="3">Economy</td><td>OFF</td><td rowspan="3">OFF</td><td>OFF</td><td>SLEEP</td><td>$I_{SLP1–2}$</td></tr>
<tr><td>Clock counting</td><td rowspan="4">ON</td><td rowspan="4">ON</td><td>SLEEP with OSC1SLPC</td><td>$I_{SLP3–4}$</td></tr>
<tr><td>Low-speed processing</td><td>OSC1 RUN</td><td>$I_{RUN4–5}$</td></tr>
<tr><td>Peripheral circuit operations</td><td rowspan="2">High
↓</td><td rowspan="2">Normal</td><td rowspan="2">ON</td><td>SLEEP or HALT</td><td>$I_{HALT4–5}$</td></tr>
<tr><td>High-speed processing</td><td>IOSC/OSC3/EXOSC RUN</td><td>$I_{RUN1–3, 6–7}$</td></tr>
</table>

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.

PWGACTL.REGMODE[1:0] bits of the power generator

If the PWGACTL.REGMODE[1:0] bits of the power generator is 0x2 (normal mode) when the CPU enters SLEEP mode, current consumption in SLEEP mode will be larger than I_{SLP} that is listed in "Electrical Characteristics." Set the PWGACTL.REGMODE[1:0] bits to 0x3 (economy mode) or 0x0 (automatic mode) before placing the CPU into SLEEP mode.

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 CPU enters SLEEP mode. 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 (crystal) oscillator circuit configurations

The OSC1 (crystal) 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 (SVD3CTL.SVDMD[1:0] bits = 0x0) always detects the power supply voltage, therefore, it increases current consumption. Set the supply voltage detector to intermittent operation mode or turn it on only when required.

Appendix C Mounting Precautions

This section describes various precautions for circuit board design and IC mounting.

OSC1/OSC3 oscillator circuit

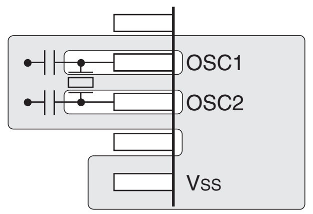
- Oscillation characteristics depend on factors such as components used (resonator, C_G, C_D) and circuit board patterns. In particular, with crystal resonators, select the appropriate capacitors (C_G, C_D) only after fully evaluating components actually mounted on the circuit board.
- Oscillator clock 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 operations. Jitter in the OSC3 output will reduce operating frequencies.

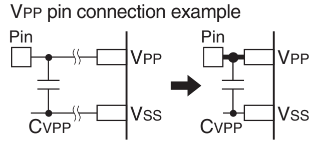
#RESET pin

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

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



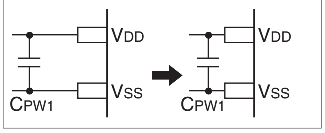
Power supply circuit

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.

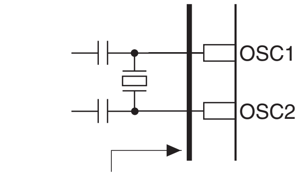
Bypass capacitor connection example



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.

Prohibited pattern



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 crystal oscillator circuit is not used, the OSC1 and OSC2 pins should be left open. If the OSC3 crystal/ceramic oscillator circuit or EXOSC input circuit is not used, the pin should be configured as a general-purpose I/O port. The control registers should be fixed at the initial status (disabled).

Miscellaneous

Minor variations over time may result in electrical damage arising from disturbances in the form of voltages exceeding the absolute maximum rating when mounting the product in addition to physical damage. The following factors can give rise to these variations:

(1) Electromagnetically-induced noise from industrial power supplies used in mounting reflow, reworking after mounting, and individual characteristic evaluation (testing) processes
(2) Electromagnetically-induced noise from a solder iron when soldering

In particular, during soldering, take care to ensure that the soldering iron GND (tip potential) has the same potential as the IC GND.

Appendix D Measures Against Noise

To improve noise immunity, take measures against noise as follows:

Noise Measures for V_{DD}, V_{DDQSPI}, 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 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.

Noise Measures for Input Pins Connected to Signal with High Driving Capability Such As Power Supply

There is a possibility of a large current flow into the pins that are directly connected to a power supply or an output of a device with high driving capability if noise is input to those pins. To prevent this, connect a 30 Ω or more pin protection resistor to the pins in series. The resistance value should be determined by evaluating it on the mounting board.

When connecting a power supply directly to the VREFA pin, insert a 100 Ω resistor in series. This resistance does not affect the A/D converter characteristics.

Revision History

Code No.	Page	Contents
414190500	All	New establishment
414190501	1-3	1.1 Features Added the following annotation to Table 1.1.1. *2 SLEEP mode refers to deep sleep mode in the Cortex®-M0+ processor. The RAM retains data even in SLEEP mode.
	2-14	2.4.2 Transition between Operating Modes SLEEP mode Added the following description: The RAM retains data even in SLEEP mode.
	4-2	4.3.1 Flash Memory Pin Deleted the following description: ~~For the V_{PP} voltage, refer to “Recommended Operating Conditions, Flash programming voltage V_{PP}” in the “Electrical Characteristics” chapter.~~
	7-6	7.4.2 Port Input/Output Control Chattering filter function The equation number was corrected. (Eq. 6.2) → (Eq. 7.2)
	15-1	15.1 Overview Corrected the description. - 1M-byte external Flash memory mapped access area that allows programmable re-mapping. Added an item to Table 15.1.1. Memory mapped access area for external Flash memory: 1M-byte area beginning with address 0x0004_0000
	15-8, 9	15.4 Data Format Figures 15.4.1 and 15.4.2 Added the following bit setting: QSPI_*n*MOD.CHDL[3:0] bits = 0x7 Figure 15.4.3 Added the following bit setting: QSPI_*n*MOD.CHDL[3:0] bits = 0x3
	15-10, 11	15.5.2 Memory Mapped Access Mode Figures 15.5.2.1 and 15.5.2.2 Corrected the description. The QSPI treats the dummy cycle as 6 cycles including 1 driving cycle. (QSPI_*n*MMACFG2.DUMDL[3:0] bits = 0x0, QSPI_*n*MMACFG2.DUMLN[3:0] bits = 0x5) The QSPI treats the data cycle as 2 cycles including 2 driving cycles. (QSPI_*n*MOD.CHDL[3:0] bits = 0x1, QSPI_*n*MOD.CHLN[3:0] bits = 0x1)
	15-11	15.5.2 Memory Mapped Access Mode Corrected the description. The memory mapped access area for external Flash memory in the system memory area is used to map the external Flash memory and to access from the CPU.
	15-17	15.5.6 Data Reception in Memory Mapped Access Mode Data receiving procedure Corrected the description. 4. Read the memory mapped access area for external Flash memory with an 8, 16, or 32-bit memory read instruction. This operation directly reads data within the 1M-byte external Flash memory area remapped to the memory mapped access area for external Flash memory at Step 2.
	15-29	15.8 Control Registers QSPI Ch.*n* Mode Register Deleted the following description of the CHDL[3:0] bits: ~~This setting is required to output the XIP confirmation bit to Micron Flash memories or to output the mode byte to Spansion Flash memories.~~
	15-35	15.8 Control Registers QSPI Ch.*n* Memory Mapped Access Configuration Register 2 Modified the register table. DUMDL[3:0], DUMLN[3:0]: Initial = 0x0 → 0x7
	15-37	15.8 Control Registers QSPI Ch.*n* Mode Byte Register Added the following description to the XIPEXT[7:0] bits: However, set these bits as follows when the HW processor (HWP) is used: • Before enabling the HWP, set to the same value as the QSPI_*n*MB.XIPACT[7:0] bits. • Before disabling the HWP, set to the mode byte for terminating the XIP session.

REVISION HISTORY

Code No.	Page	Contents
414190501	16-7	16.4.3 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.
	16-9	16.4.3 Data Reception in Master Mode Data reception using DMA Corrected the description. This automates the data receiving procedure Steps 6, 8, and 10 described above.
	22-17	22.4.3 External QSPI Flash Memory Access Added a new section.
	22-30	22.7 Control Registers SDAC2 Control Register Added a description on “Bit 0 SDACEN.”
	AP-A-37	Appendix A List of Peripheral Circuit Control Registers QSPI_0MMACFG2 (QSPI Ch.0 Memory Mapped Access Configuration Register 2) Modified the register table. DUMDL[3:0], DUMLN[3:0]: Initial = 0x0 → 0x7

EPSON

International Sales Operations

America

Epson America, Inc.

Headquarter:
3840 Kilroy Airport Way
Long Beach, California 90806-2452 USA
Phone: +1-562-290-4677

San Jose Office:
214 Devcon Drive
San Jose, CA 95112 USA
Phone: +1-800-228-3964 or +1-408-922-0200

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 1701 & 1704, 17 Floor, Greenland Center II,
562 Dong An Road, Xu Hui District, Shanghai, 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: +86-10-3299-0588 Fax: +86-10-3299-0560

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.

1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 Fax: +65-6271-3182

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

Device Sales & Marketing Department

29th Floor, JR Shinjuku Miraina Tower, 4-1-6 Shinjuku,
Shinjuku-ku, Tokyo 160-8801, Japan

Document Code: 414190501
First Issue July 2021
Revised October 2021 in JAPAN Ⓛ