EPSON
EXCEED YOUR VISION

CMOS 16-BIT SINGLE CHIP MICROCONTROLLER

S1C17501
Technical Manual

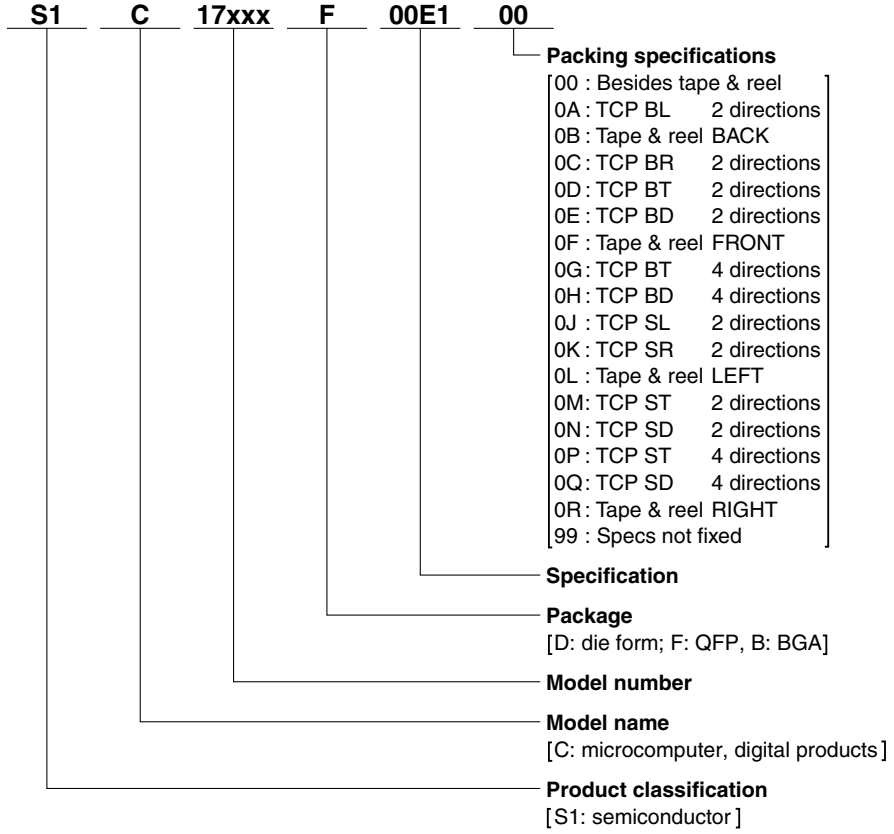
SEIKO EPSON CORPORATION

Rev. 1.1

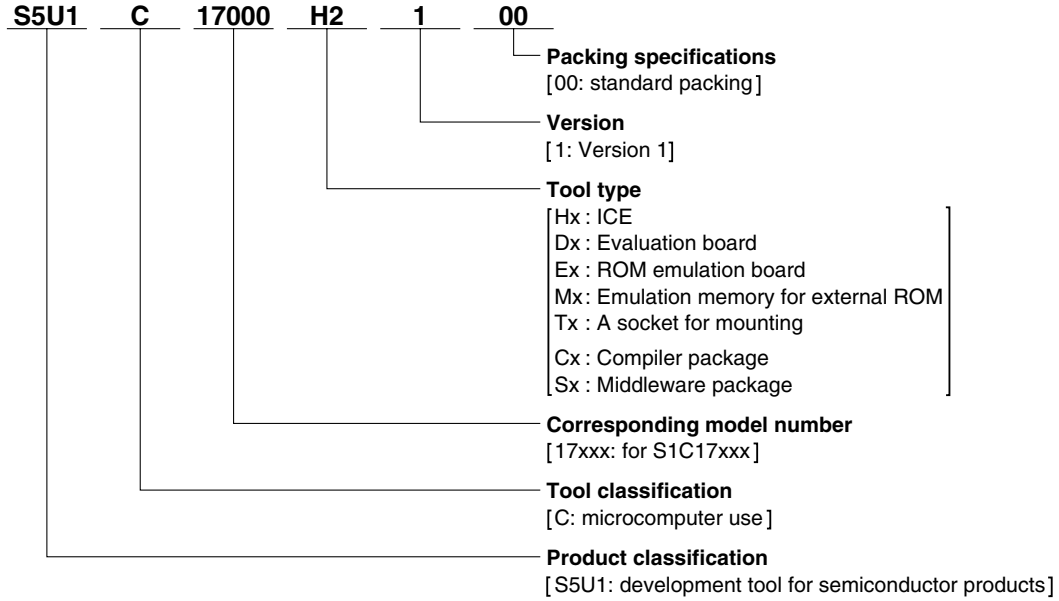
NOTICE

Configuration of product number

Devices



Development tools



– Contents –

I S1C17501 SPECIFICATIONS

II S1C17501 CLOCK SYSTEM

III S1C17501 MEMORY CONTROLLER MODULES

IV S1C17501 INTERRUPT SYSTEM

V S1C17501 TIMER MODULES

VI S1C17501 INTERFACE MODULES

VII S1C17501 I/O PORTS

VIII S1C17501 ANALOG MODULE

IX S1C17501 USB MODULE

APPENDIX

I S1C17501 SPECIFICATIONS

I.1 Overview

The S1C17501 is a high performance and compact 16-bit RISC application specific controller (ASC). It is suitable for various products that require analog inputs, display and interfaces for connection, such as healthcare goods, sensor systems, alarms, home electric appliance (rice cookers, microwave ovens and remote controllers).
The S1C17501 consists of a S1C17 16-bit compact RISC CPU Core, a 128K- or 96K-byte Flash EEPROM, a 4K + 2K-byte RAM, a 10-bit ADC with eight analog input channels, a USB FS-device controller, a 16-bit multi-function timer, an infrared remote controller, serial interfaces (UART with IrDA 1.0, SPI, I²C, and I²S), an RTC driven with an independent power supply, 16-bit and 8-bit timers, a watchdog timer, a NAND Flash card interface, an external bus with an SRAM controller, GPIO ports.
The USB FS-device controller may be used not only for communication with PCs but also for on-board firmware update.
The S1C17501 provides a 16 bits × 16 bits + 32 bits MAC (multiply and accumulate) instruction to implement a DSP function. Furthermore, an external ADC/DAC may be connected via the built-in I²S interface, this makes it possible to input/output ADPCM sound/voice data.
The S1C17501 has adopted the EPSON SoC (System on Chip) design technology using 0.35 μm mixed analog low power CMOS process.

This product uses SuperFlash® Technology licensed from Silicon Storage Technology, Inc.

Table I.1.1 Product Lineup

	Flash ROM size	RAM size	Package
1	128K bytes	4K + 2K* bytes	TQFP15-128pin
2	96K bytes		TQFP14-100pin

* Battery backup by separated power can be possible as for the 2K-byte RAM.

The main functions and features of the S1C17501 are outlined below.

Technology

- 0.35 μm AL-4-layers mixed analog low power CMOS process technology

CPU

- Seiko Epson original 16-bit RISC processor S1C17 Core
- Internal 3-stage pipeline
- Instruction set
  - 16-bit fixed length
  - 111 basic instructions (184 including variations)
  - Compact and fast instruction set optimized for development in C language
- Registers
  - Eight 24-bit general-purpose registers
  - Three special registers (24-bit × 2, 8-bit × 1)
- Memory space
  - Up to 16M bytes accessible (24-bit address)

Internal Memories

- Flash EEPROM
  - 128K bytes or 96K bytes
- RAM
  - 4K bytes
  - 2K bytes (Usable as a general-purpose RAM with battery backup feature)

Access Cycles

- Instruction read access cycle
  - Internal RAM Instruction read: 2 cycles (32-bit read)
  - Internal Flash EEPROM Instruction read: 2 cycles (32-bit read) when pre-fetched data is hit
    3 cycles (32-bit read) when pre-fetched data is missed
  - Internal RAM2 Instruction read: 7 cycles (32-bit read)
  - External 8-bit RAM Instruction read: 13 cycles (32-bit read)
  - External 16-bit RAM Instruction read: 7 cycles (32-bit read)

  * The numbers of cycles listed above are assumed when reading two instructions (16 bits × 2) in sequential access.
  * Note that the number of external RAM access cycles depends on the specifications of the RAM and the above list shows the minimum value.
- Data read/write access cycle
  - Internal RAM Data write: 1 cycle
    Data read: 2 cycles
  - Internal Flash EEPROM Data read: 1 cycle (16-bit read) when pre-read data is hit
    2 cycles (16-bit read) when pre-read data is missed
  - Internal RAM2 Data write: 4 cycles (16-bit write)
    Data read: 4 cycles (16-bit read)
  - External 8-bit RAM Data write: 7 cycles (16-bit write)
    Data read: 7 cycles (16-bit read)
  - External 16-bit RAM Data write: 4 cycles (16-bit write)
    Data read: 4 cycles (16-bit read)
- Branch penalty cycle in one cycle mode for the internal Flash EEPROM

Current address → branch address	Number of penalty cycles when a 3-cycle branch instruction is executed	Number of penalty cycles when a 4-cycle branch instruction is executed
4-byte boundary → 4-byte boundary	+2 cycles	+1 cycle
4-byte boundary → 2-byte boundary	+3 cycles	+2 cycles
2-byte boundary → 4-byte boundary	+3 cycles	+1 cycle
2-byte boundary → 2-byte boundary	+4 cycles	+2 cycles

- Maximum operating frequency in one cycle mode for the internal Flash EEPROM: 20 MHz

Operating Clock

- Main clock
  - 48 MHz when the USB function is used
  - 1 to 48 MHz (can be divided by 1 to 32) or 32.768 kHz when the USB function is not used
  - On-chip oscillator (crystal or ceramic) or external clock input
- Sub clock
  - 32.768 kHz (typ.) for the RTC
  - On-chip oscillator (crystal)

SRAM Controller

- Provides a 23-bit external address bus, an 8- or 16-bit width selectable data bus, and four chip enable signals to support a maximum of 15M-byte external memory space.

Interrupt Controller

- Four non-maskable interrupts
  - Reset (#RESET pin or watchdog timer)
  - Address misaligned
  - Debug
  - NMI (#NMI pin or watchdog timer)
- 28 maskable interrupts
  - Port inputs (eight systems)
  - 16-bit multi-function timer (one system)
  - A/D converter (two systems)
  - 16-bit timer of clock generator (one system)
  - 8-bit timers of clock generator (two systems)
  - UART (one system)
  - SPI (one system)
  - I^2C (one system)
  - RTC (one system)
  - 8-bit programmable timers (four systems)
  - Extended SPI (one system)
  - USB function controller (one system)
  - I^2S (two systems)
  - Remote controller (one system)
  - The interrupt level (priority) of each maskable interrupt system is configurable (levels 0 to 7).

Prescaler

- Generates the source clocks for the clock generator.

16-bit Multi-Function Timer

- One channel of 16-bit timer/counter with PWM output function is available.
- Can generate two compare-match interrupts.
- Supports the IGBT output control function using the A/D converter out-of-range signal.

Clock Generator

- One channel of 16-bit timer and two channels of 8-bit timers are available.
- Can be used as the clock source for the UART, SPI, and I^2C.
- Each timer can generate an underflow interrupt.

8-bit Programmable Timers

- Four channels of 8-bit timers (presettable down counter) are available.
- Can be used as an interval timer to trigger the ADC.
- Each timer can generate an underflow interrupt.

Watchdog Timer

- 30-bit watchdog timer to generate a reset or an NMI
- The watchdog timer overflow period (reset or NMI interrupt period) is programmable.
- The watchdog timer overflow signal can be output outside the IC.

RTC

- Contains time counters (second, minute, and hour) and calendar counters (day, day of the week, month, and year).
- The power source separated with the system power supply (V_{DD}) can be used.
- Provides the WAKEUP output pin and #STBY input pin to control standby mode.
- Periodic interrupts are possible.

UART

- One channel of UART is available.
- Supports IrDA 1.0 interface.
- Two-byte receive data buffer and one-byte transmit buffer are built in to support full-duplex communication.
- Transfer rate: 150 to 460800 bps, character length: seven or eight bits, parity mode: even, odd, or no parity, stop bit: one or two bits
- Parity error, framing error, and overrun error detectable
- Each channel can generate receive buffer full, transmit buffer empty, and receive error interrupts.

SPI

- Supports both master and slave modes.
- One-byte receive data buffer and one-byte transmit buffer are built in.
- Data length: eight bits fixed (MSB first)
- Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
- Can generate receive buffer full and transmit buffer empty interrupts.

Extended SPI

- Supports both master and slave modes.
- One-byte receive data buffer and one-byte transmit buffer are built in.
- Data length: eight bits fixed (MSB first)
- Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
- Can generate receive buffer full and transmit buffer empty interrupts.
- Exclusive clock source is available.

I^2C

- Supports master mode only.
- Data format: 8 bits (MSB first)
- Addressing mode: 7-bit addressing (10-bit addressing is not supported.)
- Supports the noise reject function controlled by a register.
- Can generate an I^2C interrupt.

I^2S

- Supports universal audio I^2S Bus Interface.
- One I^2S output channel in 24-bit resolution and one I^2S input channel in 16-bit resolution
- Operates as the master to generate the bit clock, word-select signal, data and master clock.
- Can generate an I^2S interrupt.

USB Function Controller

- Supports USB2.0 full speed (12M bps) mode.
- Supports auto negotiation function.
- Scratch and variable bulk end point size
- Embedded 1K-byte programmable FIFO
- Can generate a USB interrupt.

CARD Interface

- Generates 8- or 16-bit NAND Flash interface signals.
- The ECC function should be implemented in the application program.

Infrared Remote Controller

- Outputs a modulated carrier signal and inputs remote control pulses.
- Embedded carrier signal generator and data length counter.
- Can generates REMC interrupts.

General-Purpose I/O Ports

- Maximum 91 I/O ports and eight input ports are available.
- Can generate input interrupts from the eight ports selected with software.

∗ The GPIO ports are shared with other peripheral function pins (UART, PWM etc.). Therefore, the number of GPIO ports depends on the peripheral functions used.

A/D Converter

- 10-bit A/D converter with up to eight analog input ports
- Can generates an end of conversion interrupt and an out of range interrupt.
- Outputs an out of range signal to the IGBT circuit in the 16-bit multi-function timer module.

Operating Voltage

- V_{DD}: 3.00 to 3.60 V (3.3 V typ.)
- $RTCV_{DD}$: 3.00 to 3.60 V (3.3 V typ.)
- AV_{DD} (I/O): 2.70 to 5.50 V

I/O Interface Voltage

- V_{DD}
  (41 GPIO ports support -0.3 to 5.8 V input voltage.)

Operating Temperatures

- During Flash reading: -40 to 85°C
- During Flash erasing/programming: -40 to 70°C
- During USB operation: 0 to 70°C

Power Consumption

- During SLEEP: 20 μW (typ.)
- During HALT: 53 mW (typ.) in 48 MHz/3.3 V operation
- During execution: 122 mW (typ.) in 48 MHz/3.3 V operation
- Battery backup power: 0.28 μW (typ.) 3.3 V, OSC1 not used

∗ By controlling the clocks through the Clock-Gear (CMU), power consumption can be reduced.

Shipping Form

- TQFP15-128pin (14 mm × 14 mm × 1.2 mm, 0.4 mm pin pitch)
- TQFP14-100pin (12 mm × 12 mm × 1.2 mm, 0.4 mm pin pitch)

I.2 Block Diagram

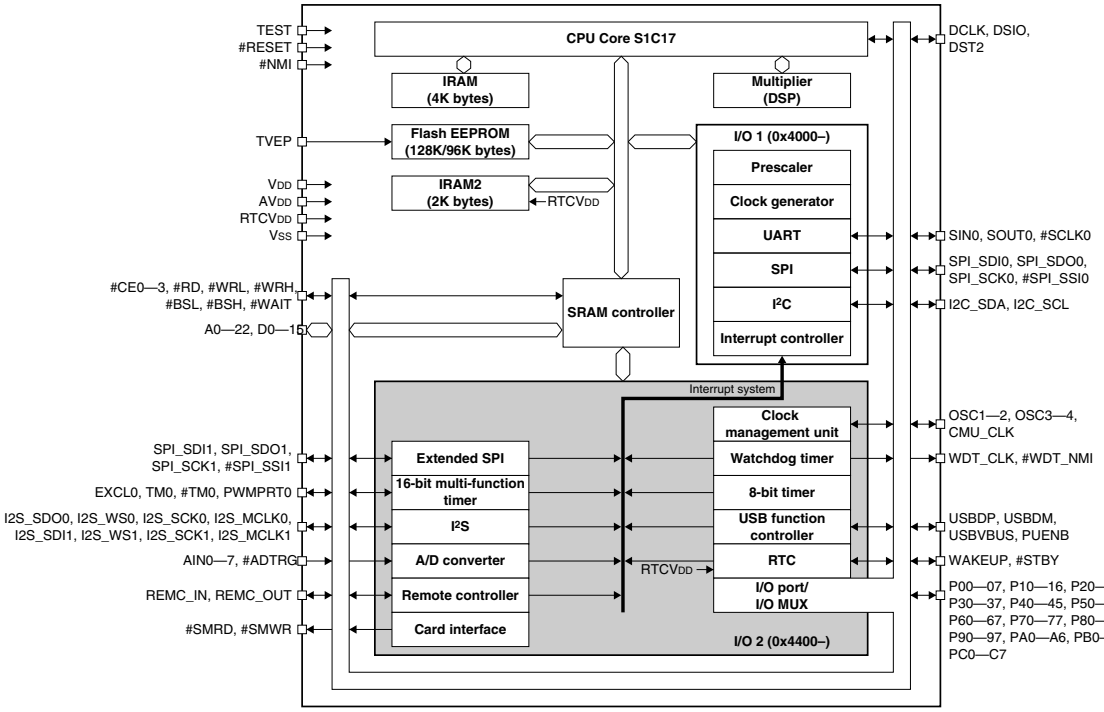


Figure I.2.1 S1C17501 Block Diagram

I.3 Pin Description

I.3.1 Pin Arrangement

The S1C17501 comes in a TQFP15-128pin or a PFBGA7U-144 package.

TQFP15-128pin package

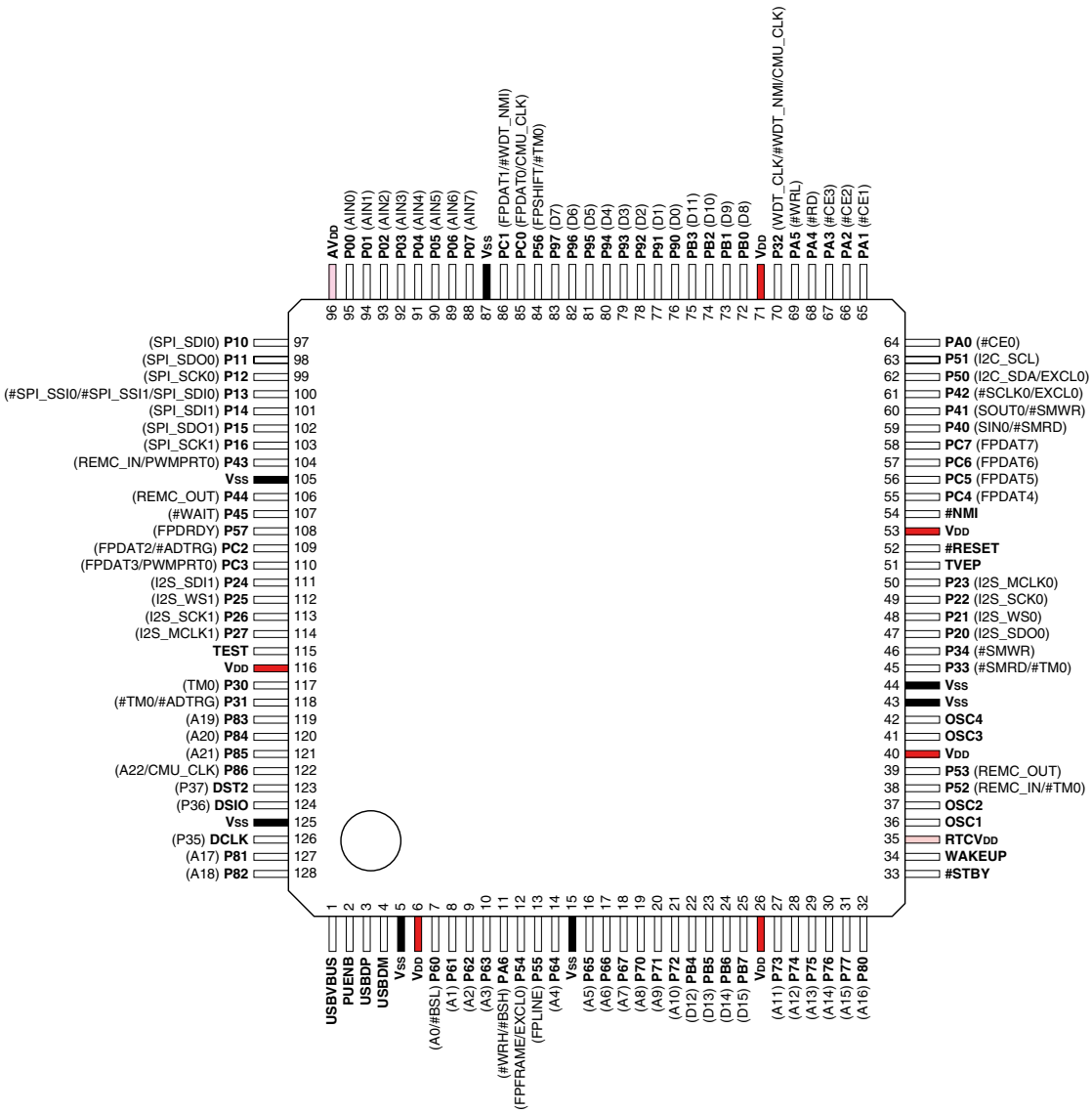


Figure I.3.1.1 Pin Arrangement (TQFP15-128pin)

TQFP14-100 package

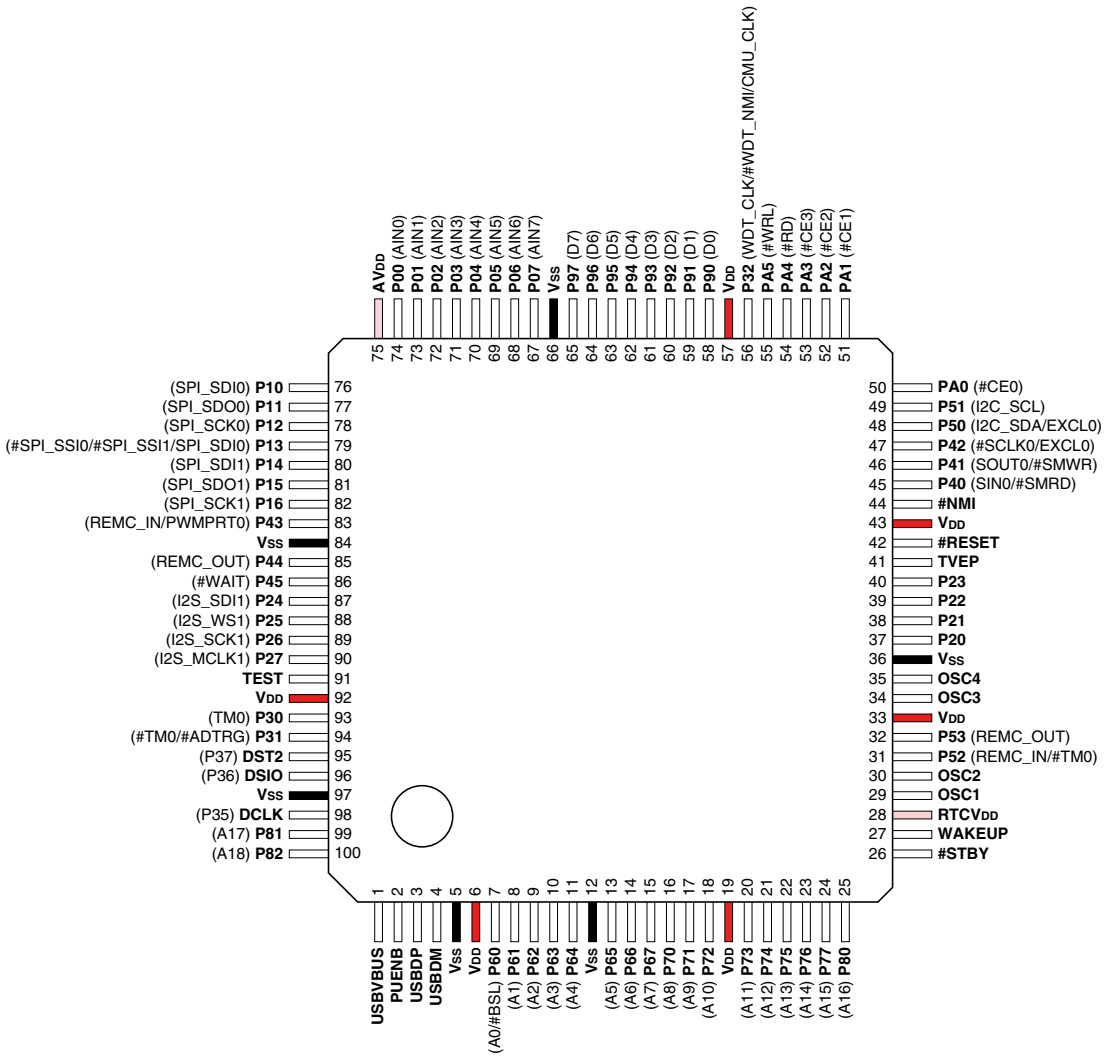


Figure I.3.1.2 Pin Arrangement (TQFP14-100pin)

I.3.2 Pin Functions

Tables I.3.2.1 to I.3.2.6 list the function of each pin on the S1C17501.

Table I.3.2.1 Power Supply Pin List

Pin name	Pin No.		I/O	Type	PU/PD	Description
	TQFP 128pin	TQFP 100pin				
V_{DD}	6, 26, 40, 53, 71, 116	6,19,33,43,57,92	–	3.3 V	–	Core and I/O power supply (+) (3.3 V)
V_{SS}	5, 15, 43, 44, 87, 105, 125	5,12,36,66,84,97	–	GND	–	GND
$RTCV_{DD}$	35	28	–	3.3 V	–	RTC power supply (+) (3.3 V) ($RTCV_{DD} = V_{DD}$)
AV_{DD}	96	75	–	3.3 V	–	Analog power supply (3.3 V) ($AV_{DD} = V_{DD}$)

Table I.3.2.2 Clock Pin List

Pin name	Pin No.		I/O	Type	PU/PD	Description
	TQFP 128pin	TQFP 100pin				
OSC3	41	34	I	Analog	–	High speed (OSC3) oscillation input (crystal/ceramic oscillator or external clock input)
OSC4	42	35	O	Analog	–	High speed (OSC3) oscillation output
OSC1	36	29	I	Analog	–	RTC (OSC1) oscillation input (crystal/ceramic oscillator or external clock input)
OSC2	37	30	O	Analog	–	RTC (OSC1) oscillation output

Table I.3.2.3 External Bus Pin List

Pin name	Pin No.		I/O	Type	PU/PD	Description	
	TQFP 128pin	TQFP 100pin					
P90 D0	76	58	I/o	LVTTL	Bus hold latch	P90: D0:	I/O port (default) Data bus D0
P91 D1	77	59	I/o	LVTTL	Bus hold latch	P91: D1:	I/O port (default) Data bus D1
P92 D2	78	60	I/o	LVTTL	Bus hold latch	P92: D2:	I/O port (default) Data bus D2
P93 D3	79	61	I/o	LVTTL	Bus hold latch	P93: D3:	I/O port (default) Data bus D3
P94 D4	80	62	I/o	LVTTL	Bus hold latch	P94: D4:	I/O port (default) Data bus D4
P95 D5	81	63	I/o	LVTTL	Bus hold latch	P95: D5:	I/O port (default) Data bus D5
P96 D6	82	64	I/o	LVTTL	Bus hold latch	P96: D6:	I/O port (default) Data bus D6
P97 D7	83	65	I/o	LVTTL	Bus hold latch	P97: D7:	I/O port (default) Data bus D7
PB0 D8	72		I/o	LVTTL	Bus hold latch	PB0: D8:	I/O port (default) Data bus D8
PB1 D9	73		I/o	LVTTL	Bus hold latch	PB1: D9:	I/O port (default) Data bus D9
PB2 D10	74		I/o	LVTTL	Bus hold latch	PB2: D10:	I/O port (default) Data bus D10
PB3 D11	75		I/o	LVTTL	Bus hold latch	PB3: D11:	I/O port (default) Data bus D11
PB4 D12	22		I/o	LVTTL	Bus hold latch	PB4: D12:	I/O port (default) Data bus D12
PB5 D13	23		I/o	LVTTL	Bus hold latch	PB5: D13:	I/O port (default) Data bus D13
PB6 D14	24		I/o	LVTTL	Bus hold latch	PB6: D14:	I/O port (default) Data bus D14
PB7 D15	25		I/o	LVTTL	Bus hold latch	PB7: D15:	I/O port (default) Data bus D15
P60 A0/#BSL	7	7	I/o	LVTTL Schmitt	–	P60: A0/#BSL:	I/O port (default) Address bus A0 / Bus strobe (low byte) signal output
P61 A1	8	8	I/o	LVTTL Schmitt	–	P61: A1:	I/O port (default) Address bus A1

Pin name	Pin No. TQFP 128pin	Pin No. TQFP 100pin	I/O	Type	PU/PD	Description	
P62	9	9	I/o	LVTTL	–	P62:	I/O port (default)
A2				Schmitt		A2:	Address bus A2
P63	10	10	I/o	LVTTL	–	P63:	I/O port (default)
A3				Schmitt		A3:	Address bus A3
P64	14	11	I/o	LVTTL	–	P64:	I/O port (default)
A4				Schmitt		A4:	Address bus A4
P65	16	13	I/o	LVTTL	–	P65:	I/O port (default)
A5				Schmitt		A5:	Address bus A5
P66	17	14	I/o	LVTTL	–	P66:	I/O port (default)
A6				Schmitt		A6:	Address bus A6
P67	18	15	I/o	LVTTL	–	P67:	I/O port (default)
A7				Schmitt		A7:	Address bus A7
P70	19	16	I/o	LVTTL	–	P70:	I/O port (default)
A8				Schmitt		A8:	Address bus A8
P71	20	17	I/o	LVTTL	–	P71:	I/O port (default)
A9				Schmitt		A9:	Address bus A9
P72	21	18	I/o	LVTTL	–	P72:	I/O port (default)
A10				Schmitt		A10:	Address bus A10
P73	27	20	I/o	LVTTL	–	P73:	I/O port (default)
A11				Schmitt		A11:	Address bus A11
P74	28	21	I/o	LVTTL	–	P74:	I/O port (default)
A12				Schmitt		A12:	Address bus A12
P75	29	22	I/o	LVTTL	–	P75:	I/O port (default)
A13				Schmitt		A13:	Address bus A13
P76	30	23	I/o	LVTTL	–	P76:	I/O port (default)
A14				Schmitt		A14:	Address bus A14
P77	31	24	I/o	LVTTL	–	P77:	I/O port (default)
A15				Schmitt		A15:	Address bus A15
P80	32	25	I/o	LVTTL	–	P80:	I/O port (default)
A16				Schmitt		A16:	Address bus A16
P81	127	99	I/o	LVTTL	–	P81:	I/O port (default)
A17				Schmitt		A17:	Address bus A17
P82	128	100	I/o	LVTTL	–	P82:	I/O port (default)
A18				Schmitt		A18:	Address bus A18
P83	119		I/o	LVTTL	–	P83:	I/O port (default)
A19				Schmitt		A19:	Address bus A19
P84	120		I/o	LVTTL	–	P84:	I/O port (default)
A20				Schmitt		A20:	Address bus A20
P85	121		I/o	LVTTL	–	P85:	I/O port (default)
A21				Schmitt		A21:	Address bus A21
P86	122		I/o	LVTTL	–	P86:	I/O port (default)
A22				Schmitt		A22:	Address bus A22
CMU_CLK						CMU_CLK:	CMU clock output
PA0	64	50	I/o	LVTTL	100k PU	PA0:	I/O port (default)
#CE0				Schmitt		#CE0:	#CE0 area chip enable signal output
PA1	65	51	I/o	LVTTL	100k PU	PA1:	I/O port (default)
#CE1				Schmitt		#CE1:	#CE1 area chip enable signal output
PA2	66	52	I/o	LVTTL	100k PU	PA2:	I/O port (default)
#CE2				Schmitt		#CE2:	#CE2 area chip enable signal output
PA3	67	53	I/o	LVTTL	100k PU	PA3:	I/O port (default)
#CE3				Schmitt		#CE3:	#CE3 area chip enable signal output
PA4	68	54	I/o	LVTTL	100k PU	PA4:	I/O port (default)
#RD				Schmitt		#RD:	Read signal output
PA5	69	55	I/o	LVTTL	100k PU	PA5:	I/O port (default)
#WRL				Schmitt		#WRL:	Write (low byte) signal output
PA6	11		I/o	LVTTL	100k PU	PA6:	I/O port (default)
#WRH/#BSH				Schmitt		#WRH/#BSH:	Write (high byte) signal / Bus strobe (high byte) signal output

Table I.3.2.4 Input/Output Port and Peripheral Circuit Pin List

Pin name	Pin No. TQFP 128pin	Pin No. TQFP 100pin	I/O	Type	PU/PD	Description
P00 AIN0	95	74	**I**	CMOS/ LVTTL	–	P00: Input port (default) AIN0: A/D converter CH.0 input
P01 AIN1	94	73	**I**	CMOS/ LVTTL	–	P01: Input port (default) AIN1: A/D converter CH.1 input
P02 AIN2	93	72	**I**	CMOS/ LVTTL	–	P02: Input port (default) AIN2: A/D converter CH.2 input
P03 AIN3	92	71	**I**	CMOS/ LVTTL	–	P03: Input port (default) AIN3: A/D converter CH.3 input
P04 AIN4	91	70	**I**	CMOS/ LVTTL	–	P04: Input port (default) AIN4: A/D converter CH.4 input
P05 AIN5	90	69	**I**	CMOS/ LVTTL	–	P05: Input port (default) AIN5: A/D converter CH.5 input
P06 AIN6	89	68	**I**	CMOS/ LVTTL	–	P06: Input port (default) AIN6: A/D converter CH.6 input
P07 AIN7	88	67	**I**	CMOS/ LVTTL	–	P07: Input port (default) AIN7: A/D converter CH.7 input
P10 SPI_SDI0	97	76	**I**/o	LVTTL Schmitt (*1)	–	P10: I/O port (default) SPI_SDI0: SPI CH.0 data input
P11 SPI_SDO0	98	77	**I**/o	LVTTL Schmitt (*1)	–	P11: I/O port (default) SPI_SDO0: SPI CH.0 data output
P12 SPI_SCK0	99	78	**I**/o	LVTTL Schmitt (*1)	–	P12: I/O port (default) SPI_SCK0: SPI CH.0 clock input/output
P13 #SPI_SSI0 #SPI_SSI1 SPI_SDI0	100	79	**I**/o	LVTTL Schmitt (*1)	–	P13: I/O port (default) SPI_SSL0: SPI CH.0 slave select signal output SPI_SSL1: SPI CH.1 slave select signal output SPI_SDI0: SPI CH.0 data input
P14 SPI_SDI1	101	80	**I**/o	LVTTL Schmitt (*1)	–	P14: I/O port (default) SPI_SDI1: SPI CH.1 data input
P15 SPI_SDO1	102	81	**I**/o	LVTTL Schmitt (*1)	–	P15: I/O port (default) SPI_SDO1: SPI CH.1 data output
P16 SPI_SCK1	103	82	**I**/o	LVTTL Schmitt (*1)	–	P16: I/O port (default) SPI_SCK1: SPI CH.1 clock input/output
P20 I2S_SDO0	47	37	**I**/o	LVTTL Schmitt (*1)	–	P20: I/O port (default) I2S_SDO0: I^2S CH.0 serial data output
P21 I2S_WS0	48	38	**I**/o	LVTTL Schmitt (*1)	–	P21: I/O port (default) I2S_WS0: I^2S CH.0 word select signal output
P22 I2S_SCK0	49	39	**I**/o	LVTTL Schmitt (*1)	–	P22: I/O port (default) I2S_SCK0: I^2S CH.0 serial bit clock output
P23 I2S_MCLK0	50	40	**I**/o	LVTTL Schmitt (*1)	–	P23: I/O port (default) I2S_MCLK0: I^2S CH.0 master clock input/output
P24 I2S_SDI1	111	87	**I**/o	LVTTL Schmitt (*1)	–	P24: I/O port (default) I2S_SDI1: I^2S CH.1 serial data input
P25 I2S_WS1	112	88	**I**/o	LVTTL Schmitt (*1)	–	P25: I/O port (default) I2S_WS1: I^2S CH.1 word select signal input
P26 I2S_SCK1	113	89	**I**/o	LVTTL Schmitt (*1)	–	P26: I/O port (default) I2S_SCK1: I^2S CH.1 serial bit clock input
P27 I2S_MCLK1	114	90	**I**/o	LVTTL Schmitt (*1)	–	P27: I/O port (default) I2S_MCLK1: I^2S CH.1 master clock input
P30 TM0	117	93	**I**/o	LVTTL Schmitt (*1)	–	P30: I/O port (default) TM0: 16-bit multi-function timer output
P31 #TM0 #ADTRG	118	94	**I**/o	LVTTL Schmitt (*1)	–	P31: I/O port (default) #TM0: 16-bit multi-function timer inverted output #ADTRG: A/D converter trigger input
P32 WDT_CLK #WDT_NMI CMU_CLK	70	56	**I**/o	LVTTL Schmitt (*1)	–	P32: I/O port (default) WDT_CLK: Watchdog timer clock output #WDT_NMI: Watchdog timer NMI signal output CMU_CLK: CMU clock output
P33 #SMRD #TM0	45		**I**/o	LVTTL Schmitt (*1)	–	P33: I/O port (default) #SMRD: Card I/F read signal output #TM0: 16-bit multi-function timer inverted output
P34 #SMWR	46		**I**/o	LVTTL Schmitt (*1)	–	P34: I/O port (default) #SMWR: Card I/F write signal output

Pin name	Pin No. TQFP 128pin	Pin No. TQFP 100pin	I/O	Type	PU/PD	Description	
P40 SIN0 #SMRD	59	45	I/o	LVTTL Schmitt (*1)	–	P40: SIN0: #SMRD:	I/O port (default) UART with IrDA CH.0 data input Card I/F read signal output
P41 SOUT0 #SMWR	60	46	I/o	LVTTL Schmitt (*1)	–	P41: SOUT0: #SMWR:	I/O port (default) UART with IrDA CH.0 data output Card I/F write signal output
P42 #SCLK0 EXCL0	61	47	I/o	LVTTL Schmitt (*1)	–	P42: #SCLK0: EXCL0:	I/O port (default) UART with IrDA CH.0 clock input/output 16-bit multi-function timer event counter input
P43 REMC_IN PWMPRT0	104	83	I/o	LVTTL Schmitt (*1)	–	P43: REMC_IN: PWMPRT0:	I/O port (default) Remote controller receive signal input 16-bit multi-function timer port protection signal input
P44 REMC_OUT	106	85	I/o	LVTTL Schmitt (*1)	–	P44: REMC_OUT:	I/O port (default) Remote controller transmit signal output
P45 #WAIT	107	86	I/o	LVTTL	100k PU	P45: #WAIT:	I/O port (default) Wait cycle request input
P50 I2C_SDA EXCL0	62	48	I/o	LVTTL Schmitt (*1)	–	P50: I2C_SDA: EXCL0:	I/O port (default) I^2C data signal 16-bit multi-function timer event counter input
P51 I2C_SCL	63	49	I/o	LVTTL Schmitt (*1)	–	P51: I2C_SCL:	I/O port (default) I^2C clock output
P52 REMC_IN #TM0	38	31	I/o	LVTTL Schmitt (*1)	–	P52: REMC_IN: #TM0:	I/O port (default) Remote controller receive signal input 16-bit multi-function timer inverted output
P53 REMC_OUT	39	32	I/o	LVTTL Schmitt (*1)	–	P53: REMC_OUT:	I/O port (default) Remote controller transmit signal output
P54 EXCL0	12		I/o	LVTTL Schmitt (*1)	–	P54: EXCL0:	I/O port (default) 16-bit multi-function timer event counter input
P55	13		I/o	LVTTL Schmitt (*1)	–	P55:	I/O port (default)
P56 #TM0	84		I/o	LVTTL Schmitt (*1)	–	P56: #TM0:	I/O port (default) 16-bit multi-function timer inverted output
P57	108		I/o	LVTTL Schmitt (*1)	–	P57:	I/O port (default)
PC0 CMU_CLK	85		I/o	LVTTL Schmitt (*1)	–	PC0: CMU_CLK:	I/O port (default) CMU clock output
PC1 #WDT_NMI	86		I/o	LVTTL Schmitt (*1)	–	PC1: #WDT_NMI:	I/O port (default) Watchdog timer NMI signal output
PC2 #ADTRG	109		I/o	LVTTL Schmitt (*1)	–	PC2: #ADTRG:	I/O port (default) A/D converter trigger input
PC3 PWMPRT0	110		I/o	LVTTL Schmitt (*1)	–	PC3: PWMPRT0:	I/O port (default) 16-bit multi-function timer port protection signal input
PC4	55		I/o	LVTTL Schmitt (*1)	–	PC4:	I/O port (default)
PC5	56		I/o	LVTTL Schmitt (*1)	–	PC5:	I/O port (default)
PC6	57		I/o	LVTTL Schmitt (*1)	–	PC6:	I/O port (default)
PC7	58		I/o	LVTTL Schmitt (*1)	–	PC7:	I/O port (default)

*1: These 41 GPIO pins allow input of a voltage within the range from -0.3 to 5.8 V.

Table I.3.2.5 USB Interface Pin List

Pin name	Pin No. TQFP 128pin	Pin No. TQFP 100pin	I/O	Type	PU/PD	Description
USBDP	3	3	**I**/o	USB	–	USB D+ pin (∗1)
USBDM	4	4	**I**/o	USB	–	USB D- pin (∗1)
USBVBUS	1	1	**I**	USB	–	USB VBUS pin. Allows input of 5 V. (∗2)
PUENB	2	2	**O**	USB	–	USB DP pull-up enable output (P channel open drain output). (∗1)

∗1: These pins must be left open when the USB function controller is not used.
∗2: This pin must be connected to GND (V_{SS}) when the USB function controller is not used.

Table I.3.2.6 Other Pin List

Pin name	Pin No. TQFP 128pin	Pin No. TQFP 100pin	I/O	Type	PU/PD	Description
#RESET	52	42	**I**	LVTTL Schmitt	100k PU	Reset input (with noise filter)
#NMI	54	44	**I**	LVTTL Schmitt	100k PU	NMI request input (with noise filter)
DCLK P35	126	98	i/**O**	LVTTL Schmitt	–	DCLK: DCLK (Debug SIO Clock) signal output (default) P35: I/O port
DSIO P36	124	96	**I**/o	LVTTL Schmitt	100k PU	DSIO: DSIO (Debug SIO) pin (with noise filter) (default) P36: I/O port
DST2 P37	123	95	i/**O**	LVTTL Schmitt	–	DST2: DST2 (Debug Status) signal output (default) P37: I/O port
TEST	115	91	**I**	Special	50k PD	Test input. Connect to V_{SS} in user mode.
WAKEUP	34	27	**O**	–	–	C17 wakeup signal output from RTC
#STBY	33	26	**I**	LVTTL Schmitt	–	C17 standby input (except for RTC)
TVEP	51	41	**I**	Special	50k PD	FLASHC test input. Connect to V_{DD} in user mode.

Notes:
- The # prefixed to pin names indicates that input/output signals of the pin are active low.
- The pin names listed in boldface denote the default pin (signal) name.
- The I/O listed in boldface and uppercase denote the default input/output direction.
- “PU” means “Pull-up” and “PD” means “Pull-down.”

I.3.3 Switching Over the Multiplexed Pin Functions

I.3.3.1 Pin Function Select Bits

Each pin is assigned one to four functions, as listed in Table I.3.3.1.1.
When the chip is powered on or reset, each pin defaults to function 0. If any pin must be used for other than this default function, select the desired function by writing data to the corresponding pin function select bits.

Table I.3.3.1.1 List of Pin Function Select Bits

Pin function 0	Pin function 1	Pin function 2	Pin function 3	Function select bit
OSC3				
OSC4				
OSC1				
OSC2				
TEST				
#RESET				
#NMI				
DCLK	P35			CFP35[1:0] (D[3:2]/0x4427)
DSIO	P36			CFP36[1:0] (D[5:4]/0x4427)
DST2	P37			CFP37[1:0] (D[7:6]/0x4427)
P90	D0			CFP90[1:0] (D[1:0]/0x4432)
P91	D1			CFP91[1:0] (D[3:2]/0x4432)
P92	D2			CFP92[1:0] (D[5:4]/0x4432)
P93	D3			CFP93[1:0] (D[7:6]/0x4432)
P94	D4			CFP94[1:0] (D[1:0]/0x4433)
P95	D5			CFP95[1:0] (D[3:2]/0x4433)
P96	D6			CFP96[1:0] (D[5:4]/0x4433)
P97	D7			CFP97[1:0] (D[7:6]/0x4433)
PB0	D8			CFPB0[1:0] (D[1:0]/0x4436)
PB1	D9			CFPB1[1:0] (D[3:2]/0x4436)
PB2	D10			CFPB2[1:0] (D[5:4]/0x4436)
PB3	D11			CFPB3[1:0] (D[7:6]/0x4436)
PB4	D12			CFPB4[1:0] (D[1:0]/0x4437)
PB5	D13			CFPB5[1:0] (D[3:2]/0x4437)
PB6	D14			CFPB6[1:0] (D[5:4]/0x4437)
PB7	D15			CFPB7[1:0] (D[7:6]/0x4437)
P60	A0 / #BSL			CFP60[1:0] (D[1:0]/0x442c)
P61	A1			CFP61[1:0] (D[3:2]/0x442c)
P62	A2			CFP62[1:0] (D[5:4]/0x442c)
P63	A3			CFP63[1:0] (D[7:6]/0x442c)
P64	A4			CFP64[1:0] (D[1:0]/0x442d)
P65	A5			CFP65[1:0] (D[3:2]/0x442d)
P66	A6			CFP66[1:0] (D[5:4]/0x442d)
P67	A7			CFP67[1:0] (D[7:6]/0x442d)
P70	A8			CFP70[1:0] (D[1:0]/0x442e)
P71	A9			CFP71[1:0] (D[3:2]/0x442e)
P72	A10			CFP72[1:0] (D[5:4]/0x442e)
P73	A11			CFP73[1:0] (D[7:6]/0x442e)
P74	A12			CFP74[1:0] (D[1:0]/0x442f)
P75	A13			CFP75[1:0] (D[3:2]/0x442f)
P76	A14			CFP76[1:0] (D[5:4]/0x442f)
P77	A15			CFP77[1:0] (D[7:6]/0x442f)
P80	A16			CFP80[1:0] (D[1:0]/0x4430)
P81	A17			CFP81[1:0] (D[3:2]/0x4430)
P82	A18			CFP82[1:0] (D[5:4]/0x4430)
P83	A19			CFP83[1:0] (D[7:6]/0x4430)
P84	A20			CFP84[1:0] (D[1:0]/0x4431)
P85	A21			CFP85[1:0] (D[3:2]/0x4431)
P86	A22	CMU_CLK		CFP86[1:0] (D[5:4]/0x4431)

Pin function 0	Pin function 1	Pin function 2	Pin function 3	Function select bit
PA0	#CE0			CFPA0[1:0] (D[1:0]/0x4434)
PA1	#CE1			CFPA1[1:0] (D[3:2]/0x4434)
PA2	#CE2			CFPA2[1:0] (D[5:4]/0x4434)
PA3	#CE3			CFPA3[1:0] (D[7:6]/0x4434)
PA4	#RD			CFPA4[1:0] (D[1:0]/0x4435)
PA5	#WRL			CFPA5[1:0] (D[3:2]/0x4435)
PA6	#WRH / #BSH			CFPA6[1:0] (D[5:4]/0x4435)
WAKEUP				
#STBY				
P00	AIN0			CFP00[1:0] (D[1:0]/0x4420)
P01	AIN1			CFP01[1:0] (D[3:2]/0x4420)
P02	AIN2			CFP02[1:0] (D[5:4]/0x4420)
P03	AIN3			CFP03[1:0] (D[7:6]/0x4420)
P04	AIN4			CFP04[1:0] (D[1:0]/0x4421)
P05	AIN5			CFP05[1:0] (D[3:2]/0x4421)
P06	AIN6			CFP06[1:0] (D[5:4]/0x4421)
P07	AIN7			CFP07[1:0] (D[7:6]/0x4421)
P10	SPI_SDI0			CFP10[1:0] (D[1:0]/0x4422)
P11	SPI_SDO0			CFP11[1:0] (D[3:2]/0x4422)
P12	SPI_SCK0			CFP12[1:0] (D[5:4]/0x4422)
P13	#SPI_SSI0	#SPI_SSI1	SPI_SDI0	CFP13[1:0] (D[7:6]/0x4422)
P14	SPI_SDI1			CFP14[1:0] (D[1:0]/0x4423)
P15	SPI_SDO1			CFP15[1:0] (D[3:2]/0x4423)
P16	SPI_SCK1			CFP16[1:0] (D[5:4]/0x4423)
P20	I2S_SDO0			CFP20[1:0] (D[1:0]/0x4424)
P21	I2S_WS0			CFP21[1:0] (D[3:2]/0x4424)
P22	I2S_SCK0			CFP22[1:0] (D[5:4]/0x4424)
P23	I2S_MCLK0			CFP23[1:0] (D[7:6]/0x4424)
P24	I2S_SDI1			CFP24[1:0] (D[1:0]/0x4425)
P25	I2S_WS1			CFP25[1:0] (D[3:2]/0x4425)
P26	I2S_SCK1			CFP26[1:0] (D[5:4]/0x4425)
P27	I2S_MCLK1			CFP27[1:0] (D[7:6]/0x4425)
P30	TM0			CFP30[1:0] (D[1:0]/0x4426)
P31		#TM0	#ADTRG	CFP31[1:0] (D[3:2]/0x4426)
P32	WDT_CLK	#WDT_NMI	CMU_CLK	CFP32[1:0] (D[5:4]/0x4426)
P33	#SMRD	#TM0		CFP33[1:0] (D[7:6]/0x4426)
P34	#SMWR			CFP34[1:0] (D[1:0]/0x4427)
P40	SIN0	#SMRD		CFP40[1:0] (D[1:0]/0x4428)
P41	SOUT0	#SMWR		CFP41[1:0] (D[3:2]/0x4428)
P42	#SCLK0	EXCL0		CFP42[1:0] (D[5:4]/0x4428)
P43		REMC_IN	PWMPRT0	CFP43[1:0] (D[7:6]/0x4428)
P44		REMC_OUT		CFP44[1:0] (D[1:0]/0x4429)
P45			#WAIT	CFP45[1:0] (D[3:2]/0x4429)
P50	I2C_SDA	EXCL0		CFP50[1:0] (D[1:0]/0x442a)
P51	I2C_SCL			CFP51[1:0] (D[3:2]/0x442a)
P52	REMC_IN	#TM0		CFP52[1:0] (D[5:4]/0x442a)
P53	REMC_OUT			CFP53[1:0] (D[7:6]/0x442a)
P54		EXCL0		CFP54[1:0] (D[1:0]/0x442b)
P55				CFP55[1:0] (D[3:2]/0x442b)
P56		#TM0		CFP56[1:0] (D[5:4]/0x442b)
P57				CFP57[1:0] (D[7:6]/0x442b)
PC0		CMU_CLK		CFPC0[1:0] (D[1:0]/0x4438)
PC1		#WDT_NMI		CFPC1[1:0] (D[3:2]/0x4438)
PC2		#ADTRG		CFPC2[1:0] (D[5:4]/0x4438)
PC3		PWMPRT0		CFPC3[1:0] (D[7:6]/0x4438)
PC4				CFPC4[1:0] (D[1:0]/0x4439)
PC5				CFPC5[1:0] (D[3:2]/0x4439)
PC6				CFPC6[1:0] (D[5:4]/0x4439)

Pin function 0	Pin function 1	Pin function 2	Pin function 3	Function select bit
PC7				CFPC7[1:0] (D[7:6]/0x4439)
USBDP				
USBDM				
USBVBUS				
PUENB				
TVEP				

∗ The set values 0 to 3 of the pin function select bits correspond to functions 0 to 3, respectively.

I.3.3.2 List of Port Function Select Registers

Table I.3.3.2.1 List of Port Function Select Registers

Address	Register name		Function
0x4420	P0_03_CFP	P00–P03 Port Function Select Register	Selects the P00–P03 port functions.
0x4421	P0_47_CFP	P04–P07 Port Function Select Register	Selects the P04–P07 port functions.
0x4422	P1_03_CFP	P10–P13 Port Function Select Register	Selects the P10–P13 port functions.
0x4423	P1_46_CFP	P14–P16 Port Function Select Register	Selects the P14–P16 port functions.
0x4424	P2_03_CFP	P20–P23 Port Function Select Register	Selects the P20–P23 port functions.
0x4425	P2_47_CFP	P24–P27 Port Function Select Register	Selects the P24–P27 port functions.
0x4426	P3_03_CFP	P30–P33 Port Function Select Register	Selects the P30–P33 port functions.
0x4427	P3_47_CFP	P34–P37 Port Function Select Register	Selects the P34–P37 port functions.
0x4428	P4_03_CFP	P40–P43 Port Function Select Register	Selects the P40–P43 port functions.
0x4429	P4_45_CFP	P44–P45 Port Function Select Register	Selects the P44–P45 port functions.
0x442a	P5_03_CFP	P50–P53 Port Function Select Register	Selects the P50–P53 port functions.
0x442b	P5_47_CFP	P54–P57 Port Function Select Register	Selects the P54–P57 port functions.
0x442c	P6_03_CFP	P60–P63 Port Function Select Register	Selects the P60–P63 port functions.
0x442d	P6_47_CFP	P64–P67 Port Function Select Register	Selects the P64–P67 port functions.
0x442e	P7_03_CFP	P70–P73 Port Function Select Register	Selects the P70–P73 port functions.
0x442f	P7_47_CFP	P74–P77 Port Function Select Register	Selects the P74–P77 port functions.
0x4430	P8_03_CFP	P80–P83 Port Function Select Register	Selects the P80–P83 port functions.
0x4431	P8_46_CFP	P84–P86 Port Function Select Register	Selects the P84–P86 port functions.
0x4432	P9_03_CFP	P90–P93 Port Function Select Register	Selects the P90–P93 port functions.
0x4433	P9_47_CFP	P94–P97 Port Function Select Register	Selects the P94–P97 port functions.
0x4434	PA_03_CFP	PA0–PA3 Port Function Select Register	Selects the PA0–PA3 port functions.
0x4435	PA_46_CFP	PA4–PA6 Port Function Select Register	Selects the PA4–PA6 port functions.
0x4436	PB_03_CFP	PB0–PB3 Port Function Select Register	Selects the PB0–PB3 port functions.
0x4437	PB_47_CFP	PB4–PB7 Port Function Select Register	Selects the PB4–PB7 port functions.
0x4438	PC_03_CFP	PC0–PC3 Port Function Select Register	Selects the PC0–PC3 port functions.
0x4439	PC_47_CFP	PC4–PC7 Port Function Select Register	Selects the PC4–PC7 port functions.

The following describes each port function select register. These are all 8-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x4420: P00–P03 Port Function Select Register (P0_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P00–P03 Port Function Select Register (P0_03_CFP)	0x4420 (8 bits)	D7–6	CFP03[1:0]	P03 port function select	CFP03[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN3			
					0x0	P03			
		D5–4	CFP02[1:0]	P02 port function select	CFP02[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN2			
					0x0	P02			
		D3–2	CFP01[1:0]	P01 port function select	CFP01[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN1			
					0x0	P01			
		D1–0	CFP00[1:0]	P00 port function select	CFP00[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN0			
					0x0	P00			

This register selects the functions of the P00 to P03 ports.

D[7:6] CFP03[1:0]: P03 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN3
00 (R/W): P03 input port (default)

D[5:4] CFP02[1:0]: P02 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN2
00 (R/W): P02 input port (default)

D[3:2] CFP01[1:0]: P01 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN1
00 (R/W): P01 input port (default)

D[1:0] CFP00[1:0]: P00 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN0
00 (R/W): P00 input port (default)

0x4421: P04–P07 Port Function Select Register (P0_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P04–P07 Port Function Select Register (P0_47_CFP)	0x4421 (8 bits)	D7–6	CFP07[1:0]	P07 port function select	CFP07[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN7			
					0x0	P07			
		D5–4	CFP06[1:0]	P06 port function select	CFP06[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN6			
					0x0	P06			
		D3–2	CFP05[1:0]	P05 port function select	CFP05[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN5			
					0x0	P05			
		D1–0	CFP04[1:0]	P04 port function select	CFP04[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	AIN4			
					0x0	P04			

This register selects the functions of the P04 to P07 ports.

D[7:6] CFP07[1:0]: P07 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN7
00 (R/W): P07 input port (default)

D[5:4] CFP06[1:0]: P06 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN6
00 (R/W): P06 input port (default)

D[3:2] CFP05[1:0]: P05 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN5
00 (R/W): P05 input port (default)

D[1:0] CFP04[1:0]: P04 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): AIN4
00 (R/W): P04 input port (default)

0x4422: P10–P13 Port Function Select Register (P1_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P10–P13 Port Function Select Register (P1_03_CFP)	0x4422 (8 bits)	D7–6	CFP13[1:0]	P13 port function select	CFP13[1:0]	Function	0x0	R/W	
					0x3	SPI_SDI0			
					0x2	#SPI_SSI1			
					0x1	#SPI_SSI0			
					0x0	P13			
		D5–4	CFP12[1:0]	P12 port function select	CFP12[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SCK0			
					0x0	P12			
		D3–2	CFP11[1:0]	P11 port function select	CFP11[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SDO0			
					0x0	P11			
		D1–0	CFP10[1:0]	P10 port function select	CFP10[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SDI0			
					0x0	P10			

This register selects the functions of the P10 to P13 ports.

D[7:6] CFP13[1:0]: P13 Port Function Select Bits

11 (R/W): SPI_SDI0
10 (R/W): #SPI_SSI1
01 (R/W): #SPI_SSI0
00 (R/W): P13 I/O port (default)

D[5:4] CFP12[1:0]: P12 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): SPI_SCK0
00 (R/W): P12 I/O port (default)

D[3:2] CFP11[1:0]: P11 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): SPI_SDO0
00 (R/W): P11 I/O port (default)

D[1:0] CFP10[1:0]: P10 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): SPI_SDI0
00 (R/W): P10 I/O port (default)

0x4423: P14–P16 Port Function Select Register (P1_46_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P14–P16 Port Function Select Register (P1_46_CFP)	0x4423 (8 bits)	D7–6	–	reserved	–		–	–	0 when being read.
		D5–4	CFP16[1:0]	P16 port function select	CFP16[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SCK1			
					0x0	P16			
		D3–2	CFP15[1:0]	P15 port function select	CFP15[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SDO1			
					0x0	P15			
		D1–0	CFP14[1:0]	P14 port function select	CFP14[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SDI1			
					0x0	P14			

This register selects the functions of the P14 to P16 ports.

D[7:6] Reserved

D[5:4] CFP16[1:0]: P16 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): SPI_SCK1
00 (R/W): P16 I/O port (default)

D[3:2] CFP15[1:0]: P15 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): SPI_SDO1
00 (R/W): P15 I/O port (default)

D[1:0] CFP14[1:0]: P14 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): SPI_SDI1
00 (R/W): P14 I/O port (default)

0x4424: P20–P23 Port Function Select Register (P2_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P20–P23 Port Function Select Register (P2_03_CFP)	0x4424 (8 bits)	D7–6	CFP23[1:0]	P23 port function select	CFP23[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_MCLK0			
					0x0	P23			
		D5–4	CFP22[1:0]	P22 port function select	CFP22[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SCK0			
					0x0	P22			
		D3–2	CFP21[1:0]	P21 port function select	CFP21[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_WS0			
					0x0	P21			
		D1–0	CFP20[1:0]	P20 port function select	CFP20[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SDO0			
					0x0	P20			

This register selects the functions of the P20 to P23 ports.

D[7:6] CFP23[1:0]: P23 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_MCLK0
00 (R/W): P23 I/O port (default)

D[5:4] CFP22[1:0]: P22 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_SCK0
00 (R/W): P22 I/O port (default)

D[3:2] CFP21[1:0]: P21 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_WS0
00 (R/W): P21 I/O port (default)

D[1:0] CFP20[1:0]: P20 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_SDO0
00 (R/W): P20 I/O port (default)

0x4425: P24–P27 Port Function Select Register (P2_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P24–P27 Port Function Select Register (P2_47_CFP)	0x4425 (8 bits)	D7–6	CFP27[1:0]	P27 port function select	CFP27[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_MCLK1			
					0x0	P27			
		D5–4	CFP26[1:0]	P26 port function select	CFP26[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SCK1			
					0x0	P26			
		D3–2	CFP25[1:0]	P25 port function select	CFP25[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_WS1			
					0x0	P25			
		D1–0	CFP24[1:0]	P24 port function select	CFP24[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SDI1			
					0x0	P24			

This register selects the functions of the P24 to P27 ports.

D[7:6] CFP27[1:0]: P27 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_MCLK1
00 (R/W): P27 I/O port (default)

D[5:4] CFP26[1:0]: P26 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_SCK1
00 (R/W): P26 I/O port (default)

D[3:2] CFP25[1:0]: P25 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_WS1
00 (R/W): P25 I/O port (default)

D[1:0] CFP24[1:0]: P24 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2S_SDI1
00 (R/W): P24 I/O port (default)

0x4426: P30–P33 Port Function Select Register (P3_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P30–P33 Port Function Select Register (P3_03_CFP)	0x4426 (8 bits)	D7–6	CFP33[1:0]	P33 port function select	CFP33[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#TM0			
					0x1	#SMRD			
					0x0	P33			
		D5–4	CFP32[1:0]	P32 port function select	CFP32[1:0]	Function	0x0	R/W	
					0x3	CMU_CLK			
					0x2	#WDT_NMI			
					0x1	WDT_CLK			
					0x0	P32			
		D3–2	CFP31[1:0]	P31 port function select	CFP31[1:0]	Function	0x0	R/W	
					0x3	#ADTRG			
					0x2	#TM0			
					0x1	reserved			
					0x0	P31			
		D1–0	CFP30[1:0]	P30 port function select	CFP30[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	TM0			
					0x0	P30			

This register selects the functions of the P30 to P33 ports.

D[7:6] CFP33[1:0]: P33 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #TM0
01 (R/W): #SMRD
00 (R/W): P33 I/O port (default)

D[5:4] CFP32[1:0]: P32 Port Function Select Bits

11 (R/W): CMU_CLK
10 (R/W): #WDT_NMI
01 (R/W): WDT_CLK
00 (R/W): P32 I/O port (default)

D[3:2] CFP31[1:0]: P31 Port Function Select Bits

11 (R/W): #ADTRG
10 (R/W): #TM0
01 (R/W): Reserved
00 (R/W): P31 I/O port (default)

D[1:0] CFP30[1:0]: P30 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): TM0
00 (R/W): P30 I/O port (default)

0x4427: P34–P37 Port Function Select Register (P3_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P34–P37 Port Function Select Register (P3_47_CFP)	0x4427 (8 bits)	D7–6	CFP37[1:0]	P37 port function select	CFP37[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	P37			
					0x0	DST2			
		D5–4	CFP36[1:0]	P36 port function select	CFP36[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	P36			
					0x0	DSIO			
		D3–2	CFP35[1:0]	P35 port function select	CFP35[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	P35			
					0x0	DCLK			
		D1–0	CFP34[1:0]	P34 port function select	CFP34[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#SMWR			
					0x0	P34			

This register selects the functions of the P34 to P37 ports.

D[7:6] CFP37[1:0]: P37 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): P37 I/O port
00 (R/W): DST2 (default)

D[5:4] CFP36[1:0]: P36 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): P36 I/O port
00 (R/W): DSIO (default)

D[3:2] CFP35[1:0]: P35 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): P35 I/O port
00 (R/W): DCLK (default)

D[1:0] CFP34[1:0]: P34 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #SMWR
00 (R/W): P34 I/O port (default)

0x4428: P40–P43 Port Function Select Register (P4_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P40–P43 Port Function Select Register (P4_03_CFP)	0x4428 (8 bits)	D7–6	CFP43[1:0]	P43 port function select	CFP43[1:0]	Function	0x0	R/W	
					0x3	PWMPRT0			
					0x2	REMC_IN			
					0x1	reserved			
					0x0	P43			
		D5–4	CFP42[1:0]	P42 port function select	CFP42[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	#SCLK0			
					0x0	P42			
		D3–2	CFP41[1:0]	P41 port function select	CFP41[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#SMWR			
					0x1	SOUT0			
					0x0	P41			
		D1–0	CFP40[1:0]	P40 port function select	CFP40[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#SMRD			
					0x1	SIN0			
					0x0	P40			

This register selects the functions of the P40 to P43 ports.

D[7:6] CFP43[1:0]: P43 Port Function Select Bits

11 (R/W): PWMPRT0
10 (R/W): REMC_IN
01 (R/W): Reserved
00 (R/W): P43 I/O port (default)

D[5:4] CFP42[1:0]: P42 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): EXCL0
01 (R/W): #SCLK0
00 (R/W): P42 I/O port (default)

D[3:2] CFP41[1:0]: P41 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #SMWR
01 (R/W): SOUT0
00 (R/W): P41 I/O port (default)

D[1:0] CFP40[1:0]: P40 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #SMRD
01 (R/W): SIN0
00 (R/W): P40 I/O port (default)

0x4429: P44–P45 Port Function Select Register (P4_45_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P44–P45 Port Function Select Register (P4_45_CFP)	**0x4429** (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3–2	**CFP45[1:0]**	P45 port function select	CFP45[1:0]	Function	0x0	R/W	
					0x3	#WAIT			
					0x2–0x1	reserved			
					0x0	P45			
		D1–0	**CFP44[1:0]**	P44 port function select	CFP44[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	REMC_OUT			
					0x1	reserved			
					0x0	P44			

This register selects the functions of the P44 to P45 ports.

D[7:4] Reserved

D[3:2] CFP45[1:0]: P45 Port Function Select Bits

11 (R/W): #WAIT
10 (R/W): Reserved
01 (R/W): Reserved
00 (R/W): P45 I/O port (default)

D[1:0] CFP44[1:0]: P44 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): REMC_OUT
01 (R/W): Reserved
00 (R/W): P44 I/O port (default)

0x442a: P50–P53 Port Function Select Register (P5_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P50–P53 Port Function Select Register (P5_03_CFP)	**0x442a** (8 bits)	D7–6	**CFP53[1:0]**	P53 port function select	CFP53[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	REMC_OUT			
					0x0	P53			
		D5–4	**CFP52[1:0]**	P52 port function select	CFP52[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#TM0			
					0x1	REMC_IN			
					0x0	P52			
		D3–2	**CFP51[1:0]**	P51 port function select	CFP51[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2C_SCL			
					0x0	P51			
		D1–0	**CFP50[1:0]**	P50 port function select	CFP50[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	I2C_SDA			
					0x0	P50			

This register selects the functions of the P50 to P53 ports.

D[7:6] CFP53[1:0]: P53 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): REMC_OUT
00 (R/W): P53 I/O port (default)

D[5:4] CFP52[1:0]: P52 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #TM0
01 (R/W): REMC_IN
00 (R/W): P52 I/O port (default)

D[3:2] CFP51[1:0]: P51 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): I2C_SCL
00 (R/W): P51 I/O port (default)

D[1:0] CFP50[1:0]: P50 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): EXCL0
01 (R/W): I2C_SDA
00 (R/W): P50 I/O port (default)

0x442b: P54–P57 Port Function Select Register (P5_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P54–P57 Port Function Select Register (P5_47_CFP)	**0x442b** (8 bits)	D7–6	**CFP57[1:0]**	P57 port function select	CFP57[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPDRDY			
					0x0	P57			
		D5–4	**CFP56[1:0]**	P56 port function select	CFP56[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#TM0			
					0x1	FPSHIFT			
					0x0	P56			
		D3–2	**CFP55[1:0]**	P55 port function select	CFP55[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPLINE			
					0x0	P55			
		D1–0	**CFP54[1:0]**	P54 port function select	CFP54[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	FPFRAME			
					0x0	P54			

This register selects the functions of the P54 to P57 ports.

D[7:6] CFP57[1:0]: P57 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): FPDRDY
00 (R/W): P57 I/O port (default)

D[5:4] CFP56[1:0]: P56 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #TM0
01 (R/W): FPSHIFT
00 (R/W): P56 I/O port (default)

D[3:2] CFP55[1:0]: P55 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): FPLINE
00 (R/W): P55 I/O port (default)

D[1:0] CFP54[1:0]: P54 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): EXCL0
01 (R/W): FPFRAME
00 (R/W): P54 I/O port (default)

0x442c: P60–P63 Port Function Select Register (P6_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P60–P63 Port Function Select Register (P6_03_CFP)	0x442c (8 bits)	D7–6	CFP63[1:0]	P63 port function select	CFP63[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A3			
					0x0	P63			
		D5–4	CFP62[1:0]	P62 port function select	CFP62[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A2			
					0x0	P62			
		D3–2	CFP61[1:0]	P61 port function select	CFP61[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A1			
					0x0	P61			
		D1–0	CFP60[1:0]	P60 port function select	CFP60[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A0/#BSL			
					0x0	P60			

This register selects the functions of the P60 to P63 ports.

D[7:6] CFP63[1:0]: P63 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A3
00 (R/W): P63 I/O port (default)

D[5:4] CFP62[1:0]: P62 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A2
00 (R/W): P62 I/O port (default)

D[3:2] CFP61[1:0]: P61 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A1
00 (R/W): P61 I/O port (default)

D[1:0] CFP60[1:0]: P60 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A0/#BSL
00 (R/W): P60 I/O port (default)

0x442d: P64–P67 Port Function Select Register (P6_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P64–P67 Port Function Select Register (P6_47_CFP)	**0x442d** (8 bits)	D7–6	**CFP67[1:0]**	P67 port function select	CFP67[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A7			
					0x0	P67			
		D5–4	**CFP66[1:0]**	P66 port function select	CFP66[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A6			
					0x0	P66			
		D3–2	**CFP65[1:0]**	P65 port function select	CFP65[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A5			
					0x0	P65			
		D1–0	**CFP64[1:0]**	P64 port function select	CFP64[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A4			
					0x0	P64			

This register selects the functions of the P64 to P67 ports.

D[7:6] CFP67[1:0]: P67 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A7
00 (R/W): P67 I/O port (default)

D[5:4] CFP66[1:0]: P66 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A6
00 (R/W): P66 I/O port (default)

D[3:2] CFP65[1:0]: P65 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A5
00 (R/W): P65 I/O port (default)

D[1:0] CFP64[1:0]: P64 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A4
00 (R/W): P64 I/O port (default)

0x442e: P70–P73 Port Function Select Register (P7_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P70–P73 Port Function Select Register (P7_03_CFP)	0x442e (8 bits)	D7–6	CFP73[1:0]	P73 port function select	CFP73[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A11			
					0x0	P73			
		D5–4	CFP72[1:0]	P72 port function select	CFP72[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A10			
					0x0	P72			
		D3–2	CFP71[1:0]	P71 port function select	CFP71[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A9			
					0x0	P71			
		D1–0	CFP70[1:0]	P70 port function select	CFP70[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A8			
					0x0	P70			

This register selects the functions of the P70 to P73 ports.

D[7:6] CFP73[1:0]: P73 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A11
00 (R/W): P73 I/O port (default)

D[5:4] CFP72[1:0]: P72 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A10
00 (R/W): P72 I/O port (default)

D[3:2] CFP71[1:0]: P71 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A9
00 (R/W): P71 I/O port (default)

D[1:0] CFP70[1:0]: P70 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A8
00 (R/W): P70 I/O port (default)

0x442f: P74–P77 Port Function Select Register (P7_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P74–P77 Port Function Select Register (P7_47_CFP)	0x442f (8 bits)	D7–6	CFP77[1:0]	P77 port function select	CFP77[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A15			
					0x0	P77			
		D5–4	CFP76[1:0]	P76 port function select	CFP76[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A14			
					0x0	P76			
		D3–2	CFP75[1:0]	P75 port function select	CFP75[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A13			
					0x0	P75			
		D1–0	CFP74[1:0]	P74 port function select	CFP74[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A12			
					0x0	P74			

This register selects the functions of the P74 to P77 ports.

D[7:6] CFP77[1:0]: P77 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A15
00 (R/W): P77 I/O port (default)

D[5:4] CFP76[1:0]: P76 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A14
00 (R/W): P76 I/O port (default)

D[3:2] CFP75[1:0]: P75 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A13
00 (R/W): P75 I/O port (default)

D[1:0] CFP74[1:0]: P74 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A12
00 (R/W): P74 I/O port (default)

0x4430: P80–P83 Port Function Select Register (P8_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P80–P83 Port Function Select Register (P8_03_CFP)	0x4430 (8 bits)	D7–6	CFP83[1:0]	P83 port function select	CFP83[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A19			
					0x0	P83			
		D5–4	CFP82[1:0]	P82 port function select	CFP82[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A18			
					0x0	P82			
		D3–2	CFP81[1:0]	P81 port function select	CFP81[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A17			
					0x0	P81			
		D1–0	CFP80[1:0]	P80 port function select	CFP80[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A16			
					0x0	P80			

This register selects the functions of the P80 to P83 ports.

D[7:6] CFP83[1:0]: P83 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A19
00 (R/W): P83 I/O port (default)

D[5:4] CFP82[1:0]: P82 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A18
00 (R/W): P82 I/O port (default)

D[3:2] CFP81[1:0]: P81 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A17
00 (R/W): P81 I/O port (default)

D[1:0] CFP80[1:0]: P80 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A16
00 (R/W): P80 I/O port (default)

0x4431: P84–P86 Port Function Select Register (P8_46_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P84–P86 Port Function Select Register (P8_46_CFP)	**0x4431** (8 bits)	D7–6	**–**	reserved	–		–	–	0 when being read.
		D5–4	**CFP86[1:0]**	P86 port function select	CFP86[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	CMU_CLK			
					0x1	A22			
					0x0	P86			
		D3–2	**CFP85[1:0]**	P85 port function select	CFP85[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A21			
					0x0	P85			
		D1–0	**CFP84[1:0]**	P84 port function select	CFP84[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A20			
					0x0	P84			

This register selects the functions of the P84 to P86 ports.

D[7:6] Reserved

D[5:4] CFP86[1:0]: P86 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): CMU_CLK
01 (R/W): A22
00 (R/W): P86 I/O port (default)

D[3:2] CFP85[1:0]: P85 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A21
00 (R/W): P85 I/O port (default)

D[1:0] CFP84[1:0]: P84 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): A20
00 (R/W): P84 I/O port (default)

0x4432: P90–P93 Port Function Select Register (P9_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P90–P93 Port Function Select Register (P9_03_CFP)	0x4432 (8 bits)	D7–6	CFP93[1:0]	P93 port function select	CFP93[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D3			
					0x0	P93			
		D5–4	CFP92[1:0]	P92 port function select	CFP92[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D2			
					0x0	P92			
		D3–2	CFP91[1:0]	P91 port function select	CFP91[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D1			
					0x0	P91			
		D1–0	CFP90[1:0]	P90 port function select	CFP90[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D0			
					0x0	P90			

This register selects the functions of the P90 to P93 ports.

D[7:6] CFP93[1:0]: P93 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D3
00 (R/W): P93 I/O port (default)

D[5:4] CFP92[1:0]: P92 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D2
00 (R/W): P92 I/O port (default)

D[3:2] CFP91[1:0]: P91 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D1
00 (R/W): P91 I/O port (default)

D[1:0] CFP90[1:0]: P90 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D0
00 (R/W): P90 I/O port (default)

0x4433: P94–P97 Port Function Select Register (P9_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P94–P97 Port Function Select Register (P9_47_CFP)	**0x4433** (8 bits)	D7–6	**CFP97[1:0]**	P97 port function select	CFP97[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D7			
					0x0	P97			
		D5–4	**CFP96[1:0]**	P96 port function select	CFP96[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D6			
					0x0	P96			
		D3–2	**CFP95[1:0]**	P95 port function select	CFP95[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D5			
					0x0	P95			
		D1–0	**CFP94[1:0]**	P94 port function select	CFP94[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D4			
					0x0	P94			

This register selects the functions of the P94 to P97 ports.

D[7:6] CFP97[1:0]: P97 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D7
00 (R/W): P97 I/O port (default)

D[5:4] CFP96[1:0]: P96 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D6
00 (R/W): P96 I/O port (default)

D[3:2] CFP95[1:0]: P95 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D5
00 (R/W): P95 I/O port (default)

D[1:0] CFP94[1:0]: P94 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D4
00 (R/W): P94 I/O port (default)

0x4434: PA0–PA3 Port Function Select Register (PA_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PA0–PA3 Port Function Select Register (PA_03_CFP)	0x4434 (8 bits)	D7–6	CFPA3[1:0]	PA3 port function select	CFPA3[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE3			
					0x0	PA3			
		D5–4	CFPA2[1:0]	PA2 port function select	CFPA2[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE2			
					0x0	PA2			
		D3–2	CFPA1[1:0]	PA1 port function select	CFPA1[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE1			
					0x0	PA1			
		D1–0	CFPA0[1:0]	PA0 port function select	CFPA0[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE0			
					0x0	PA0			

This register selects the functions of the PA0 to PA3 ports.

D[7:6] CFPA3[1:0]: PA3 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #CE3
00 (R/W): PA3 I/O port (default)

D[5:4] CFPA2[1:0]: PA2 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #CE2
00 (R/W): PA2 I/O port (default)

D[3:2] CFPA1[1:0]: PA1 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #CE1
00 (R/W): PA1 I/O port (default)

D[1:0] CFPA0[1:0]: PA0 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #CE0
00 (R/W): PA0 I/O port (default)

0x4435: PA4–PA6 Port Function Select Register (PA_46_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PA4–PA6 Port Function Select Register (PA_46_CFP)	0x4435 (8 bits)	D7–6	–	reserved	–		–	–	0 when being read.
		D5–4	CFPA6[1:0]	PA6 port function select	CFPA6[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#WRH/#BSH			
					0x0	PA6			
		D3–2	CFPA5[1:0]	PA5 port function select	CFPA5[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#WRL			
					0x0	PA5			
		D1–0	CFPA4[1:0]	PA4 port function select	CFPA4[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#RD			
					0x0	PA4			

This register selects the functions of the PA4 to PA6 ports.

D[7:6] Reserved

D[5:4] CFPA6[1:0]: PA6 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #WRH/#BSH
00 (R/W): PA6 I/O port (default)

D[3:2] CFPA5[1:0]: PA5 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #WRL
00 (R/W): PA5 I/O port (default)

D[1:0] CFPA4[1:0]: PA4 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): #RD
00 (R/W): PA4 I/O port (default)

0x4436: PB0–PB3 Port Function Select Register (PB_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PB0–PB3 Port Function Select Register (PB_03_CFP)	0x4436 (8 bits)	D7–6	CFPB3[1:0]	PB3 port function select	CFPB3[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D11			
					0x0	PB3			
		D5–4	CFPB2[1:0]	PB2 port function select	CFPB2[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D10			
					0x0	PB2			
		D3–2	CFPB1[1:0]	PB1 port function select	CFPB1[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D9			
					0x0	PB1			
		D1–0	CFPB0[1:0]	PB0 port function select	CFPB0[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D8			
					0x0	PB0			

This register selects the functions of the PB0 to PB3 ports.

D[7:6] CFPB3[1:0]: PB3 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D11
00 (R/W): PB3 I/O port (default)

D[5:4] CFPB2[1:0]: PB2 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D10
00 (R/W): PB2 I/O port (default)

D[3:2] CFPB1[1:0]: PB1 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D9
00 (R/W): PB1 I/O port (default)

D[1:0] CFPB0[1:0]: PB0 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D8
00 (R/W): PB0 I/O port (default)

0x4437: PB4–PB7 Port Function Select Register (PB_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PB4–PB7 Port Function Select Register (PB_47_CFP)	0x4437 (8 bits)	D7–6	CFPB7[1:0]	PB7 port function select	CFPB7[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D15			
					0x0	PB7			
		D5–4	CFPB6[1:0]	PB6 port function select	CFPB6[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D14			
					0x0	PB6			
		D3–2	CFPB5[1:0]	PB5 port function select	CFPB5[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D13			
					0x0	PB5			
		D1–0	CFPB4[1:0]	PB4 port function select	CFPB4[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D12			
					0x0	PB4			

This register selects the functions of the PB4 to PB7 ports.

D[7:6] CFPB7[1:0]: PB7 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D15
00 (R/W): PB7 I/O port (default)

D[5:4] CFPB6[1:0]: PB6 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D14
00 (R/W): PB6 I/O port (default)

D[3:2] CFPB5[1:0]: PB5 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D13
00 (R/W): PB5 I/O port (default)

D[1:0] CFPB4[1:0]: PB4 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): D12
00 (R/W): PB4 I/O port (default)

0x4438: PC0–PC3 Port Function Select Register (PC_03_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PC0–PC3 Port Function Select Register (PC_03_CFP)	0x4438 (8 bits)	D7–6	CFPC3[1:0]	PC3 port function select	CFPC3[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	PWMPRT0			
					0x1	reserved			
					0x0	PC3			
		D5–4	CFPC2[1:0]	PC2 port function select	CFPC2[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#ADTRG			
					0x1	reserved			
					0x0	PC2			
		D3–2	CFPC1[1:0]	PC1 port function select	CFPC1[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#WDT_NMI			
					0x1	reserved			
					0x0	PC1			
		D1–0	CFPC0[1:0]	PC0 port function select	CFPC0[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	CMU_CLK			
					0x1	reserved			
					0x0	PC0			

This register selects the functions of the PC0 to PC3 ports.

D[7:6] CFPC3[1:0]: PC3 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): PWMPRT0
01 (R/W): Reserved
00 (R/W): PC3 I/O port (default)

D[5:4] CFPC2[1:0]: PC2 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #ADTRG
01 (R/W): Reserved
00 (R/W): PC2 I/O port (default)

D[3:2] CFPC1[1:0]: PC1 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): #WDT_NMI
01 (R/W): Reserved
00 (R/W): PC1 I/O port (default)

D[1:0] CFPC0[1:0]: PC0 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): CMU_CLK
01 (R/W): Reserved
00 (R/W): PC0 I/O port (default)

0x4439: PC4–PC7 Port Function Select Register (PC_47_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PC4–PC7 Port Function Select Register (PC_47_CFP)	0x4439 (8 bits)	D7–6	CFPC7[1:0]	PC7 port function select	CFPC7[1:0]	Function	0x0	R/W	
					0x3–0x1	reserved			
					0x0	PC7			
		D5–4	CFPC6[1:0]	PC6 port function select	CFPC6[1:0]	Function	0x0	R/W	
					0x3–0x1	reserved			
					0x0	PC6			
		D3–2	CFPC5[1:0]	PC5 port function select	CFPC5[1:0]	Function	0x0	R/W	
					0x3–0x1	reserved			
					0x0	PC5			
		D1–0	CFPC4[1:0]	PC4 port function select	CFPC4[1:0]	Function	0x0	R/W	
					0x3–0x1	reserved			
					0x0	PC4			

This register selects the functions of the PC4 to PC7 ports.

D[7:6] CFPC7[1:0]: PC7 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): Reserved
00 (R/W): PC7 I/O port (default)

D[5:4] CFPC6[1:0]: PC6 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): Reserved
00 (R/W): PC6 I/O port (default)

D[3:2] CFPC5[1:0]: PC5 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): Reserved
00 (R/W): PC5 I/O port (default)

D[1:0] CFPC4[1:0]: PC4 Port Function Select Bits

11 (R/W): Reserved
10 (R/W): Reserved
01 (R/W): Reserved
00 (R/W): PC4 I/O port (default)

I.3.4 Input/Output Cells and Input/Output Characteristics

Table I.3.4.1 Pin Characteristics

Pin name	Direction	Cell name	Input level	I_{OH}/I_{OL}	Pull-up/down
OSC3	I	XLIN	–	–	–
OSC4	O	XLOT	–	–	–
OSC1	I	XLIN	–	–	–
OSC2	O	XLOT	–	–	–
TEST	I	XITST1	–	–	50k pull-down
#RESET	I	XIBHP2	LVTTL Schmitt	–	100k pull-up
#NMI	I	XIBHP2	LVTTL Schmitt	–	100k pull-up
DCLK (P35)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
DSIO (P36)	I/O	XBH1P2T	LVTTL Schmitt	2 mA	100k pull-up
DST2 (P37)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P90 (D0)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P91 (D1)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P92 (D2)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P93 (D3)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P94 (D4)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P95 (D5)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P96 (D6)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P97 (D7)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB0 (D8)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB1 (D9)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB2 (D10)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB3 (D11)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB4 (D12)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB5 (D13)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB6 (D14)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
PB7 (D15)	I/O	XBC1HT	LVTTL	2 mA	Bus hold latch
P60 (A0/ #BSL)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P61 (A1)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P62 (A2)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P63 (A3)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P64 (A4)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P65 (A5)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P66 (A6)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P67 (A7)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P70 (A8)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P71 (A9)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P72 (A10)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P73 (A11)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P74 (A12)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P75 (A13)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P76 (A14)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P77 (A15)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P80 (A16)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P81 (A17)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P82 (A18)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P83 (A19)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P84 (A20)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P85 (A21)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
P86 (A22/CMU_CLK)	I/O	XBH1T	LVTTL Schmitt	2 mA	–
PA0 (#CE0)	I/O	XBHMP2T	LVTTL Schmitt	1 mA	100k pull-up
PA1 (#CE1)	I/O	XBHMP2T	LVTTL Schmitt	1 mA	100k pull-up
PA2 (#CE2)	I/O	XBHMP2T	LVTTL Schmitt	1 mA	100k pull-up
PA3 (#CE3)	I/O	XBHMP2T	LVTTL Schmitt	1 mA	100k pull-up
PA4 (#RD)	I/O	XBH1P2T	LVTTL Schmitt	2 mA	100k pull-up
PA5 (#WRL)	I/O	XBH1P2T	LVTTL Schmitt	2 mA	100k pull-up
PA6 (#WRH/#BSH)	I/O	XBH1P2T	LVTTL Schmitt	2 mA	100k pull-up
WAKEUP	O	XOB1T	–	2 mA	–
#STBY	I	XIBH	LVTTL Schmitt	–	–

Pin name	Direction	Cell name	Input level	I_{OH}/I_{OL}	Pull-up/down
P00 (AIN0)	I	XHIBCLINW	CMOS/LVTTL	–	–
P01 (AIN1)	I	XHIBCLINW	CMOS/LVTTL	–	–
P02 (AIN2)	I	XHIBCLINW	CMOS/LVTTL	–	–
P03 (AIN3)	I	XHIBCLINW	CMOS/LVTTL	–	–
P04 (AIN4)	I	XHIBCLINW	CMOS/LVTTL	–	–
P05 (AIN5)	I	XHIBCLINW	CMOS/LVTTL	–	–
P06 (AIN6)	I	XHIBCLINW	CMOS/LVTTL	–	–
P07 (AIN7)	I	XHIBCLINW	CMOS/LVTTL	–	–
P10 (SPI_SDI0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P11 (SPI_SDO0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P12 (SPI_SCK0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P13 (#SPI_SSI0/#SPI_SSI1/SPI_SDI0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P14 (SPI_SDI1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P15 (SPI_SDO1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P16 (SPI_SCK1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P20 (I2S_SDO0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P21 (I2S_WS0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P22 (I2S_SCK0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P23 (I2S_MCLK0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P24 (I2S_SDI1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P25 (I2S_WS1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P26 (I2S_SCK1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P27 (I2S_MCLK1) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P30 (TM0) ∗1	I/O	XBG3A	LVTTL Schmitt	12 mA	–
P31 (#TM0/#ADTRG) ∗1	I/O	XBG3A	LVTTL Schmitt	12 mA	–
P32 (WDT_CLK/#WDT_NMI/CMU_CLK) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P33 (#SMRD/#TM0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P34 (#SMWR) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P40 (SIN0/#SMRD) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P41 (SOUT0/#SMWR) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P42 (#SCLK0/EXCL0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P43 (REMC_IN/PWMPRT0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P44 (REMC_OUT) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P45 (EXCL1/#WAIT)	I/O	XBB1P2	LVTTL	2 mA	100k pull-up
P50 (I2C_SDA/EXCL0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P51 (I2C_SCL) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P52 (REMC_IN/#TM0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P53 (REMC_OUT) ∗1	I/O	XBG3A	LVTTL Schmitt	12 mA	–
P54 (EXCL0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P55 ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P56 (#TM0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
P57 ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC0 (CMU_CLK) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC1 (#WDT_NMI) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC2 (#ADTRG) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC3 (PWMPRT0) ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC4 ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC5 ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC6 ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
PC7 ∗1	I/O	XBG1	LVTTL Schmitt	2 mA	–
USBDP	I/O	XUSFULL	–	–	–
USBDM	I/O	XUSFULL	–	–	–
USBVBUS	I	–	–	–	–
PUENB	O	–	–	–	–
TVEP	I	EIEF_HZTSTTX501	–	–	–

∗1 These I/O cells are Fail-Safe bidirectional buffers for 5 V tolerant.

I.3.5 Package

I.3.5.1 TQFP15-128pin Package

(Unit: mm)

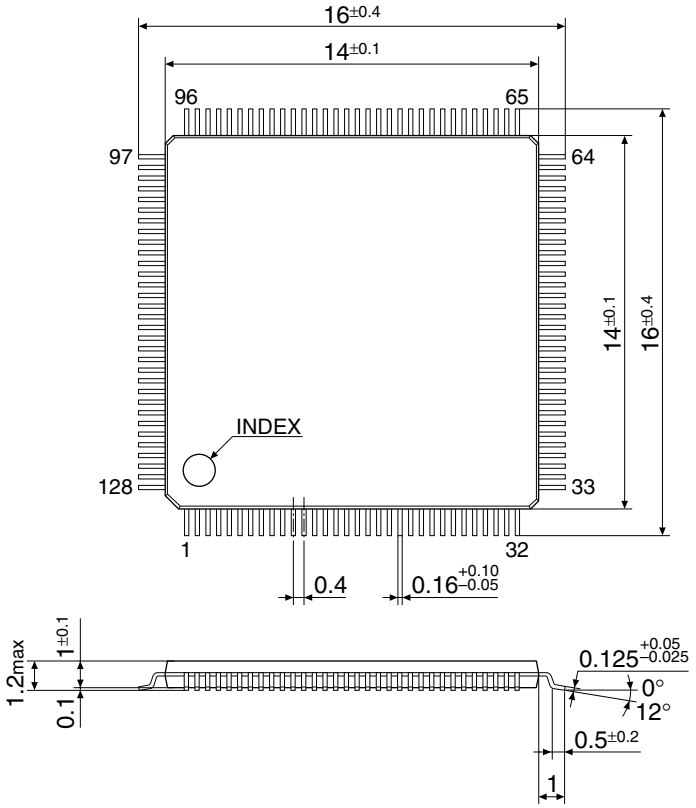


Figure I.3.5.1.1 TQFP15-128pin Package Dimensions

I.3.5.2 TQFP14-100pin Package

(Unit: mm)

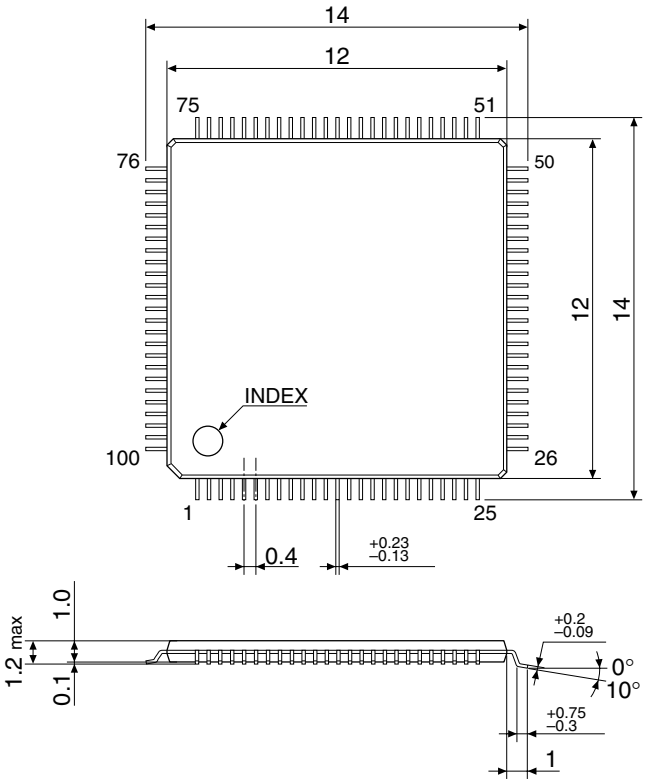


Figure I.3.5.2.1 TQFP14-100pin Package Dimensions

I.3.5.3 Thermal Resistance of the Package

The chip temperature of LSI devices tends to increase with the power consumed on the chip. The chip temperature when encapsulated in a package is calculated from its ambient temperature (Ta), the thermal resistance of the package (θ), and power dissipation (P_D).

Chip temperature (Tj) = Ta + ($P_D \times \theta$) [°C]

Make sure that the chip temperature (Tj) is 125°C or less during Flash reading, or 100°C or less during Flash erasing/programming and USB operation.

Thermal resistance of the TQFP15-128pin package and TQFP14-100pin package

1. When mounted on a board (windless condition)

Thermal resistance (θj-a) = 33.3°C/W

This value indicates the thermal resistance of the package when measured under a windless condition, with the sample mounted on a measurement board (size: 114 × 76 × 1.6 mm thick, FR4/4 layered board).

2. When suspended alone (windless condition)

Thermal resistance = 90–100°C/W

This value indicates the thermal resistance of the package when measured under a windless condition, with the sample suspended alone.

Note: The thermal resistance of the package varies significantly depending on how it is mounted on the board and whether forcibly air-cooled.

When you use a TQFP14-100pin package-type product, note that the package contains following pins with floating electric potential. The pins must therefore be switched to the Low output to maintain the stability.
Failure to do so could increase current consumption.

P33, P34
P54, P55, P56, P57
P83, P84, P85, P86
PA6
PB0, PB1, PB2, PB3, PB4, PB5, PB6, PB7
PC0, PC1, PC2, PC3, PC4, PC5, PC6, PC7

Following processing is required within the initialization of the software.
(You may change settings that do not affect the pins.)

0x4406 P3 Port Input/Output Data Register (P3_DAT) = 0x00
0x440a P5 Port Input/Output Data Register (P5_DAT) = 0x00
0x4410 P8 Port Input/Output Data Register (P8_DAT) = 0x00
0x4414 PA Port Input/Output Data Register (PA_DAT) = 0x00
0x4416 PB Port Input/Output Data Register (PB_DAT) = 0x00
0x4418 PC Port Input/Output Data Register (PC_DAT) = 0x00

0x4407 P3 I/O Control Register (P3_IOC) = 0x14
0x440b P5 I/O Control Register (P5_IOC) = 0xF0
0x4411 P8 I/O Control Register (P8_IOC) = 0x74
0x4415 PA I/O Control Register (PA_IOC) = 0x40
0x4417 PB I/O Control Register (PB_IOC) = 0xFF
0x4419 PC I/O Control Register (PC_IOC) = 0xFF

0x4426 P30-P33 Port Function Select Register (P3_03_CFP) = 0x00 (Init.)
0x4427 P34-P37 Port Function Select Register (P3_47_CFP) = 0x00 (Init.)
0x442a P50-P53 Port Function Select Register (P5_03_CFP) = 0x00 (Init.)
0x442b P54-P57 Port Function Select Register (P5_47_CFP) = 0x00 (Init.)
0x4430 P80-P83 Port Function Select Register (P8_03_CFP) = 0x00 (Init.)
0x4431 P84-P86 Port Function Select Register (P8_46_CFP) = 0x00 (Init.)
0x4434 PA0-PA3 Port Function Select Register (PA_03_CFP) = 0x00 (Init.)
0x4435 PA4-PA6 Port Function Select Register (PA_46_CFP) = 0x00 (Init.)
0x4436 PB0-PB3 Port Function Select Register (PB_03_CFP) = 0x00 (Init.)
0x4437 PB4-PB7 Port Function Select Register (PB_47_CFP) = 0x00 (Init.)
0x4438 PC0-PC3 Port Function Select Register (PC_03_CFP) = 0x00 (Init.)
0x4439 PC4-PC7 Port Function Select Register (PC_47_CFP) = 0x00 (Init.)

I.4 Power Supply

This section explains the operating voltage of the S1C17501.

I.4.1 Power Supply Pins

The S1C17501 has the power supply pins shown in Table I.4.1.1.

Table I.4.1.1 Power Supply Pins

Pin name	Pin No.		I/O	Type	PU/PD	Description
	TQFP 128pin	TQFP 100pin				
VDD	6, 26, 40, VSS, 53, 71, 116	6,19,33,42,57,92	–	3.3 V	–	Core and I/O power supply (+) (3.3 V)
VSS	5, 15, 43, 87, 105, 125	5,12,36,66,84,97	–	GND	–	GND
RTCVDD	35	28	–	3.3 V	–	RTC power supply (+) (3.3 V) (RTCVDD = VDD)
AVDD	96	75	–	3.3 V	–	Analog power supply (3.3 V) (AVDD = VDD)

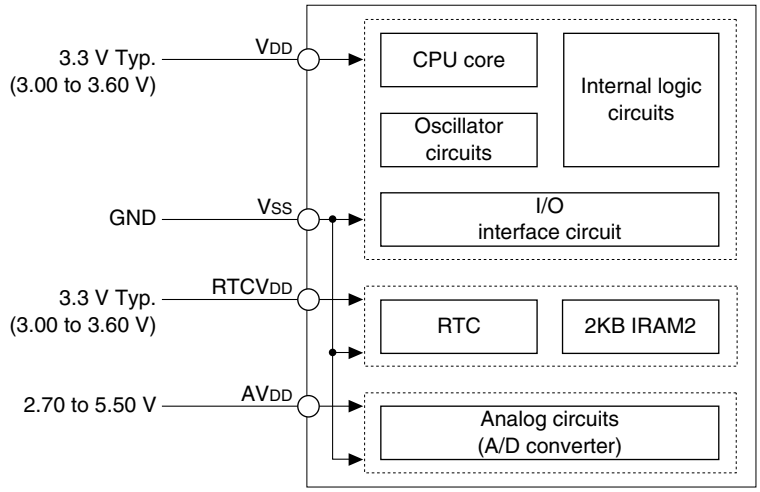


Figure I.4.1.1 Power Supply System

I.4.2 Operating Voltage (V_{DD}, V_{SS})

The core CPU and internal logic circuits operate with a voltage supplied between the V_{DD} and V_{SS} pins.
The following operating voltage can be used:

V_{DD} = 3.0 V to 3.6 V (3.3 V ± 0.3 V, V_{SS} = GND)

Note: The S1C17501 TQFP package has seven V_{DD} pins and six V_{SS} pins. The PFBGA package has 13 V_{DD} pins and 12 V_{SS} pins. Be sure to supply the operating voltage to all the pins. Do not open any of them.

Also the V_{DD} voltage is used for interfacing with external I/O signals except the following I/O ports:

P10–P16, P20–P27, P30–P34, P40–P44, P50–P57, PC0–PC7 (41 ports)

These ports allows an input voltage within -0.3 to 5.8 V.

I.4.3 Power Supply for RTC ($RTCV_{DD}$)

The RTC power supply pin ($RTCV_{DD}$) is provided separately from the V_{DD} pin in order to run RTC at system power down. Supply the same voltage level as the V_{DD} to the $RTCV_{DD}$ pin.

$RTCV_{DD}$ = V_{DD} (3.3 V ± 0.3 V, V_{SS} = GND)

The $RTCV_{DD}$ is also used for the 2KB battery-backup RAM (IRAM2).

I.4.4 Power Supply for Analog Circuits (AV_{DD})

The analog power supply pin (AV_{DD}) is provided separately from the V_{DD} pin in order that the digital circuits do not affect the analog circuit (A/D converter). The AV_{DD} pin is used to supply an analog power voltage and the V_{SS} pin is used as the analog ground.
The following voltage is enabled for AV_{DD}:

AV_{DD} = 2.7 V to 5.5 V (V_{SS} = GND)

Note: Be sure to supply a voltage within the range from 2.7 to 5.5 V to the AV_{DD} pin even if the analog circuit is not used. It is not necessary to supply a voltage same as the V_{DD} level.

Noise on the analog power lines decrease the A/D converting precision, so use a stabilized power supply and make the board pattern with consideration given to that.

I.4.5 Precautions on Power Supply

Power-on sequence

In order to operate the device normally, supply power in accordance with the following timing.

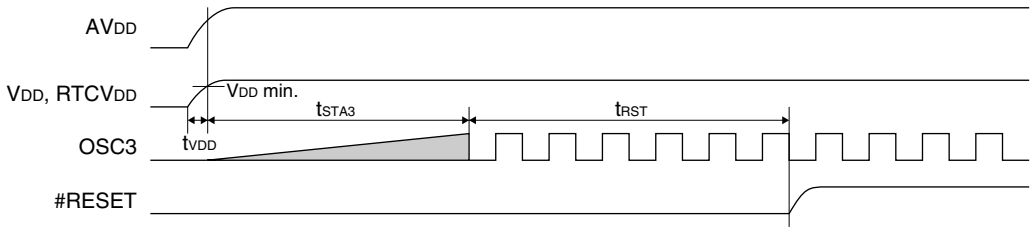


Figure I.4.5.1 Power-On Sequence

(1) tVDD: Elapsed time until the power supply stabilizes after power-on
Supply power in the following sequence.
Power-on: VDD (and RTCVDD*) → AVDD (A/D) → Apply the input signal
or VDD (and RTCVDD*), AVDD (A/D) → Apply the input signal
(See Notes in "Power-off sequence" below.)

* The RTCVDD can be always supplied to the chip to operate the RTC and to back up the IRAM2.

(2) tSTA3: Time at which OSC3 oscillation starts

(3) tRST: Minimum reset pulse width
Time at which the clock supplied to the chip stabilizes plus at least six clocks; Keep the #RESET signal low.

Power-off sequence

Shut off the power supply in the following sequence.
Power-off: Turn off the input signal → AVDD (A/D) → VDD (and RTCVDD)
or Turn off the input signal → AVDD (A/D), VDD (and RTCVDD) (See Notes below.)

Notes: • Applying only VDD makes a diode circuit on the path from VDD to AVDD that results current flowing to the AVDD power supply. In order to avoid this statue, the power supplies should be turned off simultaneously.

• Be sure to avoid applying AVDD for a duration of one second or more when the VDD power is off, as a breakdown may occur in the device or the characteristics may be degraded due to flow-through current of the AVDD.

Latch-up

The CMOS device may be in the latch-up condition. This is the phenomenon caused by conduction of the parasitic PNPN junction (thyristor) contained in the CMOS IC, resulting in a large current between VDD and VSS and leading to breakage.
Latch-up occurs when the voltage applied to the input / output exceeds the rated value and a large current flows into the internal element, or when the voltage at the VDD pin exceeds the rated value and the internal element is in the breakdown condition. In the latter case, even if the application of a voltage exceeding the rated value is instantaneous, the current remains high between VDD and VSS once the device is in the latch-up condition. As this may result in heat generation or smoking, the following points must be taken into consideration:

(1) The voltage level at the input / output must not exceed the range specified in the electrical characteristics. In other words, it must be below the power-supply voltage and above VSS. The power-on timing should also be taken into consideration.
(2) Abnormal noise must not be applied to the device.
(3) The potential at the unused input should be fixed at VDD, AVDD, or VSS.
(4) No outputs should be shorted.

I.5 CPU

The S1C17501 contains the S1C17 Core as its core processor.
The S1C17 Core is a Seiko Epson original 16-bit RISC-type processor.
It features low power consumption, high-speed operation, large address space, main instructions executable in one clock cycle, and a small sized design. The S1C17 Core is suitable for embedded applications that do not need a lot of data processing power like the S1C33 Cores the high-end processors, such as controllers and sequencers for which an eight-bit CPU is commonly used.
For details of the S1C17 Core, refer to the "S1C17 Family S1C17 Core Manual."

I.5.1 Features of the S1C17 Core

Processor type

- Seiko Epson original 16-bit RISC processor
- 0.35–0.15 μm low power CMOS process technology

Instruction set

- Code length: 16-bit fixed length
- Number of instructions: 111 basic instructions (184 including variations)
- Execution cycle: Main instructions executed in one cycles
- Extended immediate instructions: Immediate extended up to 24 bits
- Compact and fast instruction set optimized for development in C language

Register set

- Eight 24-bit general-purpose registers
- Two 24-bit special registers
- One 8-bit special register

Memory space and bus

- Up to 16M bytes of memory space (24-bit address)
- Harvard architecture using separated instruction bus (16 bits) and data bus (32 bits)

Interrupts

- Reset, NMI, and 32 external interrupts supported
- Address misaligned interrupt
- Debug interrupt
- Direct branching from vector table to interrupt handler routine
- Programmable software interrupts with a vector number specified (all vector numbers specifiable)

Power saving

- HALT (`halt` instruction)
- SLEEP (`slp` instruction)

I.5.2 CPU Registers

The S1C17 Core contains eight general-purpose registers and three special registers.

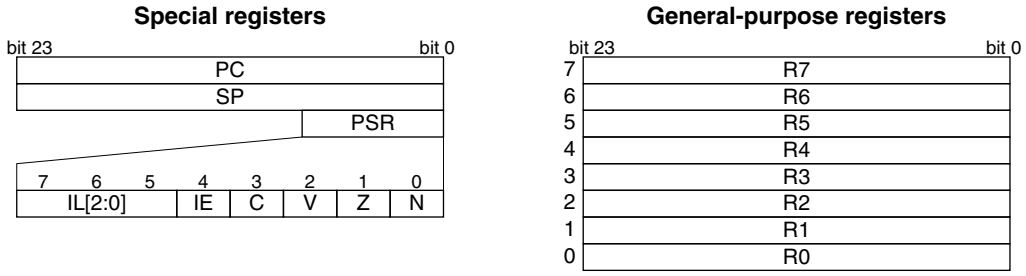


Figure I.5.2.1 Registers

I.5.3 Instruction Set

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

Table I.5.3.1 List of S1C17 Core Instructions

Classification	Mnemonic		Function
Data transfer	ld.b	%rd,%rs	General-purpose register (byte) → general-purpose register (sign-extended)
		%rd,[%rb]	Memory (byte) → general-purpose register (sign-extended)
		%rd,[%rb]+	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		%rd,[%rb]-	
		%rd,-[%rb]	
		%rd,[%sp+imm7]	Stack (byte) → general-purpose register (sign-extended)
		%rd,[imm7]	Memory (byte) → general-purpose register (sign-extended)
		[%rb],%rs	General-purpose register (byte) → memory
		[%rb]+,%rs	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		[%rb]-,%rs	
		-[%rb],%rs	
		[%sp+imm7],%rs	General-purpose register (byte) → stack
		[imm7],%rs	General-purpose register (byte) → memory
	ld.ub	%rd,%rs	General-purpose register (byte) → general-purpose register (zero-extended)
		%rd,[%rb]	Memory (byte) → general-purpose register (zero-extended)
		%rd,[%rb]+	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		%rd,[%rb]-	
		%rd,-[%rb]	
		%rd,[%sp+imm7]	Stack (byte) → general-purpose register (zero-extended)
		%rd,[imm7]	Memory (byte) → general-purpose register (zero-extended)
	ld	%rd,%rs	General-purpose register (16 bits) → general-purpose register
		%rd,sign7	Immediate → general-purpose register (sign-extended)
		%rd,[%rb]	Memory (16 bits) → general-purpose register
		%rd,[%rb]+	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		%rd,[%rb]-	
		%rd,-[%rb]	
		%rd,[%sp+imm7]	Stack (16 bits) → general-purpose register
		%rd,[imm7]	Memory (16 bits) → general-purpose register
		[%rb],%rs	General-purpose register (16 bits) → memory
		[%rb]+,%rs	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		[%rb]-,%rs	
		-[%rb],%rs	
		[%sp+imm7],%rs	General-purpose register (16 bits) → stack
		[imm7],%rs	General-purpose register (16 bits) → memory
	ld.a	%rd,%rs	General-purpose register (24 bits) → general-purpose register
		%rd,imm7	Immediate → general-purpose register (zero-extended)
		%rd,[%rb]	Memory (32 bits) → general-purpose register *
		%rd,[%rb]+	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		%rd,[%rb]-	
		%rd,-[%rb]	
		%rd,[%sp+imm7]	Stack (32 bits) → general-purpose register *
		%rd,[imm7]	Memory (32 bits) → general-purpose register *
		[%rb],%rs	General-purpose register (32 bits, zero-extended) → memory *
		[%rb]+,%rs	Memory address post-increment, post-decrement, and pre-decrement functions can be used.
		[%rb]-,%rs	
		-[%rb],%rs	
		[%sp+imm7],%rs	General-purpose register (32 bits, zero-extended) → stack *
		[imm7],%rs	General-purpose register (32 bits, zero-extended) → memory *
		%rd,%sp	SP → general-purpose register
		%rd,%pc	PC → general-purpose register
		%rd,[%sp]	Stack (32 bits) → general-purpose register *
		%rd,[%sp]+	Stack pointer post-increment, post-decrement, and pre-decrement functions can be used.
		%rd,[%sp]-	
		%rd,-[%sp]	

Classification	Mnemonic		Function
Data transfer	ld.a	[%sp],%*rs*	General-purpose register (32 bits, zero-extended) → stack *
		[%sp]+,%*rs*	Stack pointer post-increment, post-decrement, and pre-decrement functions
		[%sp]-,%*rs*	can be used.
		-[%sp],%*rs*	
		%sp,%*rs*	General-purpose register (24 bits) → SP
		%sp,*imm7*	Immediate → SP
Integer arithmetic operation	add	%*rd*,%*rs*	16-bit addition between general-purpose registers
	add/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	add/nc		
	add	%*rd*,*imm7*	16-bit addition of general-purpose register and immediate
	add.a	%*rd*,%*rs*	24-bit addition between general-purpose registers
	add.a/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	add.a/nc		
	add.a	%sp,%*rs*	24-bit addition of SP and general-purpose register
		%*rd*,*imm7*	24-bit addition of general-purpose register and immediate
		%sp,*imm7*	24-bit addition of SP and immediate
	adc	%*rd*,%*rs*	16-bit addition with carry between general-purpose registers
	adc/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	adc/nc		
	adc	%*rd*,*imm7*	16-bit addition of general-purpose register and immediate with carry
	sub	%*rd*,%*rs*	16-bit subtraction between general-purpose registers
	sub/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	sub/nc		
	sub	%*rd*,*imm7*	16-bit subtraction of general-purpose register and immediate
	sub.a	%*rd*,%*rs*	24-bit subtraction between general-purpose registers
	sub.a/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	sub.a/nc		
	sub.a	%sp,%*rs*	24-bit subtraction of SP and general-purpose register
		%*rd*,*imm7*	24-bit subtraction of general-purpose register and immediate
		%sp,*imm7*	24-bit subtraction of SP and immediate
	sbc	%*rd*,%*rs*	16-bit subtraction with carry between general-purpose registers
	sbc/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	sbc/nc		
	sbc	%*rd*,*imm7*	16-bit subtraction of general-purpose register and immediate with carry
	cmp	%*rd*,%*rs*	16-bit comparison between general-purpose registers
	cmp/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	cmp/nc		
	cmp	%*rd*,*sign7*	16-bit comparison of general-purpose register and immediate
	cmp.a	%*rd*,%*rs*	24-bit comparison between general-purpose registers
	cmp.a/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	cmp.a/nc		
	cmp.a	%*rd*,*imm7*	24-bit comparison of general-purpose register and immediate
	cmc	%*rd*,%*rs*	16-bit comparison with carry between general-purpose registers
	cmc/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	cmc/nc		
	cmc	%*rd*,*sign7*	16-bit comparison of general-purpose register and immediate with carry
Logical operation	and	%*rd*,%*rs*	Logical AND between general-purpose registers
	and/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	and/nc		
	and	%*rd*,*sign7*	Logical AND of general-purpose register and immediate
	or	%*rd*,%*rs*	Logical OR between general-purpose registers
	or/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	or/nc		
	or	%*rd*,*sign7*	Logical OR of general-purpose register and immediate
	xor	%*rd*,%*rs*	Exclusive OR between general-purpose registers
	xor/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	xor/nc		
	xor	%*rd*,*sign7*	Exclusive OR of general-purpose register and immediate
	not	%*rd*,%*rs*	Logical inversion between general-purpose registers (1's complement)
	not/c		Supports conditional execution (/c: executed if C = 1, /nc: executed if C = 0).
	not/nc		
	not	%*rd*,*sign7*	Logical inversion of general-purpose register and immediate (1's complement)

Classification	Mnemonic		Function	
Shift and swap	sr	%rd,%rs	Logical shift to the right with the number of bits specified by the register	
		%rd,imm7	Logical shift to the right with the number of bits specified by immediate	
	sa	%rd,%rs	Arithmetic shift to the right with the number of bits specified by the register	
		%rd,imm7	Arithmetic shift to the right with the number of bits specified by immediate	
	sl	%rd,%rs	Logical shift to the left with the number of bits specified by the register	
		%rd,imm7	Logical shift to the left with the number of bits specified by immediate	
	swap	%rd,%rs	Bytewise swap on byte boundary in 16 bits	
Immediate extension	ext	imm13	Extend operand in the following instruction	
Conversion	cv.ab	%rd,%rs	Convert signed 8-bit data into 24 bits	
	cv.as	%rd,%rs	Convert signed 16-bit data into 24 bits	
	cv.al	%rd,%rs	Convert 32-bit data into 24 bits	
	cv.la	%rd,%rs	Converts 24-bit data into 32 bits	
	cv.ls	%rd,%rs	Converts 16-bit data into 32 bits	
Branch	jpr	sign10	PC relative jump	
	jpr.d	%rb	Delayed branching possible	
	jpa	imm7	Absolute jump	
	jpa.d	%rb	Delayed branching possible	
	jrgt	sign7	PC relative conditional jump	Branch condition: !Z & !(N ^ V)
	jrgt.d		Delayed branching possible	
	jrge	sign7	PC relative conditional jump	Branch condition: !(N ^ V)
	jrge.d		Delayed branching possible	
	jrlt	sign7	PC relative conditional jump	Branch condition: N ^ V
	jrlt.d		Delayed branching possible	
	jrle	sign7	PC relative conditional jump	Branch condition: Z | N ^ V
	jrle.d		Delayed branching possible	
	jrugt	sign7	PC relative conditional jump	Branch condition: !Z & !C
	jrugt.d		Delayed branching possible	
	jruge	sign7	PC relative conditional jump	Branch condition: !C
	jruge.d		Delayed branching possible	
	jrult	sign7	PC relative conditional jump	Branch condition: C
	jrult.d		Delayed branching possible	
	jrule	sign7	PC relative conditional jump	Branch condition: Z | C
	jrule.d		Delayed branching possible	
	jreq	sign7	PC relative conditional jump	Branch condition: Z
	jreq.d		Delayed branching possible	
	jrne	sign7	PC relative conditional jump	Branch condition: !Z
	jrne.d		Delayed branching possible	
	call	sign10	PC relative subroutine call	
	call.d	%rb	Delayed call possible	
	calla	imm7	Absolute subroutine call	
	calla.d	%rb	Delayed call possible	
	ret		Return from subroutine	
	ret.d		Delayed return possible	
	int	imm5	Software interrupt	
	intl	imm5,imm3	Software interrupt with interrupt level setting	
	reti		Return from interrupt handling	
	reti.d		Delayed call possible	
	brk		Debug interrupt	
	retd		Return from debug processing	
System control	nop		No operation	
	halt		HALT mode	
	slp		SLEEP mode	
	ei		Enable interrupts	
	di		Disable interrupts	
Coprocessor control	ld.cw	%rd,%rs	Transfer data to coprocessor	
		%rd,imm7		
	ld.ca	%rd,%rs	Transfer data to coprocessor and get results and flag statuses	
		%rd,imm7		
	ld.cf	%rd,%rs	Transfer data to coprocessor and get flag statuses	
		%rd,imm7		

* The ld.a instruction accesses memories in 32-bit length. During data transfer from a register to a memory other than the IRAM area, the 32-bit data in which the eight high-order bits are set to 0 is written to the memory. During data transfer from a register to the IRAM area, the eight high-order bits are not written to the memory.

The symbols in the above table each have the meanings specified below.

Table I.5.3.2 Symbol Meanings

Symbol	Description
%*rs*	General-purpose register, source
%*rd*	General-purpose register, destination
[%*rb*]	Memory addressed by general-purpose register
[%*rb*]+	Memory addressed by general-purpose register with address post-incremented
[%*rb*]-	Memory addressed by general-purpose register with address post-decremented
-[%*rb*]	Memory addressed by general-purpose register with address pre-decremented
%sp	Stack pointer
[%sp],[%sp+*imm7*]	Stack
[%sp]+	Stack with address post-incremented
[%sp]-	Stack with address post-decremented
-[%sp]	Stack with address pre-decremented
imm3,*imm5*,*imm7*,*imm13*	Unsigned immediate (numerals indicating bit length)
sign7,*sign10*	Signed immediate (numerals indicating bit length)

I.5.4 Vector Table

The vector table contains the vectors to the interrupt handler routines (handler routine start address) that will be read by the S1C17 Core to execute the handler when an interrupt occurs. The boot address from which the program starts running after a reset must be written to the top of the vector table.
Table I.5.4.1 shows the vector table of the S1C17501.

Table I.5.4.1 Vector Table

Vector No. Software interrupt No.	Vector address	Hardware interrupt name	Cause of hardware interrupt	Priority
0 (0x00)	TTBR + 0x00	Reset	• Low input to the #RESET pin • Watchdog timer overflow *2	1
1 (0x01)	TTBR + 0x04	Address misaligned interrupt	Memory access instruction	2
–	(0xfffc00)	Debugging interrupt	`brk` instruction, etc.	3
2 (0x02)	TTBR + 0x08	NMI	• Low input to the #NMI pin • Watchdog timer overflow *2	4
3 (0x03)	TTBR + 0x0c	C compiler (reserved)	Used in emulation library for C compiler	5
4 (0x04)	TTBR + 0x10	Port input interrupt 0	P*x*0 input (rising/falling edge or high/low level)	High *1
5 (0x05)	TTBR + 0x14	Port input interrupt 1	P*x*1 input (rising/falling edge or high/low level)	↑
6 (0x06)	TTBR + 0x18	Port input interrupt 2	P*x*2 input (rising/falling edge or high/low level)	
7 (0x07)	TTBR + 0x1c	Port input interrupt 3	P*x*3 input (rising/falling edge or high/low level)	
8 (0x08)	TTBR + 0x20	MFT interrupt	• Compare-match • Period-match • ADC protection input • Port protection input	
9 (0x09)	TTBR + 0x24	reserved	–	
10 (0x0a)	TTBR + 0x28	A/D converter	Out of range results (upper- and lower-limit)	
11 (0x0b)	TTBR + 0x2c		End of conversion	
12 (0x0c)	TTBR + 0x30	CLG_T16U0 timer interrupt	Timer underflow	
		Port input interrupt 4	P*x*4 input (rising/falling edge or high/low level)	
13 (0x0d)	TTBR + 0x34	Port input interrupt 5	P*x*5 input (rising/falling edge or high/low level)	
14 (0x0e)	TTBR + 0x38	CLG_T8S timer interrupt	Timer underflow	
		Port input interrupt 6	P*x*6 input (rising/falling edge or high/low level)	
15 (0x0f)	TTBR + 0x3c	CLG_T8I timer interrupt	Timer underflow	
		Port input interrupt 7	P*x*7 input (rising/falling edge or high/low level)	
16 (0x10)	TTBR + 0x40	UART with IrDA CH.0 interrupt	• Transmit buffer empty • Receive buffer full • Receive error	
		Port input interrupt 4	P*x*4 input (rising/falling edge or high/low level)	
17 (0x11)	TTBR + 0x44	Port input interrupt 5	P*x*5 input (rising/falling edge or high/low level)	
18 (0x12)	TTBR + 0x48	SPI CH.0 interrupt	• Transmit buffer empty • Receive buffer full	
		Port input interrupt 6	P*x*6 input (rising/falling edge or high/low level)	
19 (0x13)	TTBR + 0x4c	I^2C interrupt	• Transmit completed • Receive completed	
		Port input interrupt 7	P*x*7 input (rising/falling edge or high/low level)	
20 (0x14)	TTBR + 0x50	RTC interrupt	1/64 second, 1 second, 1 minute, or 1 hour count up	
21 (0x15)	TTBR + 0x54	PT8 CH.0 interrupt	Timer 0 underflow	
22 (0x16)	TTBR + 0x58	PT8 CH.1 interrupt	Timer 1 underflow	
23 (0x17)	TTBR + 0x5c	PT8 CH.2 interrupt	Timer 2 underflow	
24 (0x18)	TTBR + 0x60	PT8 CH.3 interrupt	Timer 3 underflow	
25 (0x19)	TTBR + 0x64	reserved	–	
26 (0x1a)	TTBR + 0x68	SPI CH.1 interrupt	• Transmit buffer empty • Receive buffer full	
27 (0x1b)	TTBR + 0x6c	USB function controller interrupt	USB interrupt	
28 (0x1c)	TTBR + 0x70	I^2S interrupt	• I^2S FIFO empty	
29 (0x1d)	TTBR + 0x74		• I^2S FIFO full	
30 (0x1e)	TTBR + 0x78	REMC interrupt	• Envelope counter underflow • REMC_IN rising edge detection • REMC_IN falling edge detection	↓
31 (0x1f)	TTBR + 0x7c	reserved	–	Low *1

*1 When the same interrupt level is set
*2 Either reset or NMI can be selected as the watchdog timer interrupt with software.

The S1C17501 allows the base (starting) address of the vector table to be set using the TTBR_LOW and TTBR_HIGH registers (0x5814, 0x5816). "TTBR" described in Table I.5.4.1 means the value set to these registers. After an initial reset, the TTBR_LOW/HIGH registers are set to 0x20000. Therefore, even when the trap table position is changed, it is necessary that at least the reset vector be written to the above address. Bits 7 to 0 in the TTBR_LOW register are fixed at 0, so the trap table starting address always begins with a 256-byte boundary address.

0x5814–0x5816: Trap Table Base Registers (TTBR_LOW, TTBR_HIGH)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Trap Table Base Register 0 (TTBR_LOW)	**0x5814** (16 bits)	D15–8	**TTBR[15:8]**	Trap table base address A[15:8]	0x0–0xff	0x0	R/W	
		D7–0	**TTBR[7:0]**	Trap table base address A[7:0] (fixed at 0)	0x0	0x0	R	
Trap Table Base Register 1 (TTBR_HIGH)	**0x5816** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TTBR[23:16]**	Trap table base address A[23:16]	0x0–0xff	0x2	R/W	

Note: The Trap Table Base Registers are write-protected. Before these registers can be rewritten, write protection must be removed by writing data 0x96 to the FLASHC Protect Register (0x5810). Note that since unnecessary rewrites to the Trap Table Base Registers could lead to erratic system operation, the FLASHC Protect Register (0x5810) should be set to other than 0x96 unless the Trap Table Base Registers must be rewritten.

I.5.5 On-chip Debugger

I.5.5.1 Debug Functions

The S1C17 Core has an embedded debug unit to assist in software development by the user.
The debug unit provides the following functions that are used with debugging tools:

- Instruction break
  A debug interrupt is generated before the set instruction address is executed. An instruction break can be set at one address location.
- Single step
  A debug interrupt is generated every instruction executed.
- Forcible break
  A debug interrupt is generated by an external input signal (DSIO = 0).
- Software break
  A debug interrupt is generated when the `brk` instruction is executed.

When a debug interrupt occurs, the processor performs the following processing:

(1) Suspends the instructions currently being executed.

(2) Saves the contents of the PC and PSR, and R0, in that order, to the addresses specified below.
PC/PSR → DBRAM + 0x0
R0 → DBRAM + 0x4 (DBRAM: Start address of the work area for debugging in the user RAM)

(3) Loads address 0xfffc00 to PC and branches to the debug interrupt handler routine.

In the interrupt handler routine, the `retd` instruction should be executed at the end of processing to return to the suspended instructions. When returning from the interrupt by the `retd` instruction, the processor restores the saved data in order of the R0 and the PC and PSR.
Neither hardware interrupts nor NMI interrupts are accepted during a debug interrupt.

I.5.5.2 Work Area for Debugging

A 64-byte work area is required for debugging. In the S1C17501, the address range from 0x0 to 0x3f in the internal RAM is reserved as the work area for debugging. When using the debug functions, do not access this area from the application program.
The debug RAM start address can be read out from the DBRAM register (0xffff90).

0xffff90: Debug RAM Base Register (DBRAM)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Debug RAM Base Register (DBRAM)	**0xffff90** (32 bits)	D31–24	–	Unused (fixed at 0)	0x0	0x0	R	
		D23–0	**DBRAM[23:0]**	Debug RAM base address	0x0	0x0	R	

D[31:24] Unused (fixed at 0)

D[23:0] DBRAM[23:0]: Debug RAM Base Address Bits
This is a read-only register that contains the start address of a work area (64 bytes) for debugging.

I.5.5.3 Debugging Tools

Debugging is performed by connecting the ICD (In-Circuit Debugger) such as S5U1C17001H (ICD Mini) to the debug pins of the S1C17501 and entering debug commands from the debugger being run on a personal computer. The tools listed below are required for debugging.
- S1C17 Family In-Circuit Debugger (e.g. S5U1C17001H)
- S1C17 Family C Compiler Package (e.g. S5U1C17001C)

I.5.5.4 Debug Pins

The ICD (e.g. S5U1C17001H) is connected to the debug pins listed below.

Table I.5.5.4.1 List of Debug Pins

Pin name	I/O	Size	Function
DCLK (P35)	O	1	On-chip debugger clock output pin This pin outputs a clock to the ICD.
DSIO (P36)	I/O	1	On-chip debugger data input/output pin This pin inputs/outputs data for debugging and inputs a break signal.
DST2 (P37)	O	1	On-chip debugger status signal output pin This pin outputs the processor status during debugging (goes low in normal mode or goes high in debug mode).

The on-chip debugger input/output pins (DCLK, DSIO, DST2) are shared with the I/O ports (P35, P36, P37) and they are initialized as debug pins by default. When the debug function is not used, these pins can be configured for general-purpose I/O ports using the P3_47_CFP register (0x4427). For details on switching pin function, see Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

I.5.5.5 Clock for Debugging

The embedded debug unit communicates with the ICD in serial data transfer. DCLK is the sync clock for transfer and its frequency is always half of the CCLK frequency.

I.5.5.6 Debugger Status Signal (DST2)

The DST2 signal is set to low during normal operation and it goes high when the S1C17 Core enters debug mode by a debug interrupt.

I.6 Memory Map

Figure I.6.1 shows the S1C17501 memory map.

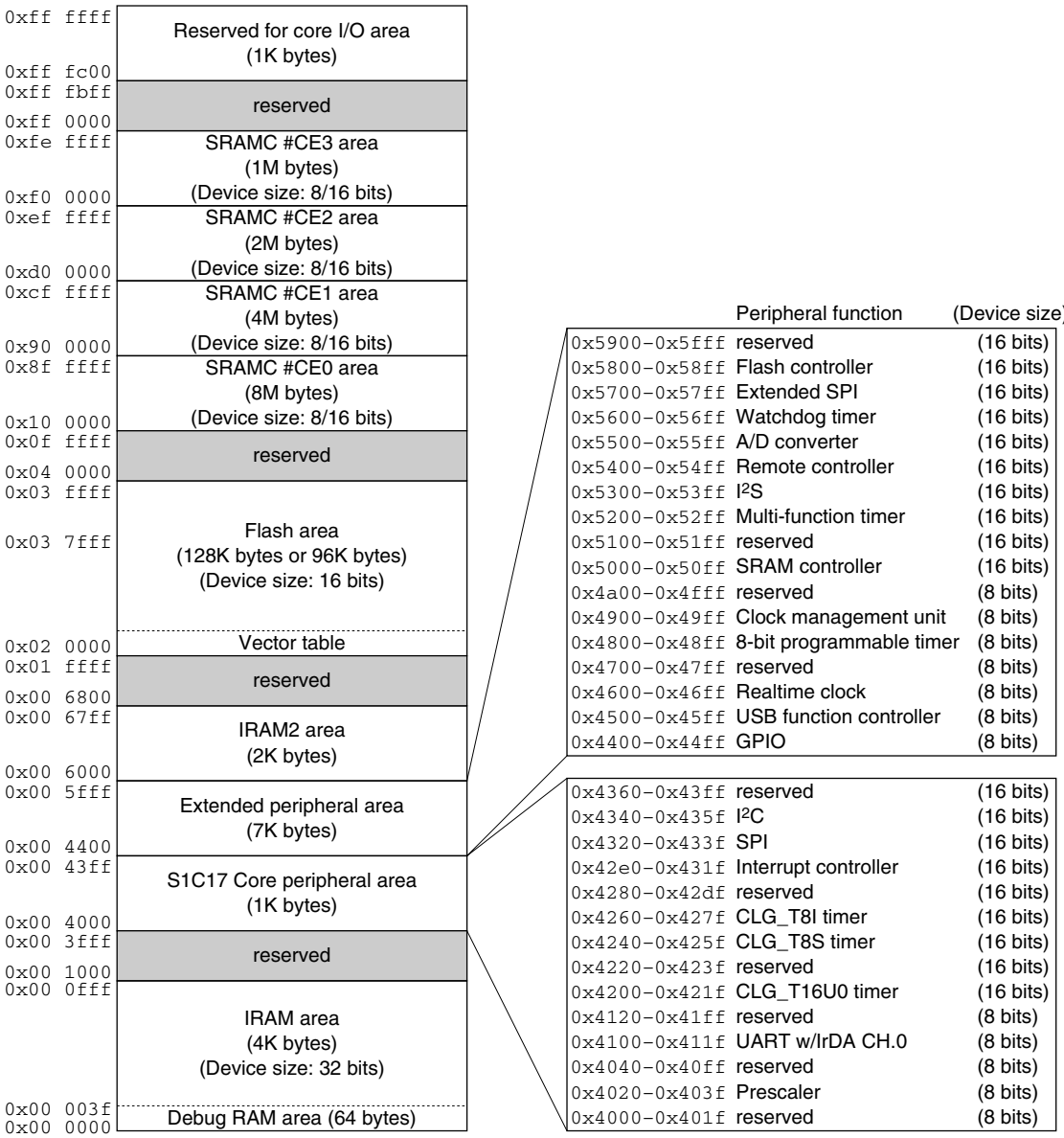


Figure I.6.1 S1C17501 Memory Map

I.6.1 Access Cycle

As shown in Table I.6.1.1, the number of cycles required for one bus access depends on the peripheral or memory module. Furthermore, the number of bus accesses depends on the CPU instruction (access size) and device size.

Table I.6.1.1 Number of Access Cycles for Data Read/Write

Module	Access condition		Write	Read
IRAM	–	8-bit access	1	2
		16-bit access	1	2
		24/32-bit access	1	2
IRAM2	CPU access	8-bit access	4	4 + w
		16-bit access	4	4 + w
		24/32-bit access	7	7 + w × 2
FLASH	Random Miss	8-bit access	–	2 + w
		16-bit access	Software control	2 + w
		24/32-bit access	–	1 + (1 + w) × 2
	Random Hit	8-bit access	–	1
		16-bit access	Software control	1
		24/32-bit access	–	1 + 1 + w
	Burst read (sequential address)	16-bit × N access	–	1 + (1 + w) × N
		24/32-bit × N access	–	1 + (1 + w) × 2 × N
Peripheral control registers	8-bit device	8-bit access	4 + w	4 + w
		16-bit access	7 + w × 2	7 + w × 2
		24-bit access	13 + w × 4	13 + w × 4
	16-bit device	8-bit access	4 + w	4 + w
		16-bit access	4 + w	4 + w
		24-bit access	7 + w × 2	7 + w × 2
External memory	8-bit RAM CPU access	8-bit access	4 + w	4 + w
		16-bit access	7 + w × 2	7 + w × 2
		24/32-bit access	13 + w × 4	13 + w × 4
	16-bit RAM CPU access	8-bit access	4 + w	4 + w
		16-bit access	4 + w	4 + w
		24/32-bit access	7 + w × 2	7 + w × 2
MAC operation	–	–	1 or 2	

Notes: • "w" means the number of wait cycles.

- "N" means the number of burst cycles.
- When MAC_WAIT (D0/0x5014) = 0, a MAC operation needs 1 cycle for processing.
  When MAC_WAIT = 1 (default), a MAC operation needs 2 cycles for processing.

Table I.6.1.2 Number of Access Cycles for Instruction Read

Module	Access condition		Instruction Read
IRAM	CPU read	32-bit read	2 (Note)
IRAM2	CPU read	32-bit read	7 + w × 2
FLASH	Random Miss	32-bit read	1 + (1 + w) × 2
	Random Hit	32-bit read	1 + 1 + w
	Burst read (sequential address)	32-bit × N read	1 + (1 + w) × 2 × N
External memory	8-bit RAM, CPU access	32-bit read	13 + w ×4
	16-bit RAM, CPU access	32-bit read	7 + w × 2

Notes: • "w" means the number of wait cycles.

- "N" means the number of burst cycles.
- The CPU can read a 16-bit instruction from the IRAM in 1 clock cycle.

Handling the eight high-order bits during 32-bit accesses

During writing, the eight high-order bits of 32-bit data are written as 0. However, the eight high-order bits are not written when data is written to IRAM using the "`ld.a`" instruction.

During reading from a memory, the eight high-order bits are ignored. However, the eight high-order bits are effective as the PSR value only in the stack operation when an interrupt occurs.

I.6.1.1 Restrictions on Access Size

The modules shown below have a restriction on the access size. Appropriate instructions should be used in programming.

I²C, WDT

The I²C and watchdog timer registers allow only 16-bit read/write instructions for accessing.

Other peripheral modules can be accessed with an 8-bit, or 16-bit instruction (the I²S FIFO can be accessed with an 16-bit, or 24-bit instruction). However, reading for an unnecessary register may change the peripheral module status and it may cause a problem. Therefore, use the appropriate instructions according to the device size.

I.6.1.2 Simultaneous Access to Instruction and Data by Harvard Architecture

The S1C17 Core has adopted Harvard Architecture. An instruction fetch and a data access are performed simultaneously under one of the conditions listed below, this makes it possible to improve the execution speed.

- When the S1C17501 accesses data in the IRAM area and executes the instruction stored in the Flash area, IRAM2 or an external memory
- When the S1C17501 executes the instruction stored in the IRAM area and accesses data in the Flash area, IRAM2, S1C17 Core peripheral area (0x4000–), extended peripheral area (0x4400–), or an external memory
- When the S1C17501 accesses data in the S1C17 Core peripheral area (0x4000–) and executes the instruction stored in the IRAM, Flash area, IRAM2, or an external memory.

I.6.2 IRAM Area

The S1C17501 contains a RAM in the 4K-byte area from address 0x0 to address 0xfff. The RAM is accessed in one cycle for data writing or two cycles for data reading regardless of the access size. An instruction can be read in one cycle from the IRAM.

Notes: • The 64-byte area at the beginning of the RAM (0x0–0x3f) is reserved for the on-chip debugger. When using the debug functions under application development, do not access this area from the application program.
This area can be used for applications of mass-produced devices that do not need debugging.

• When data is written to IRAM using the "`ld.a`" instruction, the S1C17 Core does not write anything to the eight high-order bits (D[31:24]) of the 32-bit space.
Example: `ld.a [%rb],%rs`

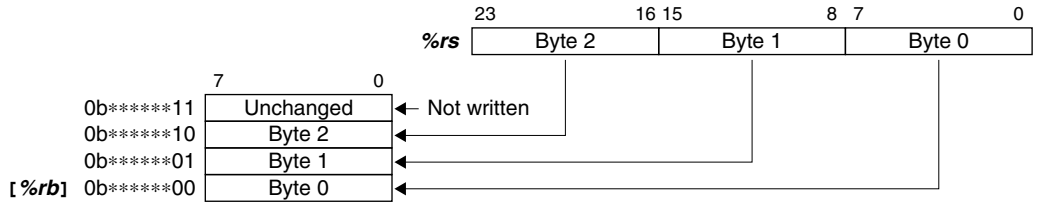


Figure I.6.2.1 24-bit Write to IRAM

I.6.3 IRAM2 Area

The S1C17501 contains a RAM (IRAM2) in the 2K-byte area from address 0x6000 to address 0x67ff.
The IRAM2 uses the RTCV$_{DD}$ voltage to operate, so it works as a battery backup RAM while the system power supply voltage (V$_{DD}$) is turned off.

I.6.4 Flash Area

The S1C17501 contains a Flash memory (4K bytes/sector) in the 128K-byte area from address 0x20000 to address 0x3ffff or the 96K-byte area from address 0x20000 to address 0x37fff for storing application programs and data. Address 0x20000 is defined as the vector table base address by default, therefore the reset vector must be placed on this address. The vector table base address can be changed using the TTBR_LOW/HIGH registers (0x5814, 0x5816).
The Flash memory can be read in a minimum of one cycle.
For more information on the Flash memory, see Section III.1, "Flash Controller (FLASHC)."

Note: When the halt instruction located in the Flash memory is executed, the S1C17501 enters HALT mode with the Flash area chip select signal asserted. This increases current consumption, as the Flash memory stays active during HALT status. Therefore, when setting the S1C17501 into HALT mode, the halt instruction should be executed in IRAM.

I.6.5 Internal Peripheral Area

The I/O and control registers for the internal peripheral modules are located in two areas beginning with addresses 0x4000 and 0x4400.

I.6.5.1 S1C17 Core Peripheral Area (0x4000–)

The S1C17 Core peripheral area beginning with address 0x4000 contains the I/O memory for the peripheral functions included in the Core module listed below and this area can be accessed in a minimum of four cycles.

- Prescaler (PSC, 8-bit device)
- UART (UART, 8-bit device)
- Clock generator (CLG, 16-bit device)
- Interrupt controller (ITC, 16-bit device)
- SPI (SPI, 16-bit device)
- I^2C (I2C, 16-bit device)

I.6.5.2 Extended Peripheral Area (0x4400–)

The extended peripheral area beginning with address 0x4400 contains the I/O memory for the peripheral functions listed below and this area can be accessed in a minimum of four cycles.

- GPIO (GPIO, 8-bit device)
- USB function controller (USB, 8-bit device)
- Realtime clock (RTC, 8-bit device)
- 8-bit programmable timer (PT8, 8-bit device)
- Clock management unit (CMU, 8-bit device)
- SRAM controller (SRAMC, 16-bit device)
- 16-bit Multi-function timer (MFT, 16-bit device)
- I^2S (I2S, 16-bit device)
- Remote controller (REMC, 16-bit device)
- A/D converter (ADC, 16-bit device)
- Watchdog timer (WDT, 16-bit device)
- Extended SPI (ESPI, 16-bit device)
- Flash controller (FLASHC, 16-bit device)

I.6.5.3 I/O Map

This section shows the I/O map table for the internal peripheral area. For details of each control register, see the I/O register list in Appendix or description for each peripheral module.

Table I.6.5.3.1 I/O Map (S1C17 Core Peripheral Area)

Peripheral	Address	Register name		Function
Prescaler (8-bit device)	0x4020	PSC_CTL	Prescaler Control Register	Starts/stops the prescaler.
	0x4021–0x403f	–	–	Reserved
UART (with IrDA) (8-bit device)	0x4100	UART_ST	UART Status Register	Indicates transfer, buffer and error statuses.
	0x4101	UART_TXD	UART Transmit Data Register	Transmit data
	0x4102	UART_RXD	UART Receive Data Register	Receive data
	0x4103	UART_MOD	UART Mode Register	Sets transfer data format.
	0x4104	UART_CTL	UART Control Register	Controls data transfer.
	0x4105	UART_EXP	UART Expansion Register	Sets IrDA mode.
	0x4106–0x411f	–	–	Reserved
CLG_T16U0 timer (16-bit device)	0x4200	CLG_T16U0_CLK	CLG_T16U0 Input Clock Select Register	Selects a prescaler output clock.
	0x4202	CLG_T16U0_TR	CLG_T16U0 Reload Data Register	Sets reload data.
	0x4204	CLG_T16U0_TC	CLG_T16U0 Counter Data Register	Counter data
	0x4206	CLG_T16U0_CTL	CLG_T16U0 Control Register	Sets the timer mode and starts/stops the timer.
	0x4208–0x421f	–	–	Reserved

Peripheral	Address	Register name		Function
CLG_T8S timer (16-bit device)	0x4240	CLG_T8S_CLK	CLG_T8S Input Clock Select Register	Selects a prescaler output clock.
	0x4242	CLG_T8S_TR	CLG_T8S Reload Data Register	Sets reload data.
	0x4244	CLG_T8S_TC	CLG_T8S Counter Data Register	Counter data
	0x4246	CLG_T8S_CTL	CLG_T8S Control Register	Sets the timer mode and starts/stops the timer.
	0x4248–0x425f	–	–	Reserved
CLG_T8I timer (16-bit device)	0x4260	CLG_T8I_CLK	CLG_T8I Input Clock Select Register	Selects a prescaler output clock.
	0x4262	CLG_T8I_TR	CLG_T8I Reload Data Register	Sets reload data.
	0x4264	CLG_T8I_TC	CLG_T8I Counter Data Register	Counter data
	0x4266	CLG_T8I_CTL	CLG_T8I Control Register	Sets the timer mode and starts/stops the timer.
	0x4268–0x427f	–	–	Reserved
Interrupt controller (16-bit device)	0x42e0	ITC_AIFLG	Additional Interrupt Flag Register	Indicates/resets interrupt occurrence status.
	0x42e2	ITC_AEN	Additional Interrupt Enable Register	Enables/disables each maskable interrupt.
	0x42e4	–	–	Reserved
	0x42e6	ITC_AILV0	Additional Interrupt Level Setup Register 0	Sets the MFT interrupt level.
	0x42e8	ITC_AILV1	Additional Interrupt Level Setup Register 1	Sets the ADC interrupt level.
	0x42ea	ITC_AILV2	Additional Interrupt Level Setup Register 2	Sets the RTC and PT8 CH.0 interrupt levels.
	0x42ec	ITC_AILV3	Additional Interrupt Level Setup Register 3	Sets the PT8 CH.1 and CH.2 interrupt levels.
	0x42ee	ITC_AILV4	Additional Interrupt Level Setup Register 4	Sets the PT8 CH.3
	0x42f0	ITC_AILV5	Additional Interrupt Level Setup Register 5	Sets the SPI CH.1 and USB interrupt levels.
	0x42f2	ITC_AILV6	Additional Interrupt Level Setup Register 6	Sets the I²S interrupt level.
	0x42f4	ITC_AILV7	Additional Interrupt Level Setup Register 7	Sets the REMC interrupt level.
	0x42f6–0x42ff	–	–	Reserved
	0x4300	ITC_IFLG	Interrupt Flag Register	Indicates/resets interrupt occurrence status.
Interrupt controller (16-bit device)	0x4302	ITC_EN	Interrupt Enable Register	Enables/disables each maskable interrupt.
	0x4304	ITC_CTL	ITC Control Register	Enables/disables the ITC.
	0x4306	ITC_ELV0	External Interrupt Level Setup Register 0	Sets the port 0 and port 1 interrupt levels and trigger modes.
	0x4308	ITC_ELV1	External Interrupt Level Setup Register 1	Sets the port 2 and port 3 interrupt levels and trigger modes.
	0x430a	ITC_ELV2	External Interrupt Level Setup Register 2	Sets the port 4 and port 5 interrupt levels and trigger modes.
	0x430c	ITC_ELV3	External Interrupt Level Setup Register 3	Sets the port 6 and port 7 interrupt levels and trigger modes.
	0x430e	ITC_ILV0	Internal Interrupt Level Setup Register 0	Sets the CLG_T16U0 timer interrupt level.
	0x4310	ITC_ILV1	Internal Interrupt Level Setup Register 1	Sets the CLG_T8S and CLG_T8I timer interrupt levels.
	0x4312	ITC_ILV2	Internal Interrupt Level Setup Register 2	Sets the UART interrupt level.
	0x4314	ITC_ILV3	Internal Interrupt Level Setup Register 3	Sets the SPI CH.0 and I²C interrupt levels.
	0x4316–0x431f	–	–	Reserved
SPI (16-bit device)	0x4320	SPI_ST0	SPI CH.0 Status Register	Indicates transfer and buffer statuses.
	0x4322	SPI_TXD0	SPI CH.0 Transmit Data Register	Transmit data
	0x4324	SPI_RXD0	SPI CH.0 Receive Data Register	Receive data
	0x4326	SPI_CTL0	SPI CH.0 Control Register	Sets the SPI CH.0 mode and enables data transfer.
	0x4328–0x433f	–	–	Reserved
I²C (16-bit device)	0x4340	I2C_EN	I²C Enable Register	Enables the I²C module.
	0x4342	I2C_CTL	I²C Control Register	Controls the I²C operation and indicates transfer status.
	0x4344	I2C_DAT	I²C Data Register	Transmit/receive data
	0x4346	I2C_ICTL	I²C Interrupt Control Register	Controls the I²C interrupt.
	0x4348–0x435f	–	–	Reserved

Table I.6.5.3.2 I/O Map (Extended Peripheral Area)

Peripheral	Address	Register name		Function
GPIO (8-bit device)	0x4400	P0_DAT	P0 Port Input Data Register	P0 port input data
	0x4401	–	–	Reserved
	0x4402	P1_DAT	P1 Port Input/Output Data Register	P1 port input/output data
	0x4403	P1_IOC	P1 Port I/O Control Register	Selects the P1 port I/O direction.
	0x4404	P2_DAT	P2 Port Input/Output Data Register	P2 port input/output data
	0x4405	P2_IOC	P2 Port I/O Control Register	Selects the P2 port I/O direction.
	0x4406	P3_DAT	P3 Port Input/Output Data Register	P3 port input/output data
	0x4407	P3_IOC	P3 Port I/O Control Register	Selects the P3 port I/O direction.
	0x4408	P4_DAT	P4 Port Input/Output Data Register	P4 port input/output data
	0x4409	P4_IOC	P4 Port I/O Control Register	Selects the P4 port I/O direction.
	0x440a	P5_DAT	P5 Port Input/Output Data Register	P5 port input/output data
	0x440b	P5_IOC	P5 Port I/O Control Register	Selects the P5 port I/O direction.
	0x440c	P6_DAT	P6 Port Input/Output Data Register	P6 port input/output data
	0x440d	P6_IOC	P6 Port I/O Control Register	Selects the P6 port I/O direction.

Peripheral	Address	Register name		Function
GPIO (8-bit device)	0x440e	P7_DAT	P7 Port Input/Output Data Register	P7 port input/output data
	0x440f	P7_IOC	P7 Port I/O Control Register	Selects the P7 port I/O direction.
	0x4410	P8_DAT	P8 Port Input/Output Data Register	P8 port input/output data
	0x4411	P8_IOC	P8 Port I/O Control Register	Selects the P8 port I/O direction.
	0x4412	P9_DAT	P9 Port Input/Output Data Register	P9 port input/output data
	0x4413	P9_IOC	P9 Port I/O Control Register	Selects the P9 port I/O direction.
	0x4414	PA_DAT	PA Port Input/Output Data Register	PA port input/output data
	0x4415	PA_IOC	PA Port I/O Control Register	Selects the PA port I/O direction.
	0x4416	PB_DAT	PB Port Input/Output Data Register	PB port input/output data
	0x4417	PB_IOC	PB Port I/O Control Register	Selects the PB port I/O direction.
	0x4418	PC_DAT	PC Port Input/Output Data Register	PC port input/output data
	0x4419	PC_IOC	PC Port I/O Control Register	Selects the PC port I/O direction.
	0x441a–0x441f	–	–	Reserved
	0x4420	P0_03_CFP	P00–P03 Port Function Select Register	Selects the P00–P03 port functions.
	0x4421	P0_47_CFP	P04–P07 Port Function Select Register	Selects the P04–P07 port functions.
	0x4422	P1_03_CFP	P10–P13 Port Function Select Register	Selects the P10–P13 port functions.
	0x4423	P1_46_CFP	P14–P16 Port Function Select Register	Selects the P14–P16 port functions.
	0x4424	P2_03_CFP	P20–P23 Port Function Select Register	Selects the P20–P23 port functions.
	0x4425	P2_47_CFP	P24–P27 Port Function Select Register	Selects the P24–P27 port functions.
	0x4426	P3_03_CFP	P30–P33 Port Function Select Register	Selects the P30–P33 port functions.
	0x4427	P3_47_CFP	P34–P37 Port Function Select Register	Selects the P34–P37 port functions.
	0x4428	P4_03_CFP	P40–P43 Port Function Select Register	Selects the P40–P43 port functions.
	0x4429	P4_45_CFP	P44–P45 Port Function Select Register	Selects the P44–P45 port functions.
	0x442a	P5_03_CFP	P50–P53 Port Function Select Register	Selects the P50–P53 port functions.
	0x442b	P5_47_CFP	P54–P57 Port Function Select Register	Selects the P54–P57 port functions.
	0x442c	P6_03_CFP	P60–P63 Port Function Select Register	Selects the P60–P63 port functions.
	0x442d	P6_47_CFP	P64–P67 Port Function Select Register	Selects the P64–P67 port functions.
	0x442e	P7_03_CFP	P70–P73 Port Function Select Register	Selects the P70–P73 port functions.
	0x442f	P7_47_CFP	P74–P77 Port Function Select Register	Selects the P74–P77 port functions.
	0x4430	P8_03_CFP	P80–P83 Port Function Select Register	Selects the P80–P83 port functions.
	0x4431	P8_46_CFP	P84–P86 Port Function Select Register	Selects the P84–P86 port functions.
	0x4432	P9_03_CFP	P90–P93 Port Function Select Register	Selects the P90–P93 port functions.
	0x4433	P9_47_CFP	P94–P97 Port Function Select Register	Selects the P94–P97 port functions.
	0x4434	PA_03_CFP	PA0–PA3 Port Function Select Register	Selects the PA0–PA3 port functions.
	0x4435	PA_46_CFP	PA4–PA6 Port Function Select Register	Selects the PA4–PA6 port functions.
	0x4436	PB_03_CFP	PB0–PB3 Port Function Select Register	Selects the PB0–PB3 port functions.
	0x4437	PB_47_CFP	PB4–PB7 Port Function Select Register	Selects the PB4–PB7 port functions.
	0x4438	PC_03_CFP	PC0–PC3 Port Function Select Register	Selects the PC0–PC3 port functions.
	0x4439	PC_47_CFP	PC4–PC7 Port Function Select Register	Selects the PC4–PC7 port functions.
	0x443a–0x443f	–	–	Reserved
	0x4440	PINTSEL0	Port Input Interrupt 0 Select Register	Selects a P*x*0 port for input interrupt.
	0x4441	PINTSEL1	Port Input Interrupt 1 Select Register	Selects a P*x*1 port for input interrupt.
	0x4442	PINTSEL2	Port Input Interrupt 2 Select Register	Selects a P*x*2 port for input interrupt.
	0x4443	PINTSEL3	Port Input Interrupt 3 Select Register	Selects a P*x*3 port for input interrupt.
	0x4444	PINTSEL4	Port Input Interrupt 4 Select Register	Selects a P*x*4 port for input interrupt.
	0x4445	PINTSEL5	Port Input Interrupt 5 Select Register	Selects a P*x*5 port for input interrupt.
	0x4446	PINTSEL6	Port Input Interrupt 6 Select Register	Selects a P*x*6 port for input interrupt.
	0x4447	PINTSEL7	Port Input Interrupt 7 Select Register	Selects a P*x*7 port for input interrupt.
	0x4448–0x44ff	–	–	Reserved
USB function controller (8-bit device)	0x4500	MainIntStat	Main Interrupt Status Register	Indicates main interrupt status.
	0x4501	SIE_IntStat	SIE Interrupt Status Register	Indicates SIE interrupt status.
	0x4502	EPrIntStat	EPr Interrupt Status Register	Indicates EPr interrupt status.
	0x4503	–	–	Reserved
	0x4504	FIFO_IntStat	FIFO Interrupt Status Register	Indicates FIFO interrupt status.
	0x4505–0x4506	–	–	Reserved
	0x4507	EP0IntStat	EP0 Interrupt Status Register	Indicates EP0 interrupt status.
	0x4508	EPaIntStat	EPa Interrupt Status Register	Indicates EPa interrupt status.
	0x4509	EPbIntStat	EPb Interrupt Status Register	Indicates EPb interrupt status.
	0x450a	EPcIntStat	EPc Interrupt Status Register	Indicates EPc interrupt status.
	0x450b	EPdIntStat	EPd Interrupt Status Register	Indicates EPd interrupt status.
	0x450c–0x450f	–	–	Reserved
	0x4510	MainIntEnb	Main Interrupt Enable Register	Enables main interrupts.
	0x4511	SIE_IntEnb	SIE Interrupt Enable Register	Enables SIE interrupts.
	0x4512	EPrIntEnb	EPr Interrupt Enable Register	Enables EPr interrupts.
	0x4513	–	–	Reserved
	0x4514	FIFOIntEnb	FIFO Interrupt Enable	Enables FIFO interrupts.
	0x4515–0x4516	–	–	Reserved
	0x4517	EP0IntEnb	EP0 Interrupt Enable Register	Enables EP0 interrupts.

Peripheral	Address	Register name		Function
USB function controller (8-bit device)	0x4518	EPaIntEnb	EPa Interrupt Enable Register	Enables EPa interrupts.
	0x4519	EPbIntEnb	EPb Interrupt Enable Register	Enables EPb interrupts.
	0x451a	EPcIntEnb	EPc Interrupt Enable Register	Enables EPc interrupts.
	0x451b	EPdIntEnb	EPd Interrupt Enable Register	Enables EPd interrupts.
	0x451c–0x451f	–	–	Reserved
	0x4520	RevisionNum	Revision Number Register	Indicates the USB controller revision number.
	0x4521	USB_Control	USB Control Register	Controls USB conditions.
	0x4522	USB_Status	USB Status Register	Indicates the USB status.
	0x4523	XcvrControl	Xcvr Control Register	Controls the transceiver macro.
	0x4524	USB_Test	USB Test Register	Sets up the USB test mode.
	0x4525	EPnControl	Endpoint Control Register	Clears all FIFOs and sets NAK/STALL bits.
	0x4526	EPrFIFO_Clr	EPr FIFO Clear Register	Clears each FIFO.
	0x4527–0x452d	–	–	Reserved
	0x452e	FrameNumber_H	Frame Number High Register	Frame number
	0x452f	FrameNumber_L	Frame Number Low Register	
	0x4530	EP0Setup_0	EP0 Setup 0 Register	EP0 setup data (BmRequestType)
	0x4531	EP0Setup_1	EP0 Setup 1 Register	EP0 setup data (BRequest)
	0x4532	EP0Setup_2	EP0 Setup 2 Register	EP0 setup data (low-order Wvalue bits)
	0x4533	EP0Setup_3	EP0 Setup 3 Register	EP0 setup data (high-order Wvalue bits)
	0x4534	EP0Setup_4	EP0 Setup 4 Register	EP0 setup data (low-order WIndex bits)
	0x4535	EP0Setup_5	EP0 Setup 5 Register	EP0 setup data (high-order WIndex bits)
	0x4536	EP0Setup_6	EP0 Setup 6 Register	EP0 setup data (low-order WLength bits)
	0x4537	EP0Setup_7	EP0 Setup 7 Register	EP0 setup data (high-order WLength bits)
	0x4538	USB_Address	USB Address Register	Sets a USB address.
	0x4539	EP0Control	EP0 Control Register	Sets up EP0.
	0x453a	EP0ControlIN	EP0 Control In Register	Sets EP0 IN transaction conditions.
	0x453b	EP0ControlOUT	EP0 Control Out Register	Sets EP0 OUT transaction conditions.
	0x453c–0x453e	–	–	Reserved
	0x453f	EP0MaxSize	EP0 Max Packet Size Register	Sets the EP0 max packet size.
	0x4540	EPaControl	EPa Control Register	Sets up EPa.
	0x4541	EPbControl	EPb Control Register	Sets up EPb.
	0x4542	EPcControl	EPc Control Register	Sets up EPc.
	0x4543	EPdControl	EPd Control Register	Sets up EPd.
	0x4544–0x454f	–	–	Reserved
	0x4550	EPaMaxSize_H	EPa Max Packet Size High Register	Sets the EPa max packet size.
	0x4551	EPaMaxSize_L	EPa Max Packet Size Low Register	
	0x4552	EPaConfig_0	EPa Configuration 0 Register	Configures EPa.
	0x4553	EPaConfig_1	EPa Configuration 1 Register	
	0x4554	EPbMaxSize_H	EPb Max Packet Size High Register	Sets the EPb max packet size.
	0x4555	EPbMaxSize_L	EPb Max Packet Size Low Register	
	0x4556	EPbConfig_0	EPb Configuration 0 Register	Configures EPb.
	0x4557	EPbConfig_1	EPb Configuration 1 Register	
	0x4558	EPcMaxSize_H	EPc Max Packet Size High Register	Sets the EPc max packet size.
	0x4559	EPcMaxSize_L	EPc Max Packet Size Low Register	
	0x455a	EPcConfig_0	EPc Configuration 0 Register	Configures EPc.
	0x455b	EPcConfig_1	EPc Configuration 1 Register	
	0x455c	EPdMaxSize_H	EPd Max Packet Size High Register	Sets the EPd max packet size.
	0x455d	EPdMaxSize_L	EPd Max Packet Size Low Register	
	0x455e	EPdConfig_0	EPd Configuration 0 Register	Configures EPd.
	0x455f	EPdConfig_1	EPd Configuration 1 Register	
	0x4560–0x456f	–	–	Reserved
	0x4570	EPaStartAdrs_H	EPa FIFO Start Address High Register	Sets the FIFO start address for EPa.
	0x4571	EPaStartAdrs_L	EPa FIFO Start Address Low Register	
	0x4572	EPbStartAdrs_H	EPb FIFO Start Address High Register	Sets the FIFO start address for EPb.
	0x4573	EPbStartAdrs_L	EPb FIFO Start Address Low Register	
	0x4574	EPcStartAdrs_H	EPc FIFO Start Address High Register	Sets the FIFO start address for EPc.
	0x4575	EPcStartAdrs_L	EPc FIFO Start Address Low Register	
	0x4576	EPdStartAdrs_H	Epd FIFO Start Address High Register	Sets the FIFO start address for EPd.
	0x4577	EPdStartAdrs_L	EPd FIFO Start Address Low Register	
	0x4578–0x457f	–	–	Reserved
	0x4580	CPU_JoinRd	CPU Join FIFO Read Register	Sets up the FIFO data read conditions.
	0x4581	CPU_JoinWr	CPU Join FIFO Write Register	Sets up the FIFO data write conditions.
	0x4582	EnEPnFIFO _Access	EPn FIFO Access Enable Register	Enables the CPU_JoinRd and CPU_JoinWr registers.
	0x4583	EPnFIFOforCPU	EPn FIFO for CPU Register	EPn FIFO for accessing by the CPU.

Peripheral	Address	Register name		Function
USB function controller (8-bit device)	0x4584	EPnRdRemain_H	EPn FIFO Read Remain High Register	Indicates the remained data quantity in the FIFO.
	0x4585	EPnRdRemain_L	EPn FIFO Read Remain Low Register	
	0x4586	EPnWrRemain_H	EPn FIFO Write High Register	Indicates the free space capacity in the FIFO.
	0x4587	EPnWrRemain_L	EPn FIFO Write Low Register	
	0x4588	DescAdrs_H	Descriptor Address High Register	Specifies the FIFO start address for the descriptor reply function.
	0x4589	DescAdrs_L	Descriptor Address Low Register	
	0x458a	DescSize_H	Descriptor Size High Register	Specifies the number of data for the descriptor reply function.
	0x458b	DescSize_L	Descriptor Size Low Register	
	0x458c–0x458e	–	–	Reserved
	0x458f	DescDoor	Descriptor Door Register	Reads/writes descriptors.
	0x4590–0x45ff	–	–	Reserved
Real-time clock (8-bit device)	0x4600	RTC_INTSTAT	RTC Interrupt Status Register	Indicates RTC interrupt status.
	0x4601	RTC_INTMODE	RTC Interrupt Mode Register	Sets up RTC interrupt modes.
	0x4602	RTC_CNTL0	RTC Control 0 Register	Controls the RTC.
	0x4603	RTC_CNTL1	RTC Control 1 Register	
	0x460f	RTC_WAKEUP	RTC Wakeup Configuration Register	Sets up RTC wakeup conditions.
	0x4614	RTC_SEC	RTC Second Register	Second counter data
	0x4615	RTC_MIN	RTC Minute Register	Minute counter data
	0x4616	RTC_HOUR	RTC Hour Register	Hour counter data
	0x4617	RTC_DAY	RTC Day Register	Day counter data
	0x4628	RTC_MONTH	RTC Month Register	Month counter data
	0x4629	RTC_YEAR	RTC Year Register	Year counter data
	0x462a	RTC_WEEK	RTC Days of Week Register	Days of week counter data
	0x462b–0x46ff	–	–	Reserved
8-bit programmable timer CH.0 (8-bit device)	0x4800	PT8_CLK0	PT8 CH.0 Input Clock Select Register	Selects the count clock.
	0x4801	PT8_RLD0	PT8 CH.0 Reload Data Register	Sets reload data.
	0x4802	PT8_PTD0	PT8 CH.0 Counter Data Register	Counter data
	0x4803	PT8_CTL0	PT8 CH.0 Control Register	Sets the timer mode and starts/stops the timer.
8-bit programmable timer CH.1 (8-bit device)	0x4804	PT8_CLK1	PT8 CH.1 Input Clock Select Register	Selects the count clock.
	0x4805	PT8_RLD1	PT8 CH.1 Reload Data Register	Sets reload data.
	0x4806	PT8_PTD1	PT8 CH.1 Counter Data Register	Counter data
	0x4807	PT8_CTL1	PT8 CH.1 Control Register	Sets the timer mode and starts/stops the timer.
8-bit programmable timer CH.2 (8-bit device)	0x4808	PT8_CLK2	PT8 CH.2 Input Clock Select Register	Selects the count clock.
	0x4809	PT8_RLD2	PT8 CH.2 Reload Data Register	Sets reload data.
	0x480a	PT8_PTD2	PT8 CH.2 Counter Data Register	Counter data
	0x480b	PT8_CTL2	PT8 CH.2 Control Register	Sets the timer mode and starts/stops the timer.
8-bit programmable timer CH.3 (8-bit device)	0x480c	PT8_CLK3	PT8 CH.3 Input Clock Select Register	Selects the count clock.
	0x480d	PT8_RLD3	PT8 CH.3 Reload Data Register	Sets reload data.
	0x480e	PT8_PTD3	PT8 CH.3 Counter Data Register	Counter data
	0x480f	PT8_CTL3	PT8 CH.3 Control Register	Sets the timer mode and starts/stops the timer.
	0x4810–0x48ff	–	–	Reserved
Clock management unit (8-bit device)	0x4900	CMU_SYSCLKCTL	System Clock Control Register	Controls the system clock.
	0x4901	CMU_OSC3_WCNT	OSC3 Wait Timer Register	Sets the OSC3 wait timer for system wake-up.
	0x4902	CMU_NF	Noise Filter Control Register	Enables noise filters.
	0x4903	CMU_OSC3DIV	OSC3 Clock Divider Register	Selects a OSC3 system clock frequency.
	0x4904	–	–	Reserved
	0x4905	CMU_CMUCLK	CMU_CLK Select Register	Selects the output CMU_CLK frequency.
	0x4906	CMU_GATEDCLK0	Gated Clock Control 0 Register	Controls clock supply to peripheral modules.
	0x4907	CMU_GATEDCLK1	Gated Clock Control 1 Register	
	0x4908	CMU_GATEDCLK2	Gated Clock Control 2 Register	
	0x4909	CMU_USBWT	USB Wait Control Register	Sets the wait cycles for accessing the USB registers.
	0x490a–0x491f	–	–	Reserved
	0x4920	CMU_PROTECT	CMU Write Protect Register	Enables writing to the CMU registers (0x4900–0x4909).
	0x4921–0x49ff	–	–	Reserved
SRAM controller (16-bit device)	0x5000	EXTMEM_SWAIT	External Memory Static Wait Control Register	Sets up static wait cycles.
	0x5002	–	–	Reserved
	0x5004	EXTMEM_SIZE	External Memory Device Size Setup Register	Selects the device size (8/16 bits).
	0x5006	–	–	Reserved
	0x5008	EXTMEM_A0_BSL	External Memory Device Type Setup Register	Selects the device type (A0/BSL).
	0x500a-0x500f	–	–	Reserved
	0x5010	IRAM2_SWAIT	Internal Memory Static Wait Control Register	Sets up IRAM2 read cycle.
	0x5012	–	–	Reserved
	0x5014	MAC_WAIT	MAC Wait Control Register	Sets up the MAC wait cycle.
	0x5016	–	–	Reserved
	0x5018	RTC_WAIT	RTC Wait Control Register	Sets up RTC access cycle.
	0x501a–0x50ff	–	–	Reserved

Peripheral	Address	Register name		Function
Multi-function timer (16-bit device)	0x5200	MFT_TC	MFT Counter Data Register	Counter data
	0x5202	MFT_PRD	MFT Period Data Register	Sets period data.
	0x5204	MFT_CMP	MFT Compare Data Register	Sets compare data.
	0x5206	MFT_CTL	MFT Control Register	Sets the timer mode and starts/stops the timer.
	0x5208–0x521d	–	–	Reserved
	0x521e	MFT_IOCTL	MFT Input/Output Control Register	Controls the clock input/output.
	0x5230	MFT_IE	MFT Interrupt Enable Register	Enables the MFT interrupt.
	0x5238	MFT_IF	MFT Interrupt Flag Register	Indicates the MFT interrupt status.
	0x523a–0x527d	–	–	Reserved
	0x527e	MFT_TST	MFT Test Register	Controls the MFT test.
	0x5280–0x52ff	–	–	Reserved
I²S (16-bit device)	0x5300	I2S_CTL_OUT	I²S CH.0 Control Register	Sets output conditions.
	0x5302	I2S_CTL_IN	I²S CH.1 Control Register	Sets input conditions.
	0x5304	I2S_DV_MCLK	I²S MCLK Divide Ratio Register	Configures MCLK.
	0x5306	I2S_DV_AUDIO_CLK	I²S Audio Clock Divide Ratio Register	Configures the audio clock.
	0x5308	I2S_START	I²S Start/Stop Register	Controls/indicates I²S start/stop status.
	0x530a	I2S_FIFO_STAT	I²S FIFO Status Register	Indicates the FIFO status.
	0x530c	I2S_INT_MOD	I²S Interrupt Mode Select Register	Sets the I²S interrupt conditions.
	0x5310	I2S_FIFO_OUT	I²S CH.0 FIFO Register	L & R channel output data (16- or 24-bit access)
	0x5314	I2S_FIFO_IN	I²S CH.1 FIFO Register	L & R channel input data (16-bit access)
	0x5318–0x53ff	–	–	Reserved
Remote controller (16-bit device)	0x5400	REMC_PSC	REMC Prescaler Control Register	Sets up the REMC prescaler.
	0x5404	REMC_CFG	REMC Configuration Register	Sets the REMC modes and controls the REMC interrupt.
	0x5406	–	–	Reserved
	0x5408	REMC_CTL	REMC Control Register	Starts/stops transmission.
	0x540a	–	–	Reserved
	0x540c	REMC_CARL	REMC Carrier Load Register	Configures the carrier signal.
	0x540e	REMC_ENVL	REMC Envelope Load Register	Configures the envelope pulse width.
	0x5410	REMC_ENVC	REMC Envelope Capture Register	Input envelope pulse width
	0x5412–0x54ff	–	–	Reserved
A/D converter (16-bit device)	0x5500–0x551f	–	–	Reserved
	0x5520	AD_CLKCTL	A/D Clock Control Register	Controls A/D converter clock.
	0x5522–0x553f	–	–	Reserved
	0x5540	AD_DAT	A/D Conversion Result Register	A/D converted data
	0x5542	AD_TRIG_CH	A/D Trigger/Channel Select Register	Sets start/end channels and conversion mode.
	0x5544	AD_CTL	A/D Control/Status Register	Controls A/D converter and indicates conversion status.
	0x5546	AD_CH_STAT	A/D Channel Status Flag Register	Indicates overwrite error and conversion complete status.
	0x5548	AD_CH0_BUF	A/D CH.0 Conversion Result Buffer Register	A/D CH.0 converted data
	0x554a	AD_CH1_BUF	A/D CH.1 Conversion Result Buffer Register	A/D CH.1 converted data
	0x554c	AD_CH2_BUF	A/D CH.2 Conversion Result Buffer Register	A/D CH.2 converted data
	0x554e	AD_CH3_BUF	A/D CH.3 Conversion Result Buffer Register	A/D CH.3 converted data
	0x5550	AD_CH4_BUF	A/D CH.4 Conversion Result Buffer Register	A/D CH.4 converted data
	0x5552	AD_CH5_BUF	A/D CH.5 Conversion Result Buffer Register	A/D CH.5 converted data
	0x5554	AD_CH6_BUF	A/D CH.6 Conversion Result Buffer Register	A/D CH.6 converted data
	0x5556	AD_CH7_BUF	A/D CH.7 Conversion Result Buffer Register	A/D CH.7 converted data
	0x5558	AD_UPPER	A/D Upper Limit Value Register	Specifies A/D conversion upper limit value.
	0x555a	AD_LOWER	A/D Lower Limit Value Register	Specifies A/D conversion lower limit value.
	0x555c	AD_INTMASK	A/D Conversion Complete Interrupt Mask Register	Masks A/D conversion complete interrupt.
	0x555e	AD_ADVMODE	A/D Converter Mode Select/Internal Status Register	Selects A/D operating mode and indicates internal status and internal counter value.
Watchdog timer (16-bit device)	0x5600–0x565f	–	–	Reserved
	0x5660	WD_WP	WDT Write Protect Register	Enables WDT control registers for writing.
	0x5662	WD_EN	WDT Enable and Setup Register	Configures and starts watchdog timer.
	0x5664	WD_CMP_L	WDT Comparison Data L Register	Comparison data
	0x5666	WD_CMP_H	WDT Comparison Data H Register	
	0x5668	WD_CNT_L	WDT Count Data L Register	Watchdog timer counter data
	0x566a	WD_CNT_H	WDT Count Data H Register	
	0x566c	WD_CTL	WDT Control Register	Resets watchdog timer.
	0x566e–0x56ff	–	–	Reserved

Peripheral	Address	Register name		Function
Extended SPI (16-bit device)	0x5700	SPI_ST1	SPI CH.1 Status Register	Indicates transfer and buffer statuses.
	0x5702	SPI_TXD1	SPI CH.1 Transmit Data Register	Transmit data
	0x5704	SPI_RXD1	SPI CH.1 Receive Data Register	Receive data
	0x5706	SPI_CTL1	SPI CH.1 Control Register	Sets the SPI CH.1 mode and enables data transfer.
	0x5708	SPI_CLK1	SPI CH.1 Clock Control Register	Sets up the SPI clock.
	0x570a–0x57ff	–	–	Reserved
Flash controller (16-bit device)	0x5800	FLASH_CTL	FLASHC Control Register	Controls Flash erase/program operations.
	0x5802	FLASH_ADDR	FLASHC Sector Address Register	Sets the Flash address for erasing a sector.
	0x5804	FLASH_WAIT	FLASHC Wait Register	Sets the wait cycle for Flash read.
	0x5806–0x580f	–	–	Reserved
	0x5810	FLASH_PROT	FLASHC Protect Register	Enables Flash control registers for writing.
	0x5812	–	–	Reserved
	0x5814	TTBR_LOW	Trap Table Base Register 0	Sets the vector table address.
	0x5816	TTBR_HIGH	Trap Table Base Register 1	
	0x5818–0x58ff	–	–	Reserved

Note: Do not access the “Reserved” address in the table above and unused areas in the peripheral area that are not described in the table from the application program.

I.6.6 S1C17 Core I/O Area

The 1K-byte area from address 0xfffc00 to address 0xffffff is the I/O area for the CPU core in which the I/O registers listed in the table below are located.

Table I.6.6.1 I/O Map (S1C17 Core I/O Area)

Peripheral	Address	Register name		Function
S1C17 Core I/O	0xffff90	DBRAM	Debug RAM Base Register	Indicates the debug RAM base address.

See Section I.5.5.2, "Work Area for Debugging," for DBRAM.

I.7 Electrical Characteristics

I.7.1 Absolute Maximum Rating

(V_{SS} = 0V)

Item	Symbol	Condition	Rated value	Unit
Power supply voltage	V_{DD}		-0.3 to 4.0	V
RTC power supply voltage	$RTCV_{DD}$		-0.3 to 4.0	V
Input voltage	V_I		-0.3 to V_{DD} + 0.3	V
Output voltage	V_O		-0.3 to V_{DD} + 0.3	V
High level output current	I_{OH}	1 pin	-10	mA
		Total of all pins	-40	mA
Low level output current	I_{OL}	1 pin	10	mA
		Total of all pins	40	mA
Analog power supply voltage	AV_{DD}		-0.3 to 7.0	V
Analog input voltage	AV_{IN}		-0.3 to AV_{DD} + 0.3	V
Operating temperature	Topr	During Flash reading	-40 to 85	°C
		During Flash erasing/ programming	-40 to 70	°C
		During USB operation	0 to 70	°C
Storage temperature	Tstg		-65 to 150	°C
Soldering temperature/time	Tsol		260°C, 10 seconds (lead section)	–

I.7.2 Recommended Operating Conditions

(Ta = -40 to 85°C)

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Core and I/O voltage	V_{DD}		3.0	3.3	3.6	V
Analog (I/O) voltage	AV_{DD}		2.7		5.5	V
RTC power supply voltage	$RTCV_{DD}$		3.0	3.3	3.6	V
Operating frequency	f_{OSC3}	Crystal/ceramic oscillation	1		48	MHz
		External clock input			48	MHz
	f_{OSC1}	Crystal oscillation		32.768		kHz
		External clock input		32.768		kHz
Input voltage	V_I		V_{SS}		V_{DD}	V
Input rise time (normal input)	t_{ri}				50	ns
Input fall time (normal input)	t_{fi}				50	ns
Input rise time (Schmitt input)	t_{ri}				5	ms
Input fall time (Schmitt input)	t_{fi}				5	ms

I.7.3 DC Characteristics

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Input leakage current	I_{LI}		-1		1	µA
Off-state leakage current	I_{OZ}		-1		1	µA
High level output current	I_{OH}	I_{OH} = -2mA (Type 1), I_{OH} = -12mA (Type 3), V_{DD} = Min.	V_{DD} - 0.4			V
Low level output current	I_{OL}	I_{OL} = 2mA (Type 1), I_{OL} = 12mA (Type 3), V_{DD} = Min.			0.4	V
High level input voltage	V_{IH}	CMOS level, V_{DD} = Max.	2.4			V
Low level input voltage	V_{IL}	CMOS level, V_{DD} = Min.			0.4	V
Positive trigger input voltage	V_{T1+}	LVTTL Schmitt	1.1		2.4	V
Negative trigger input voltage	V_{T1-}	LVTTL Schmitt	0.6		1.8	V
Hysteresis voltage	V_H	LVTTL Schmitt	0.1			V
Pull-up resistor	R_{PU}	100kΩ type, VI = 0V	50	100	288	kΩ
		50kΩ type, VI = 0V	25	50	144	kΩ
Input pin capacitance	C_I	f = 1MHz, V_{DD} = 0V			10	pF
Output pin capacitance	C_O	f = 1MHz, V_{DD} = 0V			10	pF
I/O pin capacitance	C_{IO}	f = 1MHz, V_{DD} = 0V			10	pF

Note: See Section I.3.4, “Input/Output Cells and Input/Output Characteristics,” for pin characteristics.

I.7.4 Current Consumption

Unless otherwise specified: V_{DD} = 3.3V, AV_{DD} = 3.3V, $RTCV_{DD}$ = 3.3V, V_{SS} = 0V, Ta = 25°C, Peripheral modules: stopped

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Battery backup current	I_{BB1}	OSC1: Off ∗4, RTC: Stop, V_{DD}/AV_{DD}: Off		0.085		µA
	I_{BB2}	OSC1: 32kHz, RTC: Run, V_{DD}/AV_{DD}: Off		3.7		µA
Current consumption in SLEEP mode	I_{SLP1}	OSC1: Off ∗4, OSC3: Off, RTC: Stop		1.4		µA
	I_{SLP2}	OSC1: 32kHz, OSC3: Off, RTC: Run		6		µA
Current consumption in HALT mode (IRAM operation) ∗1	I_{HALT11}	OSC1: 32kHz, OSC3: Off, RTC: Run		12		µA
	I_{HALT12}	OSC1: 32kHz, OSC3: 1MHz, RTC: Run		0.34		mA
	I_{HALT13}	OSC1: 32kHz, OSC3: 4MHz, RTC: Run		1.3		mA
	I_{HALT14}	OSC1: 32kHz, OSC3: 8MHz, RTC: Run		2.7		mA
	I_{HALT15}	OSC1: 32kHz, OSC3: 20MHz, RTC: Run		6.6		mA
	I_{HALT16}	OSC1: 32kHz, OSC3: 25MHz, RTC: Run		8.3		mA
	I_{HALT17}	OSC1: 32kHz, OSC3: 33MHz, RTC: Run		11		mA
	I_{HALT18}	OSC1: 32kHz, OSC3: 48MHz, RTC: Run		16		mA
Current consumption during execution (IRAM operation) ∗2	I_{EXE11}	OSC1: 32kHz, OSC3: Off, RTC: Run		29		µA
	I_{EXE12}	OSC1: 32kHz, OSC3: 1MHz, RTC: Run		0.80		mA
	I_{EXE13}	OSC1: 32kHz, OSC3: 4MHz, RTC: Run		3.2		mA
	I_{EXE14}	OSC1: 32kHz, OSC3: 8MHz, RTC: Run		6.4		mA
	I_{EXE15}	OSC1: 32kHz, OSC3: 20MHz, RTC: Run		16		mA
	I_{EXE16}	OSC1: 32kHz, OSC3: 25MHz, RTC: Run		20		mA
	I_{EXE17}	OSC1: 32kHz, OSC3: 33MHz, RTC: Run		26		mA
	I_{EXE18}	OSC1: 32kHz, OSC3: 48MHz, RTC: Run		38		mA
Current consumption during execution (Flash operation) ∗3	I_{EXE21}	OSC1: 32kHz, OSC3: Off, RTC: Run		5.1		mA
	I_{EXE22}	OSC1: 32kHz, OSC3: 1MHz, RTC: Run		6.0		mA
	I_{EXE23}	OSC1: 32kHz, OSC3: 4MHz, RTC: Run		8.5		mA
	I_{EXE24}	OSC1: 32kHz, OSC3: 8MHz, RTC: Run		12		mA
	I_{EXE25}	OSC1: 32kHz, OSC3: 20MHz, RTC: Run		22		mA
	I_{EXE26}	OSC1: 32kHz, OSC3: 25MHz, RTC: Run		24		mA
	I_{EXE27}	OSC1: 32kHz, OSC3: 33MHz, RTC: Run		29		mA
	I_{EXE28}	OSC1: 32kHz, OSC3: 48MHz, RTC: Run		37		mA
Current consumption including USB operating current	I_{USB}	USB_SAPB_CLK enabled, USB_CLK enabled		38		mA
Current consumption including ADC operating current	I_{ADC1}	When the ADC is enabled and standby, System clock: OSC1, OSC1: 32kHz, OSC3: 48MHz, RTC: Stop, Other peripheral circuits: Stop (0x4906 = 0x00, 0x4907 = 0x00, 0x4908 = 0x08), Only ADC operated, Conversion clock frequency: 2MHz		17		mA
	I_{ADC2}	When the ADC is converting, System clock: OSC1, OSC1: 32kHz, OSC3: 48MHz, RTC: Stop, Other peripheral circuits: Stop (0x4906 = 0x00, 0x4907 = 0x00, 0x4908 = 0x08), Only ADC operated, Conversion clock frequency: 2MHz		18		mA
Current consumption including Flash chip erase current	I_{FCERS}			67		mA
Current consumption including Flash sector erase current	I_{FSERS}			69		mA
Current consumption including Flash programming current	I_{FPRG}			20		mA

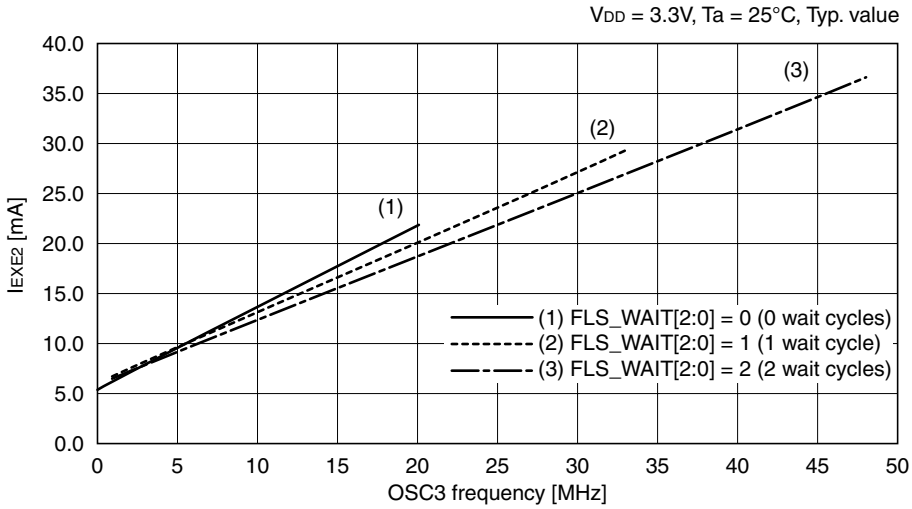
∗1) When the halt instruction is executed on the IRAM

∗2) When the program is executed on the IRAM

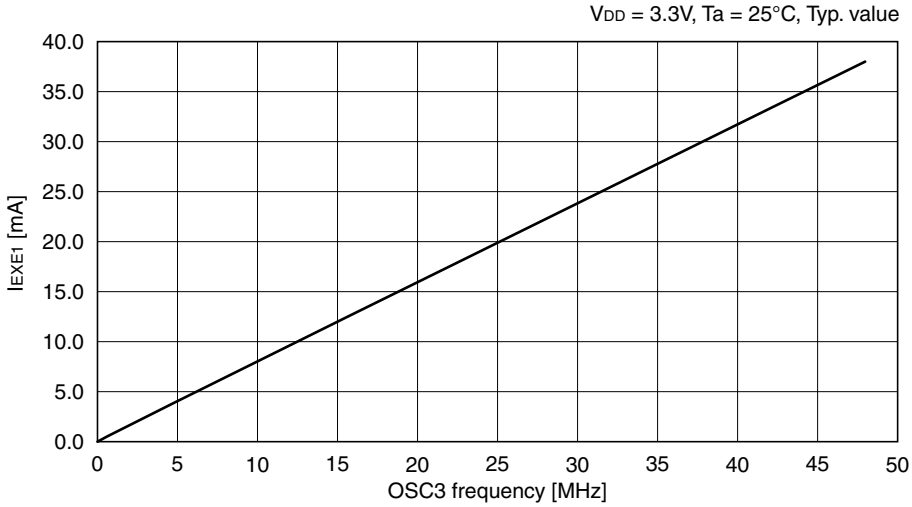
∗3) When the program is executed on the Flash
∗4) When no resonator is connected (The OSC1 oscillator circuit cannot be turned off.)

Notes:
- When the halt instruction located in the Flash memory is executed, the S1C17501 enters HALT mode with the Flash area chip select signal asserted. This increases current consumption, as the Flash memory stays active during HALT status. Therefore, when setting the S1C17501 into HALT mode, the halt instruction should be executed in IRAM.
- The current consumption during program execution in IRAM with a high-speed clock is larger than that of the Flash. This is because the program execution speed in IRAM is higher than that in the Flash.
- The current consumption during execution in the above table indicates the value when a test program that consists of 51% load instructions, 21% arithmetic operation instructions, 10% branch instructions and 18% ext instructions is being executed in the built-in memory (IRAM or Flash).

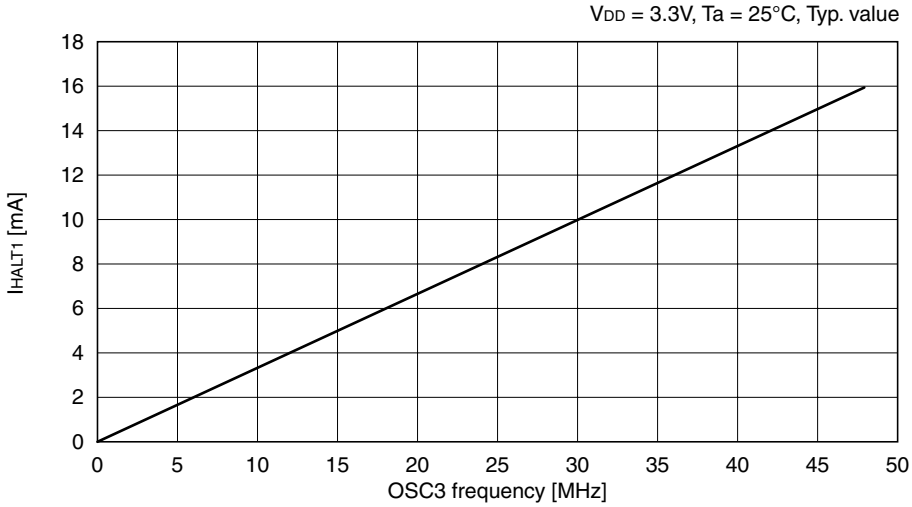
Current consumption - frequency characteristic during execution (Flash operation)



Current consumption - frequency characteristic during execution (IRAM operation)



Current consumption - frequency characteristic in HALT mode (when the halt instruction is executed in IRAM)



I.7.5 A/D Converter Characteristics

Unless otherwise specified: V_{DD} = AV_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C, ST[1:0] = 11

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Resolution	–			10		bit
Conversion time *1	–		10		1250	µs
Zero scale error	E_{ZS}		-2		2	LSB
Full scale error	E_{FS}		-2		2	LSB
Integral linearity error	E_L		-3		3	LSB
Differential linearity error	E_D		-3		3	LSB
Permissible signal source impedance	–				5	kΩ
Analog input capacitance	–				45	pF

*1) Indicates the minimum value when A/D clock = 2MHz. Indicates the maximum value when A/D clock = 16kHz.

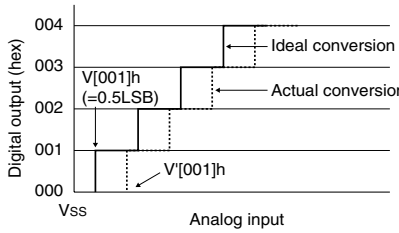
A/D conversion error

V[001]h = Ideal voltage at zero-scale point (=0.5LSB)
V'[001]h = Actual voltage at zero-scale point
V[3FF]h = Ideal voltage at full-scale point (=1022.5LSB)
V'[3FF]h = Actual voltage at full-scale point

$$1LSB = \frac{AV_{DD} - V_{SS}}{2^{10} - 1}$$

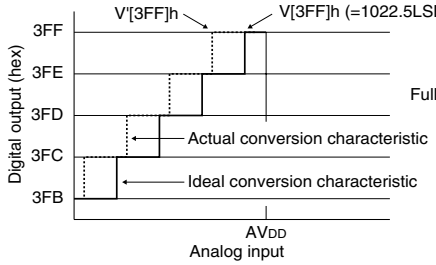
$$1LSB' = \frac{V'[3FF]h - V'[001]h}{2^{10} - 2}$$

■ **Zero scale error**



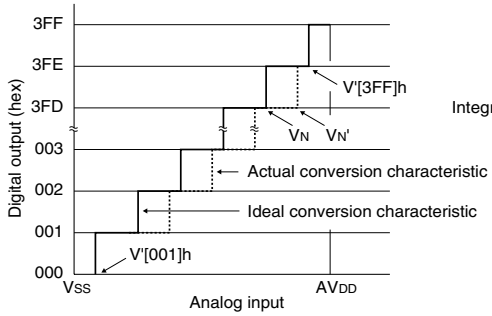
$$\text{Zero scale error } E_{ZS} = \frac{(V'[001]h - 0.5LSB') - (V[001]h - 0.5LSB)}{1LSB} \text{ [LSB]}$$

■ **Full scale error**



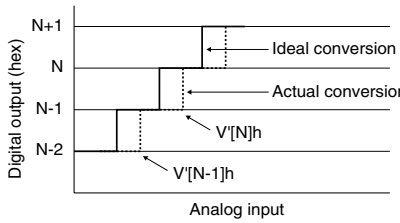
$$\text{Full scale error } E_{FS} = \frac{(V'[3FF]h + 0.5LSB') - (V[3FF]h + 0.5LSB)}{1LSB} \text{ [LSB]}$$

■ **Integral linearity error**



$$\text{Integral linearity error } E_L = \frac{V_{N'} - V_N}{1LSB'} \text{ [LSB]}$$

■ **Differential linearity error**



$$\text{Differential linearity error } E_D = \frac{V'[N]h - V'[N-1]h}{1LSB'} - 1 \text{ [LSB]}$$

I.7.6 Oscillation Characteristics

Oscillation characteristics change depending on conditions such as components used (resonator, R_f, R_d, C_G, C_D) and board pattern. Use the following characteristics as reference values. In particular, when a ceramic or crystal oscillator is used, evaluate the components adequately under real operating conditions by mounting them on the board before the external register (R_f, R_d) and capacitor (C_G, C_D) values are finally decided.

OSC1 crystal oscillation

Unless otherwise specified: V_{DD} = 3.3V, $RTCV_{DD}$ = 3.3V, V_{SS} = 0V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{STA1}	*1			3	s

OSC3 crystal oscillation

Note: A “crystal resonator that uses a fundamental” should be used for the OSC3 crystal oscillation circuit.

Unless otherwise specified: V_{DD} = 3.3V, V_{SS} = 0V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{STA3}	*1			10	ms

OSC3 ceramic oscillation

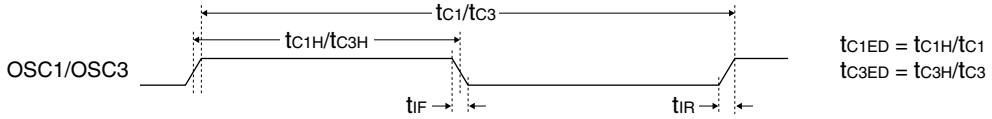
Unless otherwise specified: V_{SS} = 0V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{STA3}	*1			5	ms

*1) When the recommended parts shown in Section I.8, “Basic External Wiring Diagram,” are used

I.7.7 AC Characteristics

I.7.7.1 External Clock Input Characteristics



OSC1 external clock

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
OSC1 external clock cycle time	t_{C1}		30.51		µs
OSC1 external clock input duty	t_{C1ED}	45		55	%
OSC1 external clock input rise time	t_{IF}			5	ns
OSC1 external clock input fall time	t_{IR}			5	ns

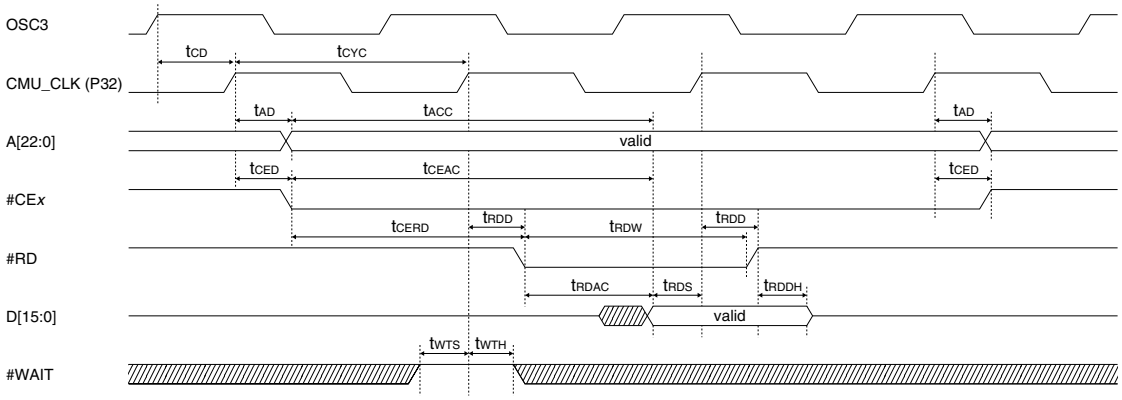
OSC3 external clock

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

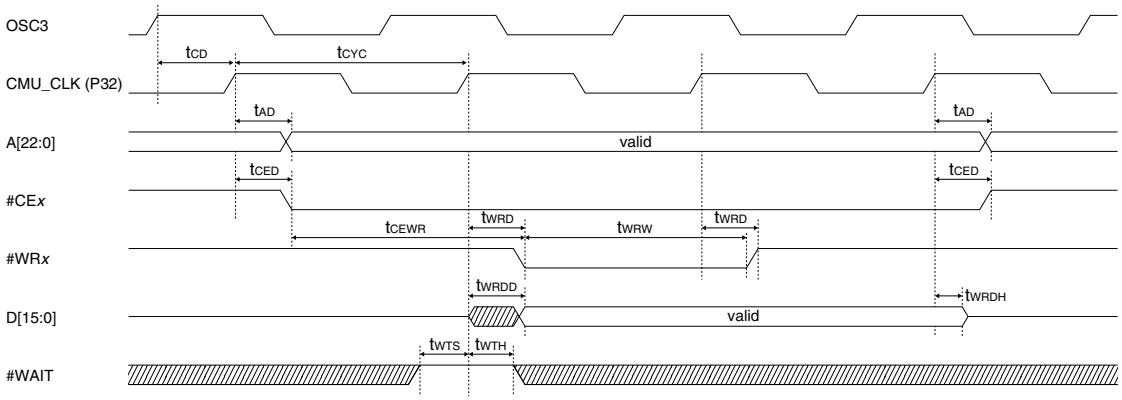
Item	Symbol	Min.	Typ.	Max.	Unit
OSC3 external clock cycle time	t_{C3}	20.83		1000	ns
OSC3 external clock input duty	t_{C3ED}	45		55	%
OSC3 external clock input rise time	t_{IF}			5	ns
OSC3 external clock input fall time	t_{IR}			5	ns

I.7.7.2 SRAMC AC Characteristics

SRAM read cycle



SRAM write cycle



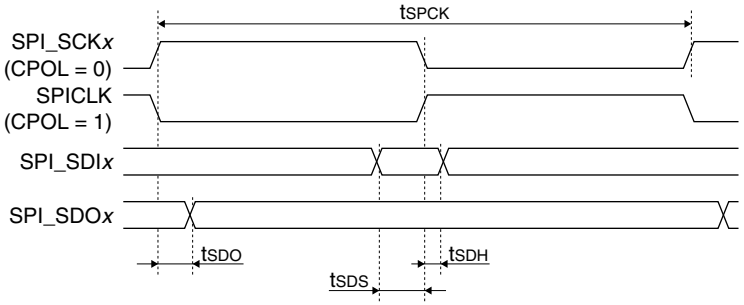
External load: #CE*x* pin = 20pF, other pins = 50pF

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
System clock period	t_{CYC}	20.833			ns
CMU_CLK output delay time	t_{CD}			24	ns
Address delay time	t_{AD}			12	ns
#CE delay time	t_{CED}			12	ns
Write delay time	t_{WRD}			9	ns
Write data delay time	t_{WRDD}			9	ns
Write data hold time	t_{WRDH}	-7			ns
#RD delay time from #CE active	t_{CERD}	13			ns
Read delay time	t_{RDD}			9	ns
Read data setup time	t_{RDS}	7			ns
Read data hold time	t_{RDDH}	-7			ns
#WR delay time from #CE active	t_{CEWR}	13			ns
#WAIT setup time	t_{WTS}	7			ns
#WAIT hold time	t_{WTH}	-7			ns
Write signal pulse width	t_{WRW}	t_{CYC}(1 + WC) - 9			ns
Read signal pulse width	t_{RDW}	t_{CYC}(1 + WC) - 9			ns
Read address access time	t_{ACC}			t_{CYC}(2 + WC) - 18	ns
Chip enable access time	t_{CEAC}			t_{CYC}(2 + WC) - 18	ns
Read signal access time	t_{RDAC}			t_{CYC}(1 + WC) - 15	ns

WC: wait cycle (min. 0)

I.7.7.3 SPI AC Characteristics



Master mode

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
SPI_SCK*x* cycle time	t_{SPCK}	500			ns
SPI_SDI*x* setup time	t_{SDS}	70			ns
SPI_SDI*x* hold time	t_{SDH}	10			ns
SPI_SDO*x* output delay time	t_{SDO}			20	ns

Slave mode

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
SPI_SCK*x* cycle time	t_{SPCK}	500			ns
SPI_SDI*x* setup time	t_{SDS}	10			ns
SPI_SDI*x* hold time	t_{SDH}	10			ns
SPI_SDO*x* output delay time	t_{SDO}			80	ns

I.7.8 USB DC and AC Characteristics

Input levels

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = 0 to 70°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
VBUS input *1	V_{BUS}		4.40		5.25	V
High (driven) *2	V_{IH}		2.0			V
High (floating) *2	V_{IHZ}		2.7		3.6	V
Low *2	V_{IL}				0.8	V
Differential input sensitivity	V_{DI}	|DP - DM|	0.2			V
Differential common mode range	V_{CM}	Include VD1 range	0.8		2.5	V

*1) Refer to Section 7.2.1 in the USB2.0 Specification for the conditions.
*2) Refer to Section 7.1.4 in the USB2.0 Specification for the conditions.

Output levels

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = 0 to 70°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Low *3	V_{OL}		0		0.3	V
High (driven) *3	V_{OH}		2.8		3.6	V
Output signal crossover voltage *4	V_{CRS}		1.3		2.0	V

*3) Refer to Section 7.1.1 in the USB2.0 Specification for the conditions.
*4) Refer to Figures 7-8 and 7-9 in the USB2.0 Specification for the conditions.

Terminations

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = 0 to 70°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Bus pull-up resistor on upstream facing port (idle Bus) *5	R_{PUI}		0.9		1.575	kΩ
Bus pull-up resistor on upstream facing port (receiving) *5	V_{PUA}		1.425		3.090	kΩ

*5) Refer to ECN in the USB2.0 Specification for the conditions.

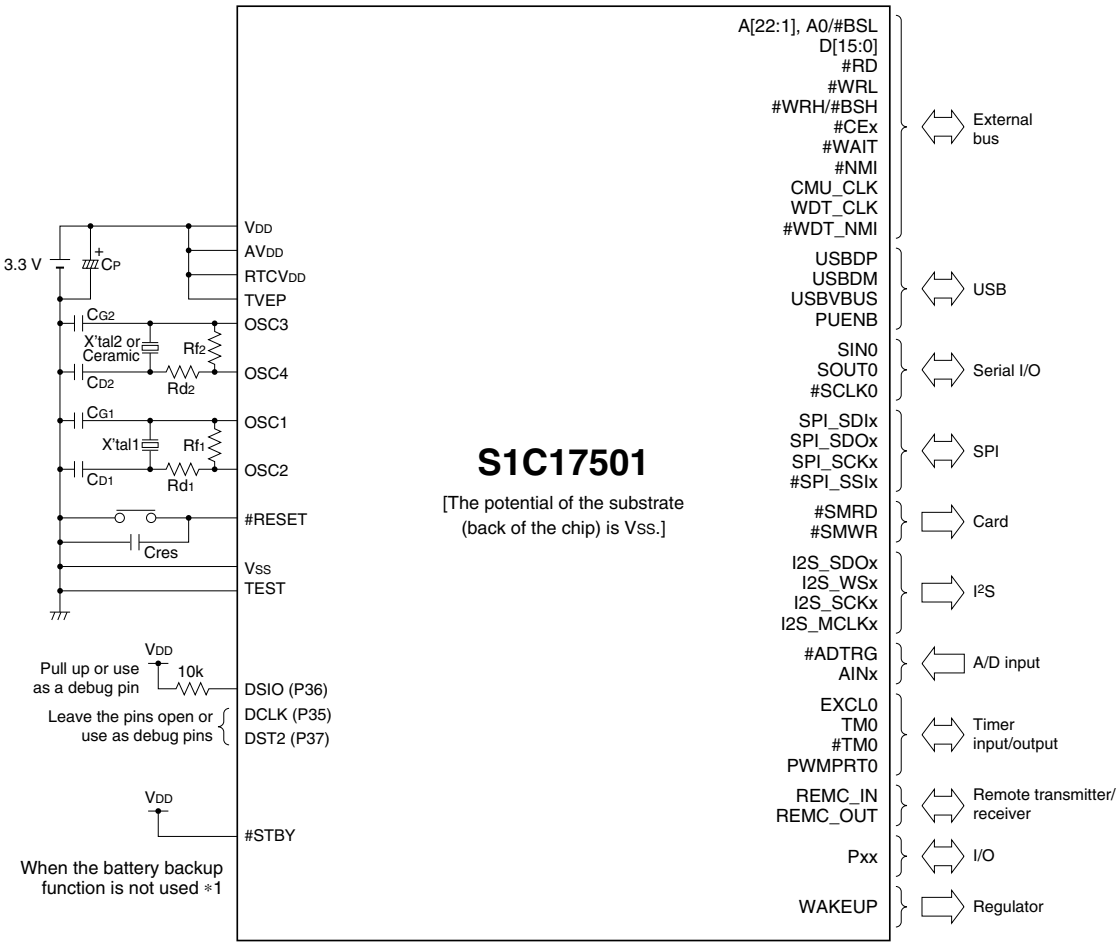
Driver characteristics

Unless otherwise specified: V_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = 0 to 70°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Rise time *4	T_{FR}		4		20	ns
Fall time *4	T_{FF}		4		20	ns
Differential rise and fall time matching	T_{FRFM}	T_{FR}/T_{FF}	90		111.11	%
Driver output resistance	Z_{DRV}		28		44	Ω
VBUS input impedance	Z_{VBUS}	R1 + R2	125			kΩ
VBUS resistor ratio		R1 : R2	1 : 2 (nominal)			

*4) Refer to Figures 7-8 and 7-9 in the USB2.0 Specification for the conditions.

I.8 Basic External Wiring Diagram



*1 Please refer to II.5.6 WAKEUP and #STBY when in use of the battery backup function.

Recommended values for external parts

External parts for the OSC1 oscillator circuit

Symbol	Resonator	Recommended manufacturer	Frequency [Hz]	Product number	Recommended values				Recommended operating condition
					C_{D1} [pF]	C_{G1} [pF]	R_{f1} [Ω]	R_{d1} [Ω]	Temperature range [°C]
X'tal1	Crystal	Epson Toyocom Corporation	32.768k	*1	–	–	–	–	–
		(Reference values)	32.768k	–	10	10	10M	0	-40 to 85

*1 Please contact the recommended manufacturer.

External parts for the OSC3 oscillator circuit

Symbol	Resonator	Recommended manufacturer	Frequency [Hz]	Product number	Recommended values *2				Recommended operating condition
					C_{D2} [pF]	C_{G2} [pF]	R_{f2} [Ω]	R_{d2} [Ω]	Temperature range [°C]
X'tal2	Crystal	Epson Toyocom Corporation	1M to 48M	*1	–	–	–	–	–
		(Reference values)	1M to 48M	–	15	15	1M	0	-40 to 85
Ceramic	Ceramic	Murata Manufacturing Co., Ltd.	1M	CSBFB1M00J58-R1 [SMD]	330	330	1M	680	-20 to 80
			1M	CSBLA1M00J58-B0 [leaded]	330	330	1M	680	-20 to 80
			4M	CSTCR4M00G55-R0 [leaded]	(39)	(39)	1M	470	-20 to 80
			4M	CSTLS4M00G56-B0 [leaded]	(47)	(47)	1M	330	-20 to 80
			10M	CSTCE10M0G55-R0 [SMD]	(33)	(33)	1M	220	-20 to 80
			10M	CSTLS10M0G56-B0 [leaded]	(47)	(47)	1M	220	-20 to 80
			20M	CSTCE20M0V53-R0 [SMD]	(15)	(15)	1M	0	-20 to 80
			20M	CSTCG20M0V53-R0 [small SMD]	(15)	(15)	1M	0	-20 to 80
			40M	CSTCW40M0X51-R0 [SMD]	(6)	(6)	1M	0	-20 to 80
			48M	CSTCZ48MOX12R**-R [SMD]	(10)	(10)	1M	0	-30 to 85

*1 Please contact the recommended manufacturer.

*2 The C_{D2} and C_{G2} values enclosed with ( ) are the built-in capacitances of the resonator.

Other

Symbol	Name	Recommended value
CP	Capacitor for power supply	3.3 µF
Cres	Capacitor for #RESET pin	0.47 µF

Notes: • The values in the above table are shown only for reference and not guaranteed.

• Crystal and ceramic resonators are extremely sensitive to influence of external components and printed-circuit boards. Before using a resonator, please contact the manufacturer for further information on conditions of use.

I.9 Precautions on Mounting

The following shows the precautions when designing the board and mounting the IC.

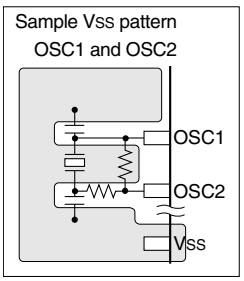
Oscillation Circuit

- Oscillation characteristics change depending on conditions such as components used (resonator, R_f, C_G, C_D) and board pattern. In particular, when a ceramic or crystal resonator is used, evaluate the components adequately under real operating conditions by mounting them on the board before the external register (R_f) and capacitor (C_G, C_D) values are finally decided.
- Disturbances of the oscillation clock due to noise may cause a malfunction. To prevent this, the following points should be taken into consideration. In particular, the latest devices are more sensitive to noise, as they are more finely processed.
  The measures against noise for the OSC2 pin, and the components and lines connected to this pin is most essential, and similar measures must also be taken for the OSC1 pin. The measures for the OSC1 and OSC2 pins are described below.
  We recommend taking measures similar to those for the high-speed oscillation system, including the OSC3 and OSC4 pins and the components and lines connected to these pins.

  (1) Components that are connected to the OSC1 (OSC3) and OSC2 (OSC4) pins, such as resonators, resistors, and capacitors, should be connected in the shortest line.

  (2) Whenever possible, configure digital signal lines with at least three millimeters clearance from the OSC1 (OSC3) and OSC2 (OSC4) pins and the components and lines connected to these pins. In particular, signals that are switched frequently must not be placed near these pins, components, and lines. The same applies to all layers on the multi-layered board as the distance between the layers is around 0.1 to 0.2 mm.
  Furthermore, do not configure digital signal lines in parallel with these components and lines when arranging them on the same or another layer of the board. Such an arrangement is strictly prohibited, even with clearance of three millimeters or more. Also, avoid arranging digital signal lines across these components and signal lines.

  (3) Shield the OSC1 (OSC3) and OSC2 (OSC4) pins and lines connected to those pins as well as the adjacent layers of the board using V_{SS}.
  As shown in the figure on the right, shield the wired layers as much as possible.
  Whenever possible, make the whole adjacent layers the ground layers, or ensure there is adequate shielding to a radius of five millimeters around the above pins and lines. As described in (2), do not configure digital signal lines in parallel with components and lines even if such precautionary measures are taken, and avoid configuring signal lines that are switched frequently across components and lines on other layers.



  (4) When an external clock is supplied to the OSC1 or OSC3 pin, the clock source should be connected to the OSC1 or OSC3 pin in the shortest line. Furthermore, do not connect anything else to the OSC2 or OSC4 pin.

  (5) After taking the above precautions, check the output clock waveform while operating the actual application program in the actual device.
  To do this, measure the output of the CMU_CLK pins with an oscilloscope.
  Check the waveform quality at the OSC3 output clock by measuring the CMU_CLK output. Ensure that the frequencies are as designed and that there is no noise or jitters.
  Check the waveform quality at the OSC1 clock by measuring the CMU_CLK output (after switching the system clock source to OSC1). Scale up the ranges around the rising and falling edges of the clock pulse to ensure that there is no noise, such as clock and spike, in the 100 ns ranges.

If conditions (1) to (3) are not satisfied, the OSC3 output may be jittery and the OSC1 output may be noisy. When the OSC3 output is jittery, the operating frequency will be lowered. When the OSC1 output is noisy, operation of the RTC using the OSC1 clock and the CPU core after the system clock is switched to OSC1 will be unstable.

Reset Circuit

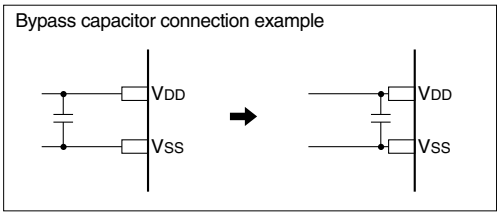
- The power-on reset signal which is input to the #RESET pin changes depending on conditions (power rise time, components used, board pattern, etc.). Decide the time constant of the capacitor and resistor after enough tests have been completed with the application product.
- In order to prevent any occurrences of unnecessary resetting caused by noise during operating, components such as capacitors and resistors should be connected to the #RESET pin in the shortest line.

Power Supply Circuit

- Sudden power supply variation due to noise may cause malfunction. Consider the following points to prevent this:

  (1) The power supply should be connected to the VDD, VSS, AVDD, and RTCVDD pins with patterns as short and large as possible. In particular, the power supply for AVDD affects A/D conversion precision.

  (2) When connecting between the VDD and VSS pins with a bypass capacitor, the pins should be connected as short as possible.

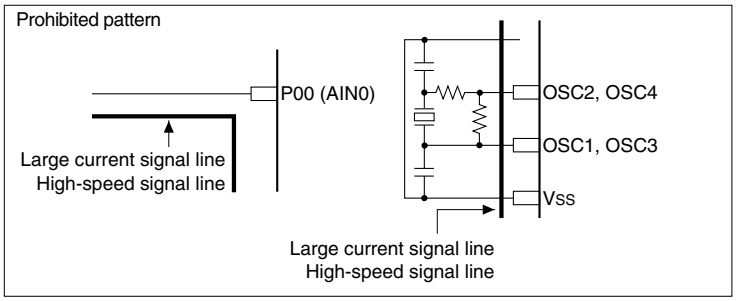


A/D Converter

- When the A/D converter is not used, the power supply pin AVDD for the analog system should be connected to VDD.

Arrangement of Signal Lines

- In order to prevent generation of electromagnetic induction noise caused by mutual inductance, do not arrange a large current signal line near the circuits that are sensitive to noise such as the oscillation unit and analog input unit.
- When a signal line is parallel with a high-speed line in long distance or intersects a high-speed line, noise may generated by mutual interference between the signals and it may cause a malfunction. Do not arrange a high-speed signal line especially near circuits that are sensitive to noise such as the oscillation unit and analog input unit.

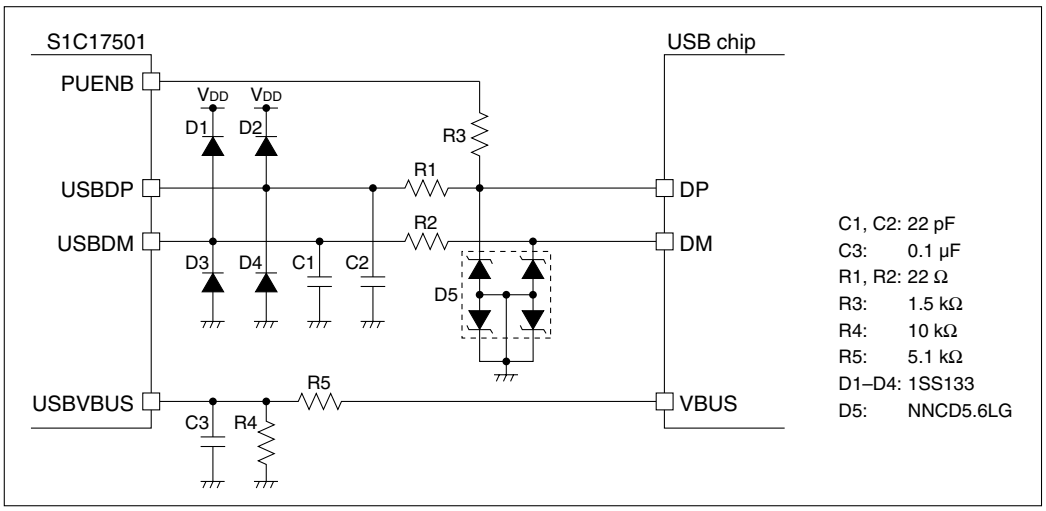


USB

The I/O block of the USB Function Controller incorporated in this chip has the following features:
The DP and DM pins can be connected directly to the USB connector.
The VBUS level is detected by means of a 2/3 resistive division internally in the chip, thus allowing for direct input of a 5 V-level signal.
The receiver does not enter a floating state even when the USB cable is disconnected from the USB connector. When the USB cable is disconnected, the VBUS pin is tied to Vss, so that leakage current will be the only source that drains power in the USB I/O block.

Precautions on VBUS

Be sure to not apply 6 V (max.) or more to the VBUS pin as the IC may be destroyed.
It is especially necessary to suppress overshoot on the input voltage and to prevent the host power source becoming unstable when the USB cable is plugged into the connector. The figure below shows an example of external connection.



In addition to the above, verify the VBUS state completely on the actual circuit board using an oscilloscope or other device. Overshoot and other symptoms are more likely to occur when using a long USB cable and connecting it to the host side connector.

Precautions on DP and DM

When designing a printed circuit board, observe the following precautions to ensure that both DP and DM signals are properly routed:

- To prevent signal skew and to stabilize differential impedance, the DP and DM signal lines must be routed in parallel and in the same length, with the pins and connector connected in the shortest distance possible. Crossed wiring of these signals should be avoided as much as possible.
- The periphery of these signal lines must be enclosed by a GND pattern, and with the GND pattern also created for the internal layer immediately below that. In particular, the routing of high-speed digital signal lines parallel to or across these signal lines should be avoided as much as possible.

We recommend that you verify the EYE pattern on the actual circuit board.

Noise-Induced Erratic Operations

If erratic IC operations appear to be attributable to noise, consider the following five points.

(1) TEST pin
If this pin is exposed to high-level noise, the entire IC enters test mode or a high-impedance state and becomes inoperable. In such cases, the IC will not be restored, even when the pin is returned to a low level. Therefore, always make sure the TEST pin is connected to GND on the circuit board. Although the IC contains internal pull-down resistors, it is susceptible to noise because these resistors are high impedance (approximately 50 to 100 kΩ).

(2) DSIO pin
Exposure of this pin to low-level noise causes the IC to enter debug mode. In debug mode, the clock is output from the DCLK pin and the DST2 pin is high, indicating that the IC is in debug mode.
In product versions, it is recommended that the DSIO pin be pulled high by connecting it directly to V_{DD} or through a resistor of 10 kΩ or less.
Although the IC contains internal pull-up resistors, it is susceptible to noise because these resistors are high impedance (approximately 50 to 100 kΩ).

(3) #RESET pin
Low-level noise on this pin resets the IC. However, the IC may not always be reset normally, depending on the input waveform.
Due to circuit design, this situation tends to occur when the reset input is in the high state, with high impedance.

(4) #NMI pin
Low-level noise on this pin causes an NMI interrupt. Due to the circuit design, this situation tends to occur when the #NMI pin is in the high state, with high impedance. Lower the impedance of #NMI when it is held high, or incorporate corrective measures into the software to protect against erratic operations.

(5) V_{DD}, $RTCV_{DD}$ and V_{SS} power supplies
If noise lower than the rated voltage enters one of these power-supply lines, the IC may operate erratically.
Take corrective measures in board design; for example, by using solid patterns for power supply lines, adding decoupling capacitors to eliminate noise, or incorporating surge/noise counteracting devices into the power supply lines.

To confirm the above, use an oscilloscope capable of observing higher-frequency waveforms of 200 MHz. The generation of fast noise may not be observed with a low-frequency oscilloscope.

If potential noise-induced erratic operations are detected through waveform observations using an oscilloscope, connect the suspected pin to the GND or power supply with low impedance (1 kΩ or less) and check once again. If erratic operations are no longer detected or occur at reduced frequency, or if different symptoms of erratic operations are observed, said pin may with reasonably certainty be considered to be the source of the erratic operations.

The TEST, DSIO, #RESET, and #NMI input circuits described above are designed to detect the edges of the input signal (#NMI can be changed to level sense mode), so that even spike noise may result in erratic operations. Among the digital signal circuits, these pins are most susceptible to noise.
In the design of the circuit board, take the following two points into consideration to protect the signal from noise.

(A) The most important measure is to lower the signal-driving impedance, as described in each item above. Connect pins to the power supply or GND, with impedance of 1 kΩ or less, preferably 0 Ω. In addition, limit the length of the connected signal lines to approximately 5 cm.

(B) Parallel routing of said signal lines with other digital lines on the board is undesirable, since the noise generated when the signal changes from high to low or vice versa may adversely affect signals. The signal may be subject to the most noise when signal lines are laid between multiple signal lines whose states change simultaneously. Take corrective measures by shortening the parallel distance (to several cm) or separating signal lines (2 mm or more).

Other

The 0.35 μm fine-pattern process is employed to manufacture this series of products.
Although the product is designed to meet EIAJ and MIL standards regarding basic IC reliability, please pay careful attention to the following points when actually mounting the chip on a board.
Since all the oscillator input/output pins are constructed to use the internal 0.35 μm transistors directly, the pins are susceptible to mechanical damage during the board-mounting process. Moreover, the pins may also be susceptible to electrical damage caused by such disturbances (listed below) whose electrical strength, varying gradually with time, could exceed the absolute maximum rated voltage (2.5 V) of the IC:

(1) Electromagnetic induction noise from the utility power supply in the reflow process during board-mounting, rework process after board-mounting, or individual characteristic evaluation (experimental confirmation), and
(2) Electromagnetic induction noise from the tip of a soldering iron

Especially when using a soldering iron, make sure that the IC GND and soldering iron GND are at the same potential before soldering.

II S1C17501 CLOCK SYSTEM

II.1 Clock System Diagram

Figure II.1.1 shows the clock system in the S1C17501.

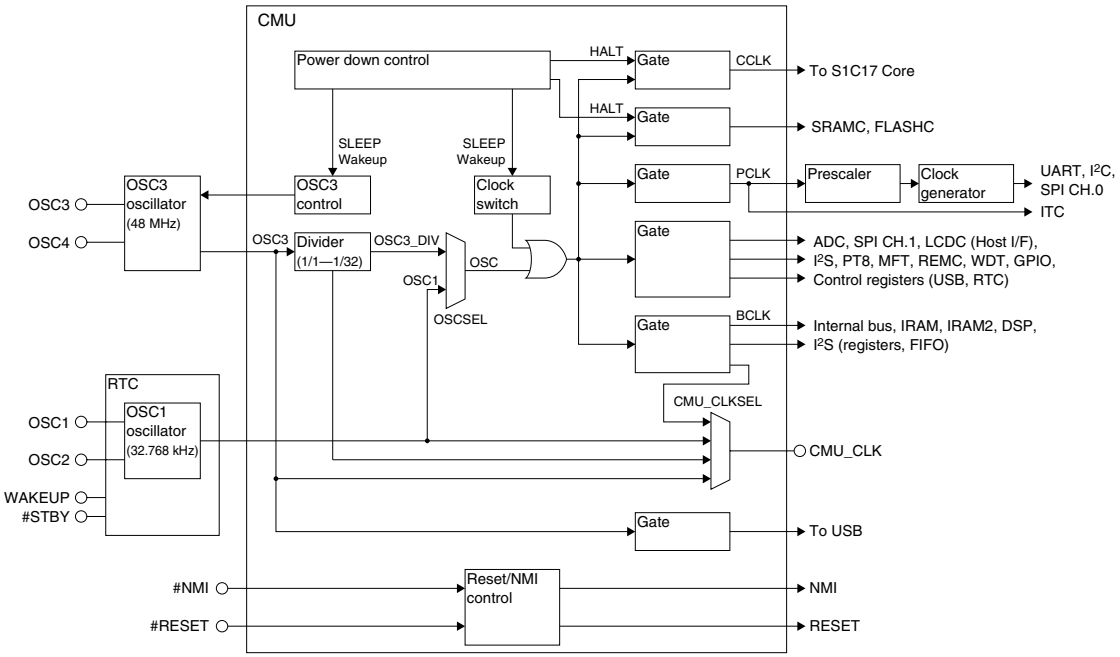


Figure II.1.1 Clock System Diagram

The S1C17501 controls the core and bus operating clocks using the clock management unit (CMU). The peripheral clocks are generated by the prescaler and clock generator.
Current consumption can be reduced by controlling the clocks according to the processing requirements as well as by using the standby mode. For methods to reduce current consumption, see Appendix C, “Power Saving.”

II.2 Clock Management Unit (CMU)

II.2.1 Overview of the CMU

The Clock Management Unit (CMU) controls the operating clock supplied to each functional module. The main functions of the CMU are outlined below.

- Controls reset and NMI inputs
- Selects the system clock source (OSC3 or OSC1)
- Controls the OSC3 oscillator circuit
- Clock control corresponding to standby modes (SLEEP and HALT)
- Selects divide ratio of the main system clock
- Selects an external bus clock
- Controls on/off of clock supply for each functional module

Through system clock selection, oscillator circuit, main system clock divide ratio selection and clock on/off control for each functional module, the CMU enables the most suitable operating clock frequency to be selected for the processing involved, as well as to turn off unnecessary clock supply, which combined with standby mode, helps to significantly reduce power consumption on the chip.

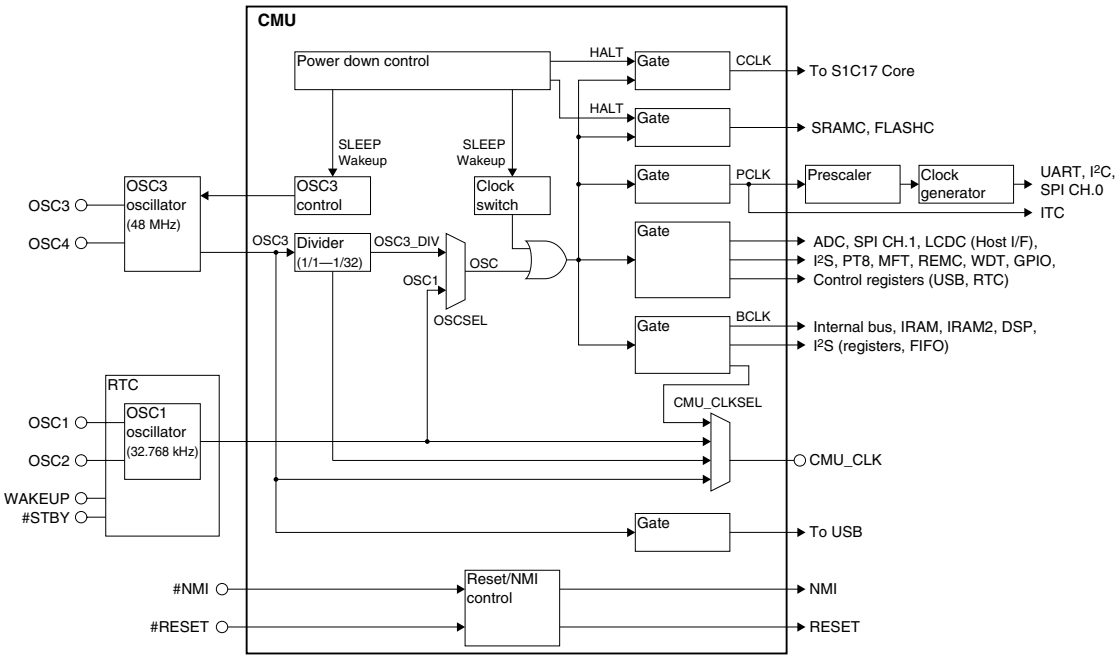


Figure II.2.1.1 CMU Block Diagram

Note: The CMU Control Registers at addresses 0x4900–0x4909 are write-protected. Before the CMU control registers can be rewritten, write protection of these registers must be removed by writing data 0x96 to the CMU Write Protect Register (0x4920). Note that since unnecessary rewrites to addresses 0x4900–0x4909 could lead to erratic system operation, the CMU Write Protect Register (0x4920) should be set to other than 0x96 unless said CMU control registers must be rewritten.

II.2.2 Reset Input and Initial Reset

The CMU also has a function to generate an internal reset signal from external reset input (#RESET).

II.2.2.1 Initial Reset Pin

The #RESET pin is used for initial reset input from outside the IC. Set the #RESET pin to 0 (low) to reset the IC.
The #RESET input signal is sampled with the OSC3 clock. Therefore, the chip cannot be reset when the OSC3 clock is not input or generated. Moreover, to assert the internal reset signal #RESET = 0 must be continuously detected at least three times in this sampling. The #RESET signal should be held low for at least three OSC3 clock cycles to confirm that the chip is reset. Also the internal reset signal is negated when #RESET = 1 (high) is continuously detected three times.
The S1C17501 is reset by the low state (= 0) on the internal reset signal, and starts operating when the reset signal is released back to high (= 1).

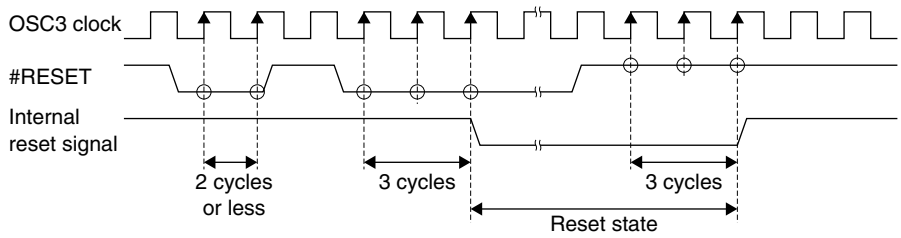


Figure II.2.2.1.1 #RESET Sampling

II.2.2.2 Initial Reset Status

The S1C17 Core and internal peripheral circuits are initialized while the internal reset signal is kept 0. The following shows the internal reset status:

CPU PC:	The reset vector at address 0x20000 is loaded to the PC.
CPU PSR:	All the PSR bits are reset to 0.
Other CPU registers:	All the registers are cleared to 0.
TTBR:	Initialized to 0x20000
CPU operating clock:	The CPU operates with the OSC3 × 1/1 clock.
Oscillator circuit:	The OSC3 oscillator circuit is turned on. The OSC1 oscillator circuit is always on.
Clock supply to peripheral modules:	All clocks are enabled except for the USB.
I/O pin status:	Initialized (see Section I.3.2, “Pin Functions.”)
Other peripheral modules:	Initialized or undefined (see each I/O map.)

Note: The S1C17501 does not support a hot reset feature that maintains I/O pin status and the TTBR value.

II.2.2.3 Power-on Reset

When turning on the power for the chip, always be sure to reset the chip to ensure that it will start operating normally.
Since the #RESET pin is a gate input, a power-on reset circuit should be configured external to the chip.
Initial reset (#RESET = 0) causes the OSC3 oscillator circuit to start oscillating, and when the reset signal is released back high, the CPU starts operating with the OSC3 clock. The OSC3 oscillator circuit requires a finite time until its oscillation stabilizes after it starts operating. To confirm that the CPU is started, the initial reset can only be deasserted after this oscillation stabilization time elapses.

Note: The oscillation start time of the OSC3 oscillator circuit varies with the device used, board patterns, and operating environment. Therefore, sufficient time should be provided before the reset signal is deasserted.

Power-on sequence

To ensure that the chip will operate normally, observe the timing requirements given below when turning on the power for the chip.

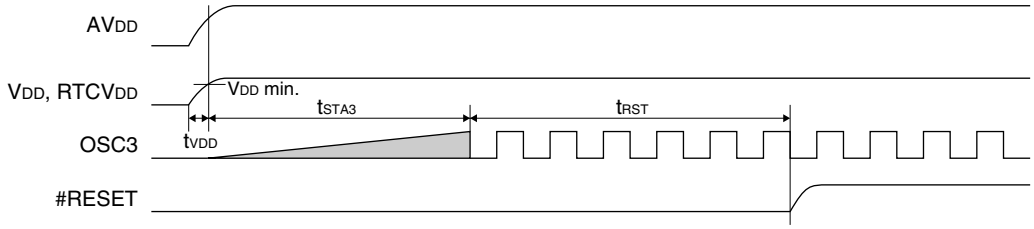


Figure II.2.2.3.1 Power-on Sequence

(1) t_{VDD}: Elapsed time until the power supply stabilizes after power-on
Supply power in the following sequence (or simultaneously).
Power-on: V_{DD} (and $RTCV_{DD}$*) → AV_{DD} (A/D) → Apply the input signal

* The $RTCV_{DD}$ can be always supplied to the chip to operate the RTC and to back up the IRAM2.

(2) t_{STA3}: Time at which OSC3 oscillation starts

(3) t_{RST}: Minimum reset pulse width
Time at which the clock supplied to the chip stabilizes plus at least six clocks; Keep the #RESET signal low.

Make sure #RESET is held low (= 0) for at least 6 clock cycles after the OSC3 clock supplied to the CMU has stabilized.

II.2.2.4 Precautions to be Taken during Initial Reset

Core CPU

When initially reset, all internal registers of the core CPU are cleared to 0. The Stack Pointer (SP) also becomes 0 when it is initialized upon reset. Note that normal operation of the program cannot be guaranteed if an interrupt occurs before the stack is set up, as the PC or PSR value may be saved to an indeterminate location. To prevent such a problem, set the SP before an interrupt occurs.

Internal RAM

The content of internal RAM becomes undefined when initially reset. Internal RAM must be initialized as required.

OSC3 oscillator circuit

When initially reset, the OSC3 oscillator circuit starts oscillating, and when the reset signal is deasserted, the CPU starts operating with the OSC3 clock. To prevent erratic operation due to an instable clock when the chip is reset at power-on or while the OSC3 oscillator circuit is idle, the reset signal should not be deasserted until after oscillation stabilizes.

OSC1 oscillator circuit

When the chip is reset at power-on, the OSC1 oscillator circuit also starts oscillating. The OSC1 oscillator circuit requires a longer time for oscillation to stabilize than the OSC3 oscillator circuit. (See the electrical characteristics table.) To prevent erratic operation due to an instable clock, the OSC1 clock should not be used until after this stabilization time elapses.

Input/output ports and input/output pins

Initial reset initializes the control and data registers of the input/output ports, therefore, be set up back again in a program.

Other internal peripheral circuits

The control and data registers of other peripheral circuits are initialized or made unstable by initial reset. Therefore, these registers should be set up as required in a program.
For details on how peripheral circuits are initialized by initial reset, see each I/O map or circuit description.

II.2.3 NMI Input

The NMI signal, which is input from the #NMI pin or generated by the watchdog timer (WDT), is input to the CMU, then forwarded to the CPU. For details about NMI exception handling by the CPU, refer to the S1C17 Family S1C17 Core Manual.

Notes:
- At least a 3-system clock width of low pulse is required to generate NMI. After the NMI signal falls, maintain it at a low level for 3 or more clock cycles.
- NMI cannot be nested. The CPU keeps NMI input masked out until the reti instruction is executed after an NMI exception occurred.

II.2.4 Selecting the System Clock Source

The CMU has the following two clock inputs, one of which can be selected as the source clock (OSC) for the system.

1. OSC3 clock

This clock is generated by the OSC3 oscillator circuit or supplied from an external source through the OSC3 pin. For details about the OSC3 oscillator circuit, see Section II.2.5.1, "OSC3 Oscillator Circuit."

2. OSC1 clock

This is the source clock (32.768 kHz, typ.) for the Real Time Clock (RTC). When high-speed operation is unnecessary, this low-speed clock may be used to operate the system, thus helping to reduce power consumption on the chip. For details about the OSC1 oscillator circuit, see Section II.2.5.3, "OSC1 Oscillator Circuit."

The clock source can be selected using OSCSEL (D2/CMU_SYSCLKCTL register).

* **OSCSEL**: OSC Clock Selection Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D2/0x4900)

When OSCSEL is set to 0 (default), OSC3 is selected as the system clock source; when OSCSEL is set to 1, OSC1 is selected.

The clock source changed here is not reflected until after the CPU returns from SLEEP mode. Therefore, the slp instruction must be executed once after setting OSCSEL. Although the CPU returns from SLEEP mode to normal operation by an external interrupt from a port, for example, several functions are provided for use in clock source changes, thus automatically returning the CPU from SLEEP mode a certain time after slp instruction execution or leaving the OSC3 oscillator circuit turned on during SLEEP mode. Section II.2.8, "Standby Modes," describes these methods of control in detail.

Note: When clock sources are changed, the CMU control registers must be set so that the CMU is supplied with a clock from the selected clock source upon returning from SLEEP mode immediately after the change. Otherwise, the chip does not restart after the return from SLEEP mode.

II.2.5 Controlling the Oscillator Circuit

II.2.5.1 OSC3 Oscillator Circuit

The OSC3 oscillator circuit generates the main clock with which to operate the S1C17 Core and internal peripheral circuits.

Input/output pins of the OSC3 oscillator circuit

Table II.2.5.1.1 lists the input/output pins of the OSC3 oscillator circuit.

Table II.2.5.1.1 Input/Output Pins of the OSC3 Oscillator Circuit

Pin name	I/O	Function
OSC3	I	OSC3 oscillator input pin: Crystal/ceramic oscillator or external clock input
OSC4	O	OSC3 oscillator output pin: Crystal/ceramic oscillator (left open when using external clock input)

Structure of the oscillator circuit

The OSC3 oscillator circuit accommodates a crystal/ceramic oscillator and external clock input.
Figure II.2.5.1.1 shows the structure of the OSC3 oscillator circuit.

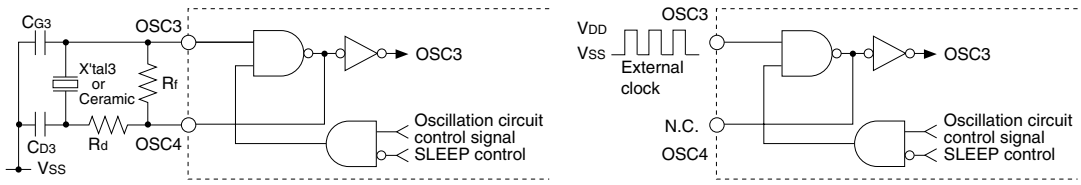


Figure II.2.5.1.1 OSC3 Oscillator Circuit

For use as a crystal or ceramic oscillator circuit, connect a crystal (X'tal3) or ceramic resonator and a feedback resistor (R_f), two capacitors (C_{G3}, C_{D3}) and, if necessary, a drain resistor (R_d) to the OSC3 and OSC4 pins and V_{SS}.
To use an external clock, leave the OSC4 pin open and input a V_{DD}-level clock (with a 50% duty cycle) to the OSC3 pin.
The range of oscillation frequencies is as follows:

- Crystal oscillator: 1 MHz (min.) to 48 MHz (max.)
- Ceramic oscillator: 1 MHz (min.) to 48 MHz (max.)
- External clock input: 48 MHz (max.)
- A 48 MHz clock source with 0.25% of accuracy should be connected for using the USB function.

For details of oscillation characteristics and external clock input characteristics, see "Electrical Characteristics."

Oscillation control

CMU register control bit SOSC3 (D1/CMU_SYSCLKCTL register) is used to control OSC3 oscillation.

∗ **SOSC3**: OSC3 Oscillator On/Off Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D1/0x4900)

Setting this control bit to 0 causes the OSC3 oscillator circuit to stop; setting it to 1 causes the OSC3 oscillator circuit to start oscillating, thereby outputting a clock signal waveform. When initially reset, this bit is set to 1 for enabling OSC3 oscillation.

Note: When the oscillator is made to start oscillating by setting SOSC3 from 0 to 1, a finite time is required until oscillation stabilizes (see "Electrical Characteristics"). To prevent system malfunction, do not use the oscillator-derived clock until this oscillation stabilization time elapses.

II.2.5.2 Setting the OSC3 Divider

An OSC3 divided clock can be used as the system clock when OSC3 is selected as the system clock source. Setting the system clock to the lowest frequency possible according to the processing can reduce current consumption.
The OSC3 divider generates six kinds of clocks from OSC3•1/1 to OSC3•1/32. Select a divided clock from those six clocks using OSC3DIV[2:0] (D[2:0]/CMU_OSC3DIV register).

∗ **OSC3DIV[2:0]**: OSC3 Clock Divider Selection Bits in the OSC3 Clock Divider (CMU_OSC3DIV) Register (D[2:0]/0x4903)

Table II.2.5.2.1 Selecting an OSC3 Divided Clock

OSC3DIV[2:0]	OSC3_DIV clock
0x7	OSC3•1/1
0x6	OSC3•1/1
0x5	OSC3•1/32
0x4	OSC3•1/16
0x3	OSC3•1/8
0x2	OSC3•1/4
0x1	OSC3•1/2
0x0	OSC3•1/1

(Default: 0x0 = OSC3•1/1)

A divided clock can be selected at any time. However, up to 32 OSC3 clock cycles are required before the clocks are actually changed after altering the register values.

II.2.5.3 OSC1 Oscillator Circuit

The S1C17501 contains an oscillator circuit (OSC1) used to generate a 32.768 kHz (typ.) clock as the clock source for timekeeping operation of the RTC.
The OSC1 clock can also be used as a power-saving operating clock for the core system or peripheral circuits.

Input/output pins of the OSC1 oscillator circuit

Table II.2.5.3.1 lists the input/output pins of the OSC1 oscillator circuit.

Table II.2.5.3.1 Input/Output Pins of the Low-speed (OSC1) Oscillator Circuit

Pin name	I/O	Function
OSC1	I	OSC1 oscillator input pin: Crystal oscillator or external clock input
OSC2	O	OSC1 oscillator output pin: Crystal oscillator (left open when using external clock input)

Structure of the OSC1 oscillator circuit

The OSC1 oscillator circuit accommodates a crystal oscillator and external clock input. As for the RTC, $RTCV_{DD}$ is used to supply power to this circuit.
Figure II.2.5.3.1 shows the structure of the OSC1 oscillator circuit.

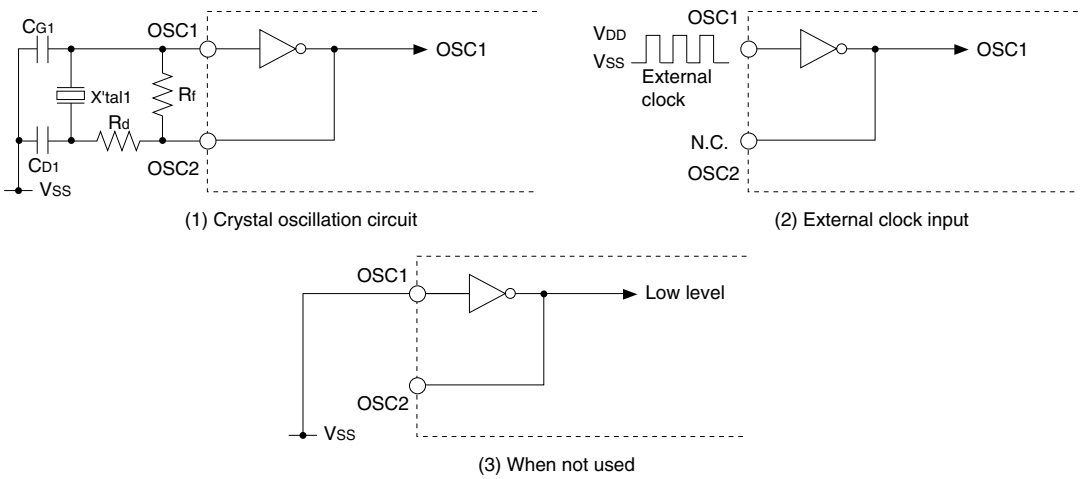


Figure II.2.5.3.1 OSC1 Oscillator Circuit

For use as a crystal oscillator circuit, connect a crystal resonator X'tal1 (32.768 kHz, typ.), feedback resistor (R_f), two capacitors (C_{G1}, C_{D1}), and, if necessary, a drain resistor (R_d) to the OSC1 and OSC2 pins and V_{SS}, as shown in Figure II.2.5.3.1 (1).
To use an external clock, leave the OSC2 pin open and input an RTCV_{DD} level clock (whose duty cycle is 50%) to the OSC1 pin.
The oscillator frequency/input clock frequency is 32.768 kHz (typ.). Make sure the crystal resonator or external clock used in the RTC has this clock frequency. With any other clock frequencies, the RTC cannot be used for timekeeping purposes.
For details of oscillation characteristics and the input characteristics of external clock, see "Electrical Characteristics."
When not using the OSC1 oscillator circuit, connect the OSC1 pin to V_{SS} and leave the OSC2 pin open.

Oscillation control

The OSC1 oscillator always operates without controlling using a register.

Note: When the oscillator is made to start oscillating at power-on, a finite time (of up to 3 seconds) is required until oscillation stabilizes. To prevent system malfunction, do not use the oscillator-derived clock until this oscillation stabilization time elapses.

II.2.6 Controlling Clock Supply

To reduce power consumption on the chip, a function is provided to turn off clock supply independently for each functional module.

II.2.6.1 Clock Supply to the S1C17 Core

In normal mode, the CMU always supplies the operating clock (CCLK) to the S1C17 Core.
When the S1C17 Core executes the halt or slp instruction, the CMU stops supplying the clock to the S1C17 Core and the S1C17 Core enters a standby (HALT or SLEEP) mode. The CMU resumes the clock supply to the S1C17 Core when the standby mode is cancelled by occurrence of an interrupt. For details of the standby mode, see Section II.2.8, "Standby Modes."

II.2.6.2 Clock Supply to Core Peripheral Modules

The core peripheral modules shown below use the core peripheral clock (PCLK).
- Prescaler
- Clock generator (16-bit/8-bit timers)
- UART
- SPI CH.0
- I^2C
- Interrupt controller

The PCLK supply can be controlled using PCLK_EN (D0/CMU_GATEDCLK0 register)

∗ **PCLK_EN**: Core Peripheral Clock Control Bit in the Gated Clock Control 0 (CMU_GATEDCLK0) Register (D0/0x4906)

When initially reset, PCLK_EN is set to 1 (on), with the clock supplied to the core peripheral modules. If all the core peripheral modules are not used, set PCLK_EN to 0 to reduce current consumption.

Note: In HALT mode, PCLK does not stop if PCLK_EN is set to 1 (on). To stop supplying the clock in HALT mode, PCLK_EN should be set to 0 before executing the halt instruction. PCLK will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

II.2.6.3 Clock Supply to Extended Peripheral Modules

Table II.2.6.3.1 lists the control bits used for controlling the operating clock supply to the extended peripheral modules. The modules listed here have one controllable clock path, so they can be turned on/off using the corresponding control bit only. See Sections II.2.6.4 to II.2.6.10 for controlling the SRAMC, FLASHC, USB, RTC, I^2S, and core bus module operating clocks.

Table II.2.6.3.1 Extended Peripheral Clock Supply Control Bits

Module	Clock	Control bit	Register
8-bit programmable timers	PT8_CLK	PT8_CLK_EN (D1)	Gated Clock Control 1 (CMU_GATEDCLK1) Register (0x4907)
Multi-function timer	MFT_CLK	MFT_CLK_EN (D0)	
SPI CH.1 ∗	SPI_CLK	SPI_CLK_EN (D5)	Gated Clock Control 2 (CMU_GATEDCLK2) Register (0x4908)
Remote controller	REMC_CLK	REMC_CLK_EN (D4)	
A/D converter	ADC_CLK	ADC_CLK_EN (D3)	
Watchdog timer	WDT_CLK	WDT_CLK_EN (D2)	
I/O ports	PORT_CLK	PORT_CLK_EN (D1)	

∗ The prescaler for SPI CH.1 is included in the 8-bit programmable timer module. Therefore, the PT8_CLK supply must be enabled in addition to the SPI_CLK supply when SPI CH.1 is used.

When initially reset, these control bits are set to 1 (on), with clocks supplied to each module. If any module is unused, set the corresponding control bit to 0, thus turning the clock for that module off.

Note: These clocks do not stop in HALT mode if the corresponding control bits are set to 1 (on). To stop supplying the clock in HALT mode, the control bit should be set to 0 (off) before executing the halt instruction. All these clocks will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

II.2.6.4 Clock Supply to the SRAMC

The SRAMC provides SRAMC_CLK_EN (D7/CMU_GATEDCLK1 register) for controlling the SRAMC clock (SRAMC_CLK). However, the SRAMC controls the internal peripheral bus (SAPB) and external bus, so SRAMC_CLK cannot be stopped while the IC is running. In other words, SRAMC_CLK does not stop in normal operation mode (except when the halt or slp instruction is executed) even if SRAMC_CLK_EN is set to 0. However, SRAMC_CLK can be automatically turned off in HALT mode (after the halt instruction is executed) by setting SRAMC_CLK_EN to 0 (default: on).

* **SRAMC_CLK_EN**: SRAMC Clock Control (in HALT mode) Bit in the Gated Clock Control 1 (CMU_GATEDCLK1) Register (D7/0x4907)

When initially reset, SRAMC_CLK_EN is set to 1 (on) to enable the SRAMC_CLK supply. If SRAMC_CLK is unused in HALT mode, set SRAMC_CLK_EN to 0 (off). The SRAMC_CLK supply will be stopped after the CPU executes the halt instruction. SRAMC_CLK will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

II.2.6.5 Clock Supply to the FLASHC

The FLASHC provides FLASHC_CLK_EN (D7/CMU_GATEDCLK0 register) for controlling the FLASHC clock (FLASHC_CLK). However, The FLASHC is required for executing the program in the Flash memory, so the FLASHC_CLK cannot be stopped while the IC is running. In other words, the FLASHC_CLK does not stop in normal operation mode (except when the halt or slp instruction is executed) even if FLASHC_CLK_EN is set to 0. However, FLASHC_CLK can be automatically turned off in HALT mode (after the halt instruction is executed) by setting FLASHC_CLK_EN to 0 (default: on).

* **FLASHC_CLK_EN**: FLASHC Clock Control (in HALT mode) Bit in the Gated Clock Control 0 (CMU_GATEDCLK0) Register (D7/0x4906)

When initially reset, FLASHC_CLK_EN is set to 1 (on) to enable the FLASHC_CLK supply. If FLASHC_CLK is unused in HALT mode, set FLASHC_CLK_EN to 0 (off). The FLASHC_CLK supply will stop after the CPU executes the halt instruction. FLASHC_CLK will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

II.2.6.6 Clock Supply to the USB

The CMU provides the clock paths with the control bit shown below for the USB module. The clock supply turns on when the control bit is set to 1 and it turns off when the control bit is set to 0.

(1) USB clock (USB_CLK)

This clock (OSC3 = 48 MHz) is used for the USB interface module. USB_CLK_EN (D4/CMU_GATEDCLK0 register) is used for clock supply control (default: off). In HALT mode, USB_CLK does not stop if USB_CLK_EN is set to 1 (on). To stop supplying the clock in HALT mode, USB_CLK_EN should be set to 0 (off) before executing the halt instruction.

Also in SLEEP mode, USB_CLK does not stop if USB_CLK_EN is set to 1 (on), SOSC3 (D1/CMU_SYSCLKCTL register) is set to 1, and OSC3OFF (D7/CMU_SYSCLKCTL register) is set to 0. USB_CLK will stop in SLEEP mode only under other setting conditions (see Section II.2.8.2 for more information).

* **USB_CLK_EN**: USB IP 48 MHz Clock Control Bit in the Gated Clock Control 0 (CMU_GATEDCLK0) Register (D4/0x4906)
* **SOSC3**: OSC3 Oscillator On/Off Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D1/0x4900)
* **OSC3OFF**: OSC3 Disable During SLEEP Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D7/0x4900)

(2) Control register clock (USB_SAPB_CLK)

This clock is used to control the USB registers. This clock is required for accessing the USB registers and it can be stopped when not in use. USB_SAPB_CLK_EN (D5/CMU_GATEDCLK0 register) is used for clock supply control (default: off). In HALT mode, USB_SAPB_CLK does not stop if USB_SAPB_CLK_EN is set to 1 (on). To stop supplying the clock in HALT mode, USB_SAPB_CLK_EN should be set to 0 (off) before executing

the halt instruction. USB_SAPB_CLK will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

* **USB_SAPB_CLK_EN**: USB SAPB I/F Clock Control Bit in the Gated Clock Control 0 (CMU_GATEDCLK0) Register (D5/0x4906)

II.2.6.7 Clock Supply to the RTC

The CMU provides the clock paths with a control bit shown below for the RTC.

(1) 32.768 kHz clock (OSC1)

This clock (OSC1 = 32.768 kHz) is used for timekeeping operations of the RTC. This clock is always supplied to the RTC (even in the standby mode).

(2) Control register clock (RTC_SAPB_CLK)

This clock (MCLK) is used to control the RTC registers. This clock is required for accessing the RTC registers and it can be stopped when not in use. RTC_SAPB_CLK_EN (D0/CMU_GATEDCLK2 register) is used for clock supply control (default: on).

* **RTC_SAPB_CLK_EN**: RTC SAPB I/F Clock Control Bit in the Gated Clock Control 2 (CMU_GATEDCLK2) Register (D0/0x4908)

The clock supply turns on when RTC_SAPB_CLK_EN is set to 1 and it turns off when it is set to 0. In HALT mode, RTC_SAPB_CLK does not stop if RTC_SAPB_CLK_EN is set to 1 (on). To stop supplying the clock in HALT mode, RTC_SAPB_CLK_EN should be set to 0 (off) before executing the halt instruction. RTC_SAPB_CLK will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

II.2.6.8 Clock Supply to the I²S

The I²S interface clocks can be controlled using I2SEN0 (D8/I2S_CTL_OUT register) and I2SEN1 (D0/I2S_CTL_IN register) that are provided in the I²S module.

* **I2SEN0**: I²S Ch.0 Enable Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D8/0x5300)
* **I2SEN1**: I²S CH.1 Enable Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D0/0x5302)

The I²S interface CH.0 clocks activate when I2SEN0 is set to 1 and deactivate when it is set to 0. The I²S interface CH.1 clocks activate when I2SEN1 is set to 1 and deactivate when it is set to 0.
In HALT mode, the I²S interface clocks do not stop if I2SEN0 or I2SEN1 is set to 1 (on). To stop supplying the clocks in HALT mode, I2SEN0 and/or I2SEN1 should be set to 0 (off) before executing the halt instruction. The I²S interface clocks will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

II.2.6.9 Clock Supply to the Core Bus and Other Modules

In normal mode, the CMU always supplies the operating clock (BCLK) to the S1C17 Core bus modules, IRAM, IRAM2, multiplier (MAC), I²S control registers, and I²S FIFO.
The BCLK does not stop even if the S1C17 Core executes the halt instruction. BCLK will stop when the S1C17 Core executes the slp instruction (see Section II.2.8.2 for more information).

II.2.7 Setting the External Clock Output (CMU_CLK)

CMU_CLK is an external output clock for the external devices.
CMU_CLK can be selected from 10 clocks using CMU_CLKSEL[3:0] (D[3:0]/CMU_CMUCLK register).

* **CMU_CLKSEL[3:0]**: CMU_CLK Selection Bits in the CMU_CLK Select (CMU_CMUCLK) Register (D[3:0]/0x4905)

Table II.2.7.1 Selecting CMU_CLK

CMU_CLKSEL[3:0]	CMU_CLK
0xf–0xa	Reserved
0x9	OSC3•1/32
0x8	OSC3•1/16
0x7	OSC3•1/8
0x6	OSC3•1/4
0x5	OSC3•1/2
0x4	OSC3•1/1
0x3	Reserved
0x2	BCLK
0x1	OSC1
0x0	OSC3

(Default: 0x0 = OSC3)

CMU_CLK can be selected at any time. However, switching over the clocks creates hazards.
When CMU_CLK must be output to external devices, it is also necessary to select a port function. For details on how to control clock output and the port to be used, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

Note: Settings other than those listed in Table II.2.7.1 are reserved for testing. Do not set undescribed values to CMU_CLKSEL[3:0] as undesired clocks may output.

II.2.8 Standby Modes

The S1C17501 supports two standby modes: HALT and SLEEP. Power consumption on the chip can be greatly reduced by placing the CPU in one of these standby modes. Moreover, the CPU must be placed in SLEEP mode before clock sources for the system (OSC3 or OSC1) are switched over (see Sections II.2.9.1 and II.2.9.2 for more information).

II.2.8.1 HALT Mode

The CPU suspends program execution upon executing the halt instruction and enters HALT mode.
In HALT mode, the CPU stops operating. Furthermore, the SRAMC clock (SRAMC_CLK) and FLASHC clock (FLASHC_CLK) can be stopped in HALT mode (after the halt instruction is executed) by setting SRAMC_CLK_EN (D7/CMU_GATEDCLK1 register) and FLASHC_CLK_EN (D7/CMU_GATEDCLK0 register) to 0, respectively (see Sections II.2.6.4 and II.2.6.5 for the SRAMC and FLASHC clocks). The other internal peripheral circuits remain in the state (idle or operating) held when the halt instruction was executed.
The CPU is released from HALT mode by initial reset, an NMI or other interrupt, or a forcible break from the debugger.
HALT mode is effective in reducing power consumption on the chip when running the CPU is unnecessary, such as when waiting for external input or responses from peripheral circuits. When the CPU is released from HALT mode by an interrupt, it enters a program executable state by interrupt processing and executes an interrupt handling routine for the interrupt generated. In interrupt processing of the CPU, the address for the instruction next to halt is saved to the stack as a return address from the interrupt handling routine, so that the reti instruction in the interrupt handling routine branches to the instruction next to halt.
The CPU is released from HALT mode when the interrupt controller (ITC) asserts the interrupt signal to be sent to the CPU. In other words, when an interrupt flag of the interrupts that have been enabled by the interrupt enable bits in the ITC is set to 1, the CPU can be released from HALT mode even if the PSR is set to disable interrupts. However, in this case the CPU does not execute the interrupt handling routine.
The #NMI signal releases the CPU from HALT mode when it goes low level.

II.2.8.2 SLEEP Mode

The CPU suspends program execution upon executing the slp instruction and enters SLEEP mode. In SLEEP mode, the CPU stops operating and the CMU can stop supplying a clock to each functional module (see Section II.2.6 for more information). Therefore, all peripheral circuits (except the OSC1 oscillator circuit and RTC) stop operating. Note that before the CMU actually stops clock output after initiating processing to enter SLEEP mode, up to 8 clock cycles of the source clock (OSC) then selected are required.
The CPU is reawaken from SLEEP mode (when WAKEUPWT = 1) by initial reset, RTC interrupt (level triggered), #NMI signal, or other interrupt from an external device (port input interrupt with level triggered). See Sections II.2.9.3 and II.2.9.4 for more information on the CMU register settings.
When the CPU is reawaken from SLEEP mode by an interrupt, it enters a program executable state by interrupt processing and executes an interrupt handling routine for the interrupt generated. In interrupt processing of the CPU, the address for the instruction next to slp is saved to the stack as a return address from the interrupt handling routine, so that the reti instruction in the interrupt handling routine branches to the instruction next to slp.
Cause-of-interrupt flags in the interrupt controller (ITC) cannot be set in SLEEP mode as the clock is not supplied to the ITC in SLEEP mode.
Therefore, when the clock is not supplied to the ITC, the level triggered interrupt signals from the interrupt sources that have been enabled to generate an interrupt are input to the CMU through the ITC and used to wake up the CPU from SLEEP mode. In this case, the interrupt flag is set after the clock has started supplying to the ITC. The CPU can wake up from SLEEP mode by a cause of interrupt as described above even if the PSR is set to disable interrupts, note however, that the CPU does not execute the interrupt handling routine.
The #NMI signal releases the CPU from SLEEP mode when it goes low level.

Notes: • In SLEEP mode, there is a time lag between input of an interrupt signal for wakeup and the start of the clock supply to the ITC, so a delay will occur until the interrupt controller (ITC) sets the interrupt flag. Therefore, no interrupt will occur if the interrupt signal is deasserted before the clock is supplied to the ITC, as the interrupt flag in the ITC is not set.

Furthermore, additional time is needed for the CPU to accept the interrupt request from the ITC, the CPU may execute a few instructions that follow the slp instruction before it starts the interrupt processing. When using a port input interrupt is used to wake up the CPU from SLEEP mode, set the interrupt trigger mode to level trigger (see Section IV.1.3.5) and assert the input signal until the clock supply has started.
The same problem may occur when the CPU wakes up from SLEEP mode by NMI. No interrupt will occur if the #NMI signal is deasserted before the clock is supplied, as the NMI flag is not set.

- Before setting the IC to SLEEP mode, the clock supply for the USB must be disabled.

Stopping OSC3 oscillation and waiting for oscillation stabilization at wakeup

By default, the OSC3 oscillator circuit does not stop operating in SLEEP mode. OSC3 oscillation can be made to stop during SLEEP mode by setting OSC3OFF (D7/CMU_SYSCLKCTL register).

∗ **OSC3OFF**: OSC3 Disable During SLEEP Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D7/0x4900)

Setting OSC3OFF to 1 causes OSC3 oscillation to stop during SLEEP mode. In this case, the OSC3 oscillator circuit starts oscillating when the CPU is reawaken from SLEEP mode. However, since the CPU may operate erratically if it starts operating with the OSC3 clock before the oscillation stabilizes, an OSC oscillation start wait timer is provided to keep the CPU waiting a while before it starts operating. The wait time can be set by using OSCTM[7:0] (D[7:0]/CMU_OSC3_WCNT register) and TMHSP (D6/CMU_SYSCLKCTL register).

∗ **OSCTM[7:0]**: OSC3 Oscillation Stabilization-Wait Timer in the OSC3 Wait Timer (CMU_OSC3_WCNT) Register (D[7:0]/0x4901)
∗ **TMHSP**: Wait-Timer High-Speed Mode Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D6/0x4900)

Table II.2.8.2.1 Oscillation Stabilization Wait Time at Wakeup

TMHSP	OSCTM[7:0]	Number of clocks	Time
1	0x0	0	0
	0x1	16	800 ns
	0x2	32	1.6 µs
	:	:	:
	0xff	4080	0.204 ms
0	0x0	0	0
	0x1	8192	0.409 ms
	0x2	16384	0.819 ms
	:	:	:
	0xff	2M	104.5 ms

(The time shown here is an example when operating with a 20 MHz OSC3.)

SLEEP control when clock sources are switched over

When the CPU reawakes from SLEEP mode, the clock sources (OSC3 or OSC1) also are switched over depending on how OSCSEL (D2/CMU_SYSCLKCTL register) is set. Before the clock sources can be switched over, the CPU must be placed once in SLEEP mode, then released. Therefore, a function is provided that automatically reawakes the CPU from SLEEP mode without using an interrupt, etc. To use this function, set WAKEUPWT (D4/CMU_SYSCLKCTL register) to 0. (By default, it is set to 0.)

∗ **OSCSEL**: OSC Clock Selection Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D2/0x4900)
∗ **WAKEUPWT**: Wakeup-Wait Function Enable Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D4/0x4900)

When the slp instruction is executed with WAKEUPWT set to 0, the CPU automatically reawakes from SLEEP mode several 10 clock cycles after that time, then restarts with the source clock selected by OSCSEL after the oscillation stabilization time described above has elapsed.
The OSC oscillation start wait timer configured using OSCTM[7:0] and TMHSP is effective even if WAKEUPWT is 0. To restart the CPU in the shortest time possible, set OSCTM[7:0] to 0x0 and TMHSP to 1.
When WAKEUPWT is set to 1, the CPU is reawaken from SLEEP mode by initial reset, RTC interrupt (level triggered), #NMI signal, or other interrupt from an external device (port input interrupt with level triggered).
For details about clock switchover and SLEEP control procedures, see Section II.2.9, “Clock Setup Procedure.”

II.2.8.3 Precautions

Executing the slp and halt instructions

Be sure to execute the slp or halt instruction stored in the IRAM when placing the S1C17501 into standby mode. Executing it in the Flash memory increases current consumption as the Flash memory keeps active status during standby mode. Furthermore, when an interrupt is used to cancel standby mode, the S1C17 Core always executes the instruction that follows the slp or halt instruction before the interrupt is accepted. Therefore, place one nop instruction following the slp or halt instruction to ensure that the interrupt can be generated. If a memory access instruction that makes the S1C17 Core go into wait status is placed, level interrupt signals may not be accepted (see "Interrupt" below) and the interrupt handling may not be executed.

Interrupt

The standby mode is released by an interrupt from the ITC, NMI, or reset. Note that the ITC must be configured so that the interrupt to be used for releasing the standby mode can be generated to the CPU. When the clock has not been supplied to the ITC, the interrupt signal from the interrupt source that has been enabled to interrupt is passed through the ITC and is input to the CMU. This signal is used to release the standby mode and to start supplying clocks. The ITC can operate with the supplied clock in HALT mode, so the cause-of-interrupt flag is set immediately after the interrupt source asserts the interrupt signal and the ITC requests an interrupt to the CPU without a delay. In SLEEP mode, the ITC will be able to set the cause-of-interrupt flag and to request an interrupt to the CPU after the CMU starts supplying the clock to the ITC. Therefore, the delay in the interrupt request to the CPU after waking up from SLEEP mode may cause the CPU to execute a few instructions that follows the slp instruction before the CPU executes the interrupt processing. Moreover, if the interrupt source deasserts the interrupt signal before the CMU starts supplying the clock to the ITC, an interrupt does not occur since the cause-of-interrupt flag is not set.

The IE and IL[2:0] bits in the CPU's PSR register do not affect the releasing of standby mode by an interrupt. For example, by setting the ITC to enable the interrupt used for releasing and setting the IE bit to disable interrupts, the CPU can wake up from SLEEP mode without an interrupt processing.

To ensure that the interrupt handler routine will be executed when a port input interrupt (level interrupt) is used to cancel standby mode, the port input signal must be asserted longer than the time shown below.

(1) When the clock is stopped during SLEEP mode
OSC3 oscillation start time + OSC3 oscillation stabilization wait time (set by the user) + 10 system clock cycles

(2) When the clock is not stopped during SLEEP mode, or in HALT mode
10 system clock cycles

Oscillator circuits

When OSC3 oscillation is set to stop during SLEEP mode, the OSC3 oscillator circuit starts oscillating upon exiting SLEEP mode. This is because the OSC3 oscillator circuit requires a finite time before its oscillation stabilizes after starting operation. To restart the CPU using the OSC3 as the source clock, OSCTM[7:0] (D[7:0]/CMU_OSC3_WCNT register) and TMHSP (D6/CMU_SYSCLKCTL register) must be properly set so that the CPU starts operating after this oscillation stabilization time elapses. The oscillation start time of the OSC3 oscillator circuit varies with the device used, board patterns, and operating environment. Therefore, the set time above should have a sufficient allowance.

Switching over the clock sources

Use the automatic SLEEP cancellation function when executing the slp instruction for switching over the clock sources. When the SLEEP mode is cancelled, the OSC oscillation start wait timer that has been configured using OSCTM[7:0] and TMHSP starts operating with the clock source after switch over. Use the switched clock frequency for calculating the oscillation wait time.

Other

The core CPU register contents are retained even during standby mode. In SLEEP mode, the input/output pins keep the status at the time the S1C17501 enters SLEEP mode. Also in HALT mode, the input/output pins keep the status at the time the S1C17501 enters HALT mode. However, some input/output pin statuses may be changed according to the module operation if the CMU has been configured to supply the module operating clock in HALT mode.

II.2.9 Clock Setup Procedure

This section describes the procedure for setting up clocks or altering clock settings.
When initially reset, the clocks are set to the following states:

OSC3 oscillator circuit: On
OSC1 oscillator circuit: On
System clock source: OSC3
System clock: OSC3•1/1
CMU_CLK: OSC3•1/1

II.2.9.1 Changing the Clock Source from OSC3 to OSC1, then Turning Off OSC3

1. CMU Write Protect Register (0x4920) = 0x96
   Disable write protection of the CMU registers.
2. OSCSEL (D2/0x4900) = 1
   Select OSC1 for the clock source.
3. Setting the OSC3 Wait Timer Register (0x4901) and System Clock Control Register (0x4900)
   • OSCTM[7:0] (D[7:0]/0x4901) = *
   • OSC3OFF (D7/0x4900) = 0
   • TMHSP (D6/0x4900) = *
   • WAKEUPWT (D4/0x4900) = 0

   * Set appropriate values so that the wait timer exceeds the stabilization time of OSC1 oscillation (e.g., 3 seconds in the S1C17501). Be aware that the wait timer operates with the OSC1 clock. For details about the OSC1 oscillation start time, see "Electrical Characteristics."

   This setting causes the CPU to automatically exit SLEEP mode and restart after the set time has passed without waiting for an interrupt.
4. Stop any peripheral circuits that are operating.
5. Execute the slp instruction.
   The chip enters SLEEP mode and the CMU temporarily stops clock output. The CPU automatically reawakens from SLEEP mode after the set time has passed from execution of the slp instruction, and restarts using OSC1 as the clock source.
6. SOSC3 (D1/0x4900) = 0
   Turn off the OSC3 oscillator circuit.
7. Newly setting the CMU registers again
   Newly alter the CMU_CLK settings, and set other CMU registers again, as required.
8. CMU Write Protect Register (0x4920) = other than 0x96
   Reenable write protection of the CMU registers.

II.2.9.2 Changing the Clock Source from OSC1 to OSC3

1. CMU Write Protect Register (0x4920) = 0x96
   Disable write protection of the CMU registers.
2. SOSC3 (D1/0x4900) = 1
   Turn on the OSC3 oscillator circuit if turned off.
3. OSCSEL (D2/0x4900) = 0
   Select OSC3 for the clock source.
4. Setting the OSC3 Wait Timer Register (0x4901) and System Clock Control Register (0x4900)
   • OSCTM[7:0] (D[7:0]/0x4901) = *
   • OSC3OFF (D7/0x4900) = 0
   • TMHSP (D6/0x4900) = *
   • WAKEUPWT (D4/0x4900) = 0

   * Set appropriate values so that the wait timer exceeds the stabilization time of OSC3 oscillation (e.g., 25 ms in the S1C17501). Be aware that the wait timer operates with the OSC3 clock. For details about the OSC3 oscillation start time, see "Electrical Characteristics."

   This setting causes the CPU to automatically exit SLEEP mode and restart after the set time has passed without waiting for an interrupt.
5. Stop any peripheral circuits that are operating, except the RTC.
6. Execute the slp instruction.
   The chip enters SLEEP mode and the CMU temporarily stops clock output. The CPU automatically reawakens from SLEEP mode after the set time has passed from execution of the slp instruction, and restarts using OSC3 as the clock source.
7. Newly setting the clock control registers again
   Newly alter the system clock or CMU_CLK settings, and set other CMU registers newly again, as required.
8. CMU Write Protect Register (0x4920) = other than 0x96
   Reenable write protection of the CMU registers.

II.2.9.3 Turning Off OSC3 during SLEEP

To turn off OSC3 oscillation during SLEEP mode when operating with OSC3 as the clock source, follow the control procedure described below.

1. CMU Write Protect Register (0x4920) = 0x96
   Disable write protection of the CMU registers.
2. Setting the OSC3 Wait Timer Register (0x4901) and System Clock Control Register (0x4900)
   • OSCTM[7:0] (D[7:0]/0x4901) and TMHSP (D6/0x4900)
   Set the wait time until the oscillation stabilizes after exiting SLEEP mode.
   Example: TMHSP = 1, OSCTM[7:0] = 0x40 (wait time = about 26 ms when OSC3 = 20 MHz)
   • OSC3OFF (D7/0x4900) = 1
   Turn off OSC3 oscillation when in SLEEP mode.
   • WAKEUPWT (D4/0x4900) = 1
   Set the CPU to awake from SLEEP mode by using an RTC interrupt (level triggered), #NMI signal, or other interrupt from an external device (port input interrupt with level triggered).
3. CMU Write Protect Register (0x4920) = other than 0x96
   Reenable write protection of the CMU registers.
4. Stop any peripheral circuits that are operating, except the RTC.
5. Execute the slp instruction.
   The chip enters SLEEP mode and the CMU temporarily stops clock output.

   The CPU is brought out of SLEEP mode by an RTC interrupt (level triggered), forced break from the debugger, #NMI signal, or other interrupt from an external device (port input interrupt with level triggered), and it restarts using the clock source (OSC3) selected with OSCSEL (D2/0x4900).

II.2.9.4 SLEEP Keeping Oscillation On (without Clock Change)

To enter SLEEP mode without a clock source change and turning off the oscillation, follow the control procedure described below. This is the control to reduce power consumption as much as possible by stopping the core and peripheral functions, with no restart time penalty.

1. CMU Write Protect Register (0x4920) = 0x96
   Disable write protection of the CMU registers.
2. Setting the OSC3 Wait Timer Register (0x4901) and System Clock Control Register (0x4900)
   • OSCTM[7:0] (D[7:0]/0x4901) = 0x0
   • OSC3OFF (D7/0x4900) = 0
   • TMHSP (D6/0x4900) = 1
   • WAKEUPWT (D4/0x4900) = 1
   This setting causes the CPU to exit SLEEP mode using an RTC interrupt (level triggered), #NMI signal, or other interrupt from an external device (port input interrupt with level triggered), and to restart in the shortest time possible (several 10 clock cycles).
3. CMU Write Protect Register (0x4920) = other than 0x96
   Reenable write protection of the CMU registers.
4. Stop any peripheral circuits that are operating, except the RTC.
5. Execute the slp instruction.
   The chip enters SLEEP mode and the CMU temporarily stops clock output.

   The CPU is brought out of SLEEP mode by an RTC interrupt (level triggered), forced break from the debugger, #NMI signal, or other interrupt from an external device (port input interrupt with level triggered), and it restarts using the clock source selected with OSCSEL (D2/0x4900).

II.2.10 Noise Filters

II.2.10.1 Noise Filter for DSIN Input

If the DSIO signal becomes active due to noise, the S1C17 Core suspends the program execution and enters debug mode. To avoid this, the S1C17501 incorporates a noise filter that operates with the system clock to remove noise from the signal before it is input to the S1C17 Core.
When using this noise filter, set DSINNF (D4/CMU_NF register) to 1. When DSINNF is set to 0 (default), the DSIO signal bypasses the noise filter.

* **DSINNF**: DSIO Input Noise Filter Enable Bit in the Noise Filter Control (CMU_NF) Register (D4/0x4902)

II.2.10.2 Noise Filters for Input Ports

The CMU module provides the noise filters to remove noise on the signals input from the ports shown below.
SPI: SPI_SDI0, SPI_SCK0, SPI_SDI1, SPI_SCK1
I²S: I2S_MCLK0, I2S_SDI1, I2S_WS1, I2S_SCK1, I2S_MCLK1
UART: #SCLK0
SRAMC: #WAIT
I²C: I2C_SDA, I2C_SCL
CG_T16U0: EXCL0

When using these noise filters, set INPORTNF (D1/CMU_NF register) to 0. When INPORTNF is set to 1 (default), the input signals bypass the noise filters.

* **INPORTNF**: Input Port Noise Filter Enable Bit in the Noise Filter Control (CMU_NF) Register (D1/0x4902)

Notes:
- These noise filters cannot be enabled individually.
- The noise filters are not effective if these ports are used as general-purpose input port.

II.2.10.3 Noise Filter for OSC3 Clock Input

To stabilize the operation when an external clock is input to the OSC3 pin, the CMU provides a noise filter to remove noise from the input clock.
When using this noise filter, set OSC3NF (D0/CMU_NF register) to 0. When OSC3NF is set to 1 (default), the input clock bypasses the noise filter.

* **OSC3NF**: OSC3 Input Noise Filter Enable Bit in the Noise Filter Control (CMU_NF) Register (D0/0x4902)

II.2.11 USB Wait Control

The USB Wait Control Register (0x4909) contains the control bits USBWT[2:0] (D[2:0]) used to set the number of wait cycles to be inserted when accessing the USB registers.

* **USBWT[2:0]**: USB Register Access Wait Control Bits in the USB Wait Control (CMU_USBWT) Register (D[2:0]/0x4909)

Table II.2.11.1 Number of Wait Cycles during USB Access

USBWT[2:0]	Number of wait cycles (in units of system clock cycles)	System clock frequency
0x7	7 cycles	Reserved
0x6	6 cycles	48 MHz or less
0x5	5 cycles	45 MHz or less
0x4	4 cycles	36 MHz or less
0x3	3 cycles	24 MHz or less
0x2	2 cycles	16 MHz or less
0x1	1 cycle	8 MHz or less
0x0	0 cycles	8 MHz or less

(Default: 0x7 = 7 cycles)

The number of wait cycles should be set according to the system clock frequency.

Also the USB Wait Control Register (0x4909) contains the USBSNZ bit (D5) that controls Snooze mode for the USB function controller. Setting USBSNZ to 1 enables Snooze mode.

* **USBSNZ**: USB Snooze Control Bit in the USB Wait Control (CMU_USBWT) Register (D5/0x4909)

Refer to Section X.1, “USB Function Controller,” for details on control of the USB function controller.

II.2.12 Details of Control Registers

Table II.2.12.1 List of CMU Registers

Address	Register name		Function
0x4900	CMU_SYSCLKCTL	System Clock Control Register	Controls the system clock.
0x4901	CMU_OSC3_WCNT	OSC3 Wait Timer Register	Sets the OSC3 wait timer for system wake-up.
0x4902	CMU_NF	Noise Filter Control Register	Enables noise filters.
0x4903	CMU_OSC3DIV	OSC3 Clock Divider Register	Selects a OSC3 system clock frequency.
0x4905	CMU_CMUCLK	CMU_CLK Select Register	Selects the output CMU_CLK frequency.
0x4906	CMU_GATEDCLK0	Gated Clock Control 0 Register	Controls clock supply to peripheral modules.
0x4907	CMU_GATEDCLK1	Gated Clock Control 1 Register	
0x4908	CMU_GATEDCLK2	Gated Clock Control 2 Register	
0x4909	CMU_USBWT	USB Wait Control Register	Sets the wait cycles for accessing the USB registers.
0x4920	CMU_PROTECT	CMU Write Protect Register	Enables writing to the CMU registers (0x4900–0x4909).

The following describes each CMU control register.
The CMU control registers are mapped as an 8-bit device at addresses 0x4900 to 0x4920, and can be accessed in units of 16 bits or bytes.

Note: The CMU registers (0x4900–0x4909) are write-protected. Before these register can be rewritten, write protection must be removed by writing data 0x96 to the CMU Write Protect Register (0x4920). Note that since unnecessary rewrites to addresses 0x4900–0x4909 could lead to erratic system operation, the CMU Write Protect Register (0x4920) should be set to other than 0x96 unless said CMU registers must be rewritten.

0x4900: System Clock Control Register (CMU_SYSCLKCTL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
System Clock Control Register (CMU_SYSCLKCTL)	0x4900 (8 bits)	D7	OSC3OFF	OSC3 disable during SLEEP	1	Stop	0	Run	0	R/W	
		D6	TMHSP	Wait-timer high-speed mode	1	High speed	0	Normal	0	R/W	
		D5	–	reserved	–				–	–	0 when being read.
		D4	WAKEUPWT	Wakeup-wait function enable	1	Wait interrupt	0	No wait	0	R/W	
		D3	–	reserved	–				–	–	0 when being read.
		D2	OSCSEL	OSC clock selection	1	OSC1	0	OSC3	0	R/W	
		D1	SOSC3	OSC3 oscillator on/off	1	On	0	Off	1	R/W	
		D0	–	reserved	–				–	–	0 when being read.

D7 OSC3OFF: OSC3 Disable During SLEEP Bit

Selects whether to turn off the OSC3 oscillator circuit during SLEEP mode.
1 (R/W): Stop
0 (R/W): Operating (default)

Continue operating OSC3 when entering SLEEP mode to switch over the clock sources (OSC), or turn it off when entering SLEEP mode for power-down purposes.

D6 TMHSP: Wait-Timer High-Speed Mode Bit

Sets count mode for the oscillation stabilization wait timer (CMU_OSC3_WCNT register).
1 (R/W): High-speed mode
0 (R/W): Normal mode (default)

The oscillation stabilization wait timer counts from 0 to 2M in units of 8,192 system clock cycles during normal mode, or from 0 to 4,080 in units of 16 system clock cycles during high-speed mode. Select either mode in which the OSC3 oscillation start time can be secured with the OSC frequency used.

D5 Reserved

D4 WAKEUPWT: Wakeup-Wait Function Enable Bit

Enables the SLEEP mode wakeup-wait function used for switching over the clocks.
1 (R/W): Wait an interrupt
0 (R/W): No wait (default)

When the slp instruction is executed while WAKEUPWT is set to 0, the CPU automatically reawakes from SLEEP mode several 10 clock cycles after instruction execution, and restarts with the source clock selected by OSCSEL (D2). Since even in this case the oscillation stabilization wait time set by OSCTM[7:0] (D[7:0]/CMU_OSC3_WCNT register) is effective, OSCTM[7:0] should be set to 0x0 when clocks must be switched over in the shortest time possible.
When WAKEUPWT is set to 1, the CPU can only be reawaken from SLEEP mode by initial reset, RTC interrupt (level triggered), forced break from the debugger, #NMI signal, and other interrupt from an external source (port input interrupt with level triggered).

D3 Reserved

D2 OSCSEL: OSC Clock Selection Bit

Selects the system clock source (OSC).
1 (R/W): OSC1
0 (R/W): OSC3 (default)

The clock sources changed here are not switched over immediately, but are actually switched over upon returning from SLEEP mode. Therefore, the CPU must be placed in SLEEP mode after setting up OSCSEL.

Note: When the clock source is changed, the clock control registers must be set so that the CMU is supplied with a clock from the selected clock source upon returning from SLEEP mode immediately after the change. Otherwise, the chip does not restart after return from SLEEP mode.

D1 SOSC3: OSC3 Oscillator On/Off Bit

Turns the OSC3 oscillator circuit on or off.
1 (R/W): On (default)
0 (R/W): Off

Note: When SOSC3 is set from 0 to 1 for initiating oscillation by the oscillator, a finite time is required until the oscillation stabilizes. To prevent erratic operation, do not use the oscillator-derived clock until the oscillation start time stipulated in the electrical characteristics table elapses.

D0 Reserved

0x4901: OSC3 Wait Timer Register (CMU_OSC3_WCNT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
OSC3 Wait Timer Register (CMU_OSC3_WCNT)	**0x4901** (8 bits)	D7–0	**OSCTM[7:0]**	OSC oscillation stabilization-wait timer	0–255	0x0	R/W	

D[7:0] OSCTM[7:0]: OSC Oscillation Stabilization-Wait Timer Bits

Sets an oscillation stabilization wait time during which the CPU is kept waiting before it starts operating upon returning from SLEEP mode. This wait time can be set in increments of 16 system clock cycles when TMHSP (D6/CMU_SYSCLKCTL register) = 1, or 8,192 clock cycles when TMHSP = 0. (Default: 0x0 = no wait time)

Table II.2.12.2 Oscillation Stabilization Wait Time at Wakeup

TMHSP	OSCTM[7:0]	Number of clocks	Time
1	0x0	0	0
	0x1	16	800 ns
	0x2	32	1.6 µs
	:	:	:
	0xff	4080	0.204 ms
0	0x0	0	0
	0x1	8192	0.409 ms
	0x2	16384	0.819 ms
	:	:	:
	0xff	2M	104.5 ms

(The time shown here is an example when operating with a 20 MHz OSC3.)

When the OSC3 oscillation is to be turned off during SLEEP mode, make sure the wait time set by these bits is equal to or greater than the OSC3 oscillation start time stipulated in the electrical characteristics table.

Note: The OSC oscillation start wait timer operates with the operating clock activated after the SLEEP mode is released. Therefore, use the switched clock frequency for calculating the oscillation wait time to be set to OSCTM[7:0] when executing the slp instruction for switching over the clock sources.

0x4902: Noise Filter Control Register (CMU_NF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Noise Filter Control Register (CMU_NF)	0x4902 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	DSINNF	DSIO input noise filter enable	1 Enable 0 Disable	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	INPORTNF	Input port noise filter enable	1 Disable 0 Enable	1	R/W	
		D0	OSC3NF	OSC3 input noise filter enable	1 Disable 0 Enable	1	R/W	

D[7:5] Reserved

D4 DSINNF: DSIO Input Noise Filter Enable Bit

Enables/disables the noise filter for the DSIO input.
1 (R/W): Enable (reject noise)
0 (R/W): Disable (bypass) (default)

When using this noise filter, set DSINNF to 1. When DSINNF is set to 0 (default), the DSIO signal bypasses the noise filter.

D[3:2] Reserved

D1 INPORTNF: Input Port Noise Filter Enable Bit

Enables/disables the noise filters for the input ports.
1 (R/W): Disable (bypass) (default)
0 (R/W): Enable (reject noise)

The CMU module provides the noise filters to remove noise on the signals input from the ports shown below.

SPI: SPI_SDI0, SPI_SCK0, SPI_SDI1, SPI_SCK1
I²S: I2S_MCLK0, I2S_SDI1, I2S_WS1, I2S_SCK1, I2S_MCLK1
UART: #SCLK0
SRAMC: #WAIT
I²C: I2C_SDA, I2C_SCL
CG_T16U0: EXCL0

When using these noise filters, set INPORTNF to 0. When INPORTNF is set to 1 (default), the input signals bypass the noise filters.

D0 OSC3NF: OSC3 Input Noise Filter Enable Bit

Enables/disables the noise filter for the OSC3 external clock input.
1 (R/W): Disable (bypass) (default)
0 (R/W): Enable (reject noise)

When using this noise filter, set OSC3NF to 0. When OSC3NF is set to 1 (default), the input clock bypasses the noise filter.

0x4903: OSC3 Clock Divider Register (CMU_OSC3DIV)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
OSC3 Clock Divider Register (CMU_OSC3DIV)	0x4903 (8 bits)	D7–3	–	reserved	–		–	–	0 when being read.
		D2–0	OSC3DIV [2:0]	OSC3 clock divider selection	OSC3DIV[2:0]	Divider	0x0	R/W	
					0x7	OSC3•1/1			
					0x6	OSC3•1/1			
					0x5	OSC3•1/32			
					0x4	OSC3•1/16			
					0x3	OSC3•1/8			
					0x2	OSC3•1/4			
					0x1	OSC3•1/2			
					0x0	OSC3•1/1			

D[7:3] Reserved

D[2:0] OSC3DIV[2:0]: OSC3 Clock Divider Selection Bits

OSC3DIV[2:0] is used to select the system clock frequency when OSC3 is selected as the system clock source. It is derived from the OSC3 clock by dividing its frequency by a given value. Use OSC3DIV[2:0] to select this clock divide ratio.

Table II.2.12.3 Selecting an OSC3 Divided Clock

OSC3DIV[2:0]	OSC3_DIV clock
0x7	OSC3•1/1
0x6	OSC3•1/1
0x5	OSC3•1/32
0x4	OSC3•1/16
0x3	OSC3•1/8
0x2	OSC3•1/4
0x1	OSC3•1/2
0x0	OSC3•1/1

(Default: 0x0 = OSC3•1/1)

A divided clock can be selected at any time. However, up to 32 OSC3 clock cycles are required before the clocks are actually changed after altering the register values.

0x4905: CMU_CLK Select Register (CMU_CMUCLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
CMU_CLK Select Register (CMU_CMUCLK)	**0x4905** (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3–0	**CMU_ CLKSEL[3:0]**	CMU_CLK selection	CMU_CLKSEL[3:0]	Clock source	0x0	R/W	
					0xf–0xa	reserved			
					0x9	OSC3•1/32			
					0x8	OSC3•1/16			
					0x7	OSC3•1/8			
					0x6	OSC3•1/4			
					0x5	OSC3•1/2			
					0x4	OSC3•1/1			
					0x3	LCDC_CLK			
					0x2	BCLK			
					0x1	OSC1			
					0x0	OSC3			

D[7:4] Reserved

D[3:0] CMU_CLKSEL[3:0]: CMU_CLK Selection Bits

CMU_CLK is the clock for the external bus. It can be selected from the 10 clocks listed in Table II.2.12.5.

Table II.2.12.5 Selecting CMU_CLK

CMU_CLKSEL[3:0]	CMU_CLK
0xf–0xa	Reserved
0x9	OSC3•1/32
0x8	OSC3•1/16
0x7	OSC3•1/8
0x6	OSC3•1/4
0x5	OSC3•1/2
0x4	OSC3•1/1
0x3	LCDC_CLK
0x2	BCLK
0x1	OSC1
0x0	OSC3

(Default: 0x0 = OSC3)

CMU_CLK can be selected at any time. However, switching over the clocks creates hazards. When CMU_CLK must be output to external devices, it is also necessary to select a port function. For details on how to control clock output and about the port to be used, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

Note: Other settings than that listed in Table II.2.12.5 are reserved for testing. Do not set undescribed values to CMU_CLKSEL[3:0] as undesired clocks may output.

0x4906: Gated Clock Control 0 Register (CMU_GATEDCLK0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Gated Clock Control 0 Register (CMU_GATEDCLK0)	0x4906 (8 bits)	D7	FLASHC_CLK_EN	FLASHC clock control (in HALT mode)	1 On 0 Off	1	R/W	
		D6	–	reserved	–	–	–	0 when being read.
		D5	USB_SAPB_CLK_EN	USB SAPB I/F clock control	1 On 0 Off	0	R/W	
		D4	USB_CLK_EN	USB IP 48 MHz clock control		0	R/W	
		D3-1	–	reserved	–	–	–	0 when being read.
		D0	PCLK_EN	Core peripheral clock control	1 On 0 Off	1	R/W	

D7 FLASHC_CLK_EN: FLASHC Clock Control (in HALT mode) Bit

Controls clock (FLASHC_CLK) supply to the FLASHC in HALT mode.
1 (R/W): On (default)
0 (R/W): Off

D6 Reserved

D5 USB_SAPB_CLK_EN: USB SAPB I/F Clock Control Bit

Controls clock (USB_SAPB_CLK) supply to the USB SAPB bus interface.
1 (R/W): On
0 (R/W): Off (default)

D4 USB_CLK_EN: USB IP 48 MHz Clock Control Bit

Controls clock (OSC3 = 48 MHz) supply to the USB module.
1 (R/W): On
0 (R/W): Off (default)

D3–1 Reserved

D0 PCLK_EN: Core Peripheral Clock Control Bit

Controls clock (PCLK) supply to the core peripheral modules.
1 (R/W): On (default)
0 (R/W): Off

0x4907: Gated Clock Control 1 Register (CMU_GATEDCLK1)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Gated Clock Control 1 Register (CMU _GATEDCLK1)	**0x4907** (8 bits)	D7	**SRAMC _CLK_EN**	SRAMC clock control (in HALT mode)	1	On	0	Off	1	R/W	
		D6–2	**–**	reserved	–				–	–	0 when being read.
		D1	**PT8_CLK _EN**	8-bit programmable timer clock control	1	On	0	Off	1	R/W	
		D0	**MFT_CLK _EN**	Multi-function timer clock control					1	R/W	

D7 SRAMC_CLK_EN: SRAMC Clock Control (in HALT mode) Bit

Controls clock (SRAMC_CLK) supply to the SRAMC in HALT mode.
1 (R/W): On (default)
0 (R/W): Off

D[6:2] Reserved

D1 PT8_CLK_EN: 8-bit Programmable Timer Clock Control Bit

Controls clock (PT8_CLK) supply to the 8-bit programmable timer.
1 (R/W): On (default)
0 (R/W): Off

D0 MFT_CLK_EN: Multi-Function Timer Clock Control Bit

Controls clock (MFT_CLK) supply to the multi-function timer.
1 (R/W): On (default)
0 (R/W): Off

0x4908: Gated Clock Control 2 Register (CMU_GATEDCLK2)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Gated Clock Control 2 Register (CMU_GATEDCLK2)	0x4908 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	SPI_CLK_EN	SPI CH.1 module clock control	1 On 0 Off	1	R/W	
		D4	REMC_CLK_EN	REMC module clock control		1	R/W	
		D3	ADC_CLK_EN	ADC module clock control		1	R/W	
		D2	WDT_CLK_EN	WDT module clock control		1	R/W	
		D1	PORT_CLK_EN	I/O port module clock control		1	R/W	
		D0	RTC_SAPB_CLK_EN	RTC SAPB I/F clock control		1	R/W	

D[7:6] Reserved

D5 SPI_CLK_EN: SPI CH.1 Module Clock Control Bit

Controls clock (SPI_CLK) supply to the SPI module.
1 (R/W): On (default)
0 (R/W): Off

D4 REMC_CLK_EN: REMC Module Clock Control Bit

Controls clock (REMC_CLK) supply to the remote controller module.
1 (R/W): On (default)
0 (R/W): Off

D3 ADC_CLK_EN: ADC Module Clock Control Bit

Controls clock (ADC_CLK) supply to the A/D converter module.
1 (R/W): On (default)
0 (R/W): Off

D2 WDT_CLK_EN: WDT Module Clock Control Bit

Controls clock (WDT_CLK) supply to the watchdog timer module.
1 (R/W): On (default)
0 (R/W): Off

D1 PORT_CLK_EN: I/O Port Module Clock Control Bit

Controls clock (PORT_CLK) supply to the I/O port module.
1 (R/W): On (default)
0 (R/W): Off

D0 RTC_SAPB_CLK_EN: RTC SAPB I/F Clock Control Bit

Controls clock (RTC_SAPB_CLK) supply to the RTC SAPB interface.
1 (R/W): On (default)
0 (R/W): Off

0x4909: USB Wait Control Register (CMU_USBWT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USB Wait Control Register (CMU_USBWT)	0x4909 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	USBSNZ	USB snooze control	1 Enable 0 Disable	0	R/W	
		D4–3	–	reserved	–	–	–	0 when being read.
		D2–0	USBWT[2:0]	USB register access wait control	USBWT[2:0] Wait cycle 0x7 7 cycles : : 0x0 0 cycles	0x7	R/W	

D[7:6] Reserved

D5 USBSNZ: USB Snooze Control Bit

This bit enables/disables the USB snooze control.

1 (R/W): Enable

0 (R/W): Disable (default)

When this bit is set to 1, the USB controller performs a transition sequence and then it enters Snooze mode. When this bit is set to 0, the USB controller resumes operating. For details of the snooze sequence, see Section X.1.4.3, "Snooze."

D[4:3] Reserved

D[2:0] USBWT[2:0]: USB Register Access Wait Control Bits

These bits set the number of wait cycles to be inserted when accessing the USB control register.

Table II.2.12.6 Number of Wait Cycles during USB Access

USBWT[2:0]	Number of wait cycles (in units of system clock cycles)	System clock frequency
0x7	7 cycles	Reserved
0x6	6 cycles	48 MHz or less
0x5	5 cycles	45 MHz or less
0x4	4 cycles	36 MHz or less
0x3	3 cycles	24 MHz or less
0x2	2 cycles	16 MHz or less
0x1	1 cycle	8 MHz or less
0x0	0 cycles	8 MHz or less

(Default: 0x7 = 7 cycles)

The number of wait cycles should be set according to the system clock frequency.

0x4920: CMU Write Protect Register (CMU_PROTECT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CMU Write Protect Register (CMU _PROTECT)	**0x4920** (8 bits)	D7–0	**CLGP[7:0]**	CMU register protect flag	Writing 10010110 (0x96) removes the write protection of the CMU registers (0x4900–0x4909). Writing another value set the write protection.	0x0	R/W	

D[7:0] CLGP[7:0]: CMU Register Protect Flag Bits

Enables/disables write protection of the CMU registers (0x4900–0x4909).

0x96 (R/W): Disable write protection

Other than 0x96 (R/W): Write-protect the register (default: 0x0)

Before altering any CMU register, write data 0x96 to the register to disable write protection. If this register is set to other than 0x96, even if an attempt is made to alter any CMU register by executing a write instruction, the content of said register will not be altered even though the instruction may have been executed without a problem. Once this register is set to 0x96, the CMU registers can be rewritten any number of times until being reset to other than 0x96. When rewriting the CMU registers has finished, this register should be set to other than 0x96 to prevent accidental writing to the CMU registers.

II.2.13 Precautions

Precautions regarding clock control

- The CMU registers (0x4900–0x4909) are write-protected. Before these registers can be rewritten, write protection must be removed by writing data 0x96 to the CMU Write Protect Register (0x4920). Once write protection is removed, the CMU registers can be written to any number of times until the protect register is reset to other than 0x96. Note that since unnecessary rewriting of the CMU registers could lead to erratic system operation, the CMU Write Protect Register (0x4920) should be set to other than 0x96 unless the CMU registers must be rewritten.
- When the clock source is changed, the CMU registers must be set so that the CMU is supplied with a clock from the selected clock source upon returning from SLEEP mode immediately after the change. Otherwise, the chip may not restart after return from SLEEP mode.
  Furthermore, note that the timer, which generates an oscillation stabilization wait time after the SLEEP mode is released, operates with the clock after switching over. Be sure to use the correct clock frequency for calculating the wait time to be set to OSCTM[7:0] (D[7:0]/CMU_OSC3_WCNT register) and TMHSP (D6/CMU_SYSCLKCTL register).
- When SOSC3 (D1/CMU_SYSCLKCTL register) is set from 0 to 1 for initiating oscillation by the oscillator, a finite time is required until the oscillation stabilizes (e.g., 25 ms in the S1C17501). To prevent erratic operation, do not use the oscillator-derived clock until the oscillation start time stipulated in the electrical characteristics table elapses.

Precautions regarding reset input

- Even if the #RESET pin is pulled low (= 0), the chip may not be reset unless supplied with a clock. To reset the chip for sure, #RESET should be held low for at least 3 OSC3 clock cycles. However, the input/output port pins will be initialized by reset regardless of whether the chip is supplied with a clock.
- The oscillation start time of the OSC3 oscillator circuit varies with the device used, board patterns, and operating environment. Therefore, a sufficient time should be provided before the reset signal is deasserted.

Precautions regarding NMI input

NMI cannot be nested. The CPU keeps NMI input masked out until the reti instruction is executed after an NMI exception occurred.

II.3 Prescaler (PSC)

II.3.1 Configuration of the Prescaler

The S1C17501 incorporates a prescaler for generating the source clock of the clock generator that generates the operating clocks for the UART (CH.0), SPI (CH.0) and I²C modules. The prescaler divides the PCLK clock, which is supplied from the CMU, by 1 to 16K to generate 15 clocks with different frequencies. A clock select register is provided for each destination peripheral module allowing selection of a prescaler output clock as the count clock.

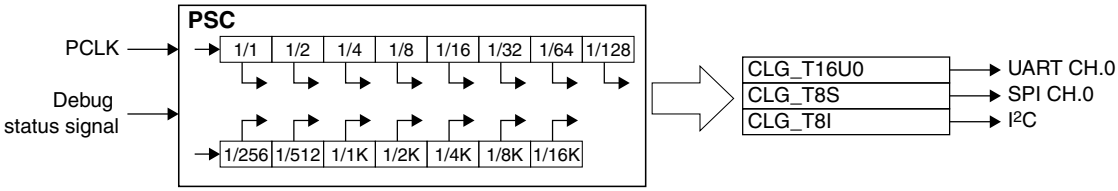


Figure II.3.1.1 Prescaler

The prescaler is controlled by the PRUN bit (D0/PSC_CTL register). Write 1 to PRUN to run the prescaler and write 0 to stop the prescaler. When the clock generator and interface modules are idle, stop the prescaler to reduce current consumption. At initial reset, the prescaler stops operating.

* **PRUN**: Prescaler Run/Stop Control Bit in the Prescaler Control (PSC_CTL) Register (D0/0x4020)

Note: Supply PCLK from the CMU before the prescaler can be used.

The prescaler provides one more control bit PRUND (D1/PSC_CTL register). This bit is used to specify the prescaler operation in debug mode. If PRUND is set to 1, the prescaler operates in debug mode. If PRUND is set to 0, the prescaler stops operating when the S1C17 Core enters debug mode. Set PRUND to 1 when using the clock generator and interface modules in debug mode.

* **PRUND**: Prescaler Run/Stop in Debug Mode Bit in the Prescaler Control (PSC_CTL) Register (D1/0x4020)

II.3.2 Details of Control Register

Table II.3.2.1 Prescaler Register

Address	Register name		Function
0x4020	PSC_CTL	Prescaler Control Register	Starts/stops the prescaler.

The prescaler register is an 8-bit register.

Note: When setting the register, be sure to write a 0, and not a 1, for all "reserved bits."

0x4020: Prescaler Control Register (PSC_CTL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Prescaler Control Register (PSC_CTL)	**0x4020** (8 bits)	D7–2	–	reserved	–				–	–	0 when being read.
		D1	**PRUND**	Prescaler run/stop in debug mode	1	Run	0	Stop	0	R/W	
		D0	**PRUN**	Prescaler run/stop control	1	Run	0	Stop	0	R/W	

D[7:2] Reserved

D1 PRUND: Prescaler Run/Stop in Debug Mode Bit

Selects the prescaler operation in debug mode.
1 (R/W): Run
0 (R/W): Stop (default)

If PRUND is set to 1, the prescaler operates in debug mode. If PRUND is set to 0, the prescaler stops operating when the S1C17 Core enters debug mode. Set PRUND to 1 when using the clock generator and interface modules in debug mode.

D0 PRUN: Prescaler Run/Stop Control Bit

Runs/stops the prescaler.
1 (R/W): Run
0 (R/W): Stop (default)

Write 1 to PRUN to run the prescaler and write 0 to stop the prescaler. When the clock generator and interface modules are idle, stop the prescaler to reduce current consumption.

II.3.3 Precaution

Supply PCLK from the CMU before the prescaler can be used.

II.4 Clock Generator (CLG)

II.4.1 Configuration of the Clock Generator

The S1C17501 is equipped with a clock generator that consists of a 16-bit timer and two 8-bit timers. The 16-bit timer generates the clock for UART CH.0 and two 8-bit timers generate the clock for SPI CH.0 and I²C. The timers count down from the initial value set in the software using a prescaler output clock as the count clock and output an underflow signal when the counter underflows. The underflow signal is used to generate an interrupt and an internal serial interface clock. This allows the application program to get any desired time intervals and programmable serial transfer rates.
Figure II.4.1.1 shows the configuration of the clock generator.

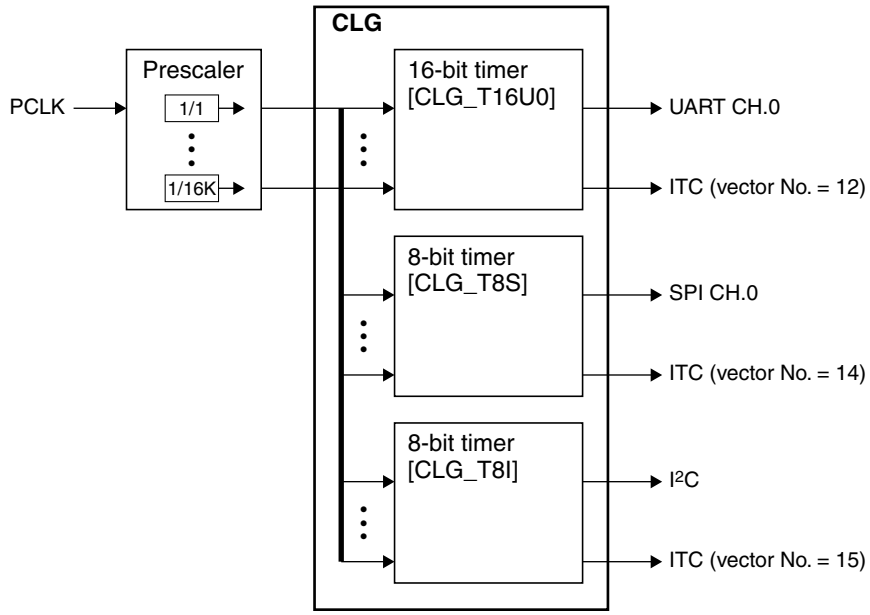


Figure II.4.1.1 Configuration of the Clock Generator

When the serial interface is not used, the timer can be used as a general-purpose programmable timer with an interrupt function.

II.4.2 16-bit Timer (CLG_T16U0)

II.4.2.1 Outline of the 16-bit Timer

The S1C17501 CLG is equipped with a 16-bit timer (CLG_T16U0).
The 16-bit timer includes a 16-bit presettable down counter and a 16-bit reload data register for setting the preset value. The timer counts down from the initial value set in the reload data register and outputs an underflow signal when the counter underflows. The underflow signal is used to generate an interrupt and the serial interface clock for UART CH.0. The underflow period can be programmed by selecting a prescaler clock and setting reload data. This allows the application program to get any desired time intervals and programmable serial transfer rates.
Figure II.4.2.1.1 shows the structure of the 16-bit timer.

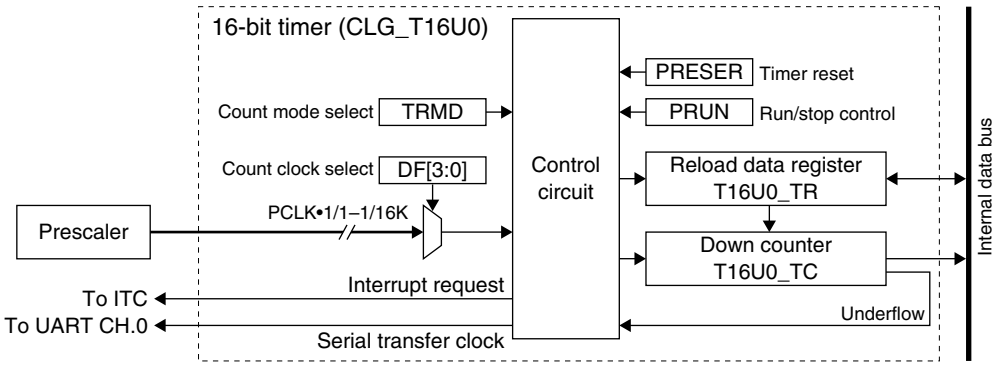


Figure II.4.2.1.1 Structure of 16-bit Timer

II.4.2.2 Count Mode

The 16-bit timer has two count modes: repeat mode and one-shot mode. It can be selected using TRMD (D4/CLG_T16U0_CTL register).

∗ **TRMD**: Count Mode Select Bit in the CLG_T16U0 Control (CLG_T16U0_CTL) Register (D4/0x4206)

Repeat mode (TRMD = 0, default)

The 16-bit timer is set in repeat mode when TRMD is set to 0.
In this mode, the 16-bit timer does not stop after it starts counting until the application program stops the timer. When the counter underflows, the timer presets the reload data register value to the counter and continues counting. The timer outputs the underflow pulses periodically. Set the 16-bit timer in this mode when generating periodical interrupts with a given interval or generating the serial transfer clock.

One-shot mode (TRMD = 1)

The 16-bit timer is set in one-shot mode when TRMD is set to 1.
In this mode, the 16-bit timer automatically stops counting when the counter underflows, so only one interrupt can be generated after starting the timer. When an underflow occurs, the counter is preset with the reload data register value before the timer operation stops. Set the 16-bit timer in this mode when a certain waiting time must be generated.

Note: When setting the count mode, make sure the 16-bit timer counter is stopped.

II.4.2.3 Count Clock

The 16-bit timer uses a prescaler output clock as the count clock. The prescaler divides PCLK by 1 to 16K to generate 15 clocks. Select one of the prescaler output clocks using DF[3:0] (D[3:0]/CLG_T16U0_CLK register).

* **DF[3:0]**: Timer Input Clock Select Bits in the CLG_T16U0 Input Clock Select (CLG_T16U0_CLK) Register (D[3:0]/0x4200)

Table II.4.2.3.1 Selecting the Count Clock

DF[3:0]	Prescaler output clock	DF[3:0]	Prescaler output clock
0xf	Reserved	0x7	PCLK•1/128
0xe	PCLK•1/16384	0x6	PCLK•1/64
0xd	PCLK•1/8192	0x5	PCLK•1/32
0xc	PCLK•1/4096	0x4	PCLK•1/16
0xb	PCLK•1/2048	0x3	PCLK•1/8
0xa	PCLK•1/1024	0x2	PCLK•1/4
0x9	PCLK•1/512	0x1	PCLK•1/2
0x8	PCLK•1/256	0x0	PCLK•1/1

(Default: 0x0)

Notes: • Before the 16-bit timer can start counting, the prescaler must be run.
• When setting the count clock, make sure the 16-bit timer counter is stopped.

For controlling the prescaler, refer to Section II.3, "Prescaler (PSC)."

II.4.2.4 16-bit Timer Reload Register and Underflow Period

The Reload Data (CLG_T16U0_TR) Register (0x4202) is used to set the initial value to the down counter.
The counter initial value set in the reload data register is preset to the down counter when the 16-bit timer is reset or when the counter underflows. When starting the 16-bit timer after resetting, the timer counts down from the reload value. So the reload value and the input clock frequency determine the period of time from starting the timer until an underflow occurs (and between underflows). This makes it possible to obtain a desired wait time, a periodical interrupt interval, or programmable transfer clock for the serial interface.

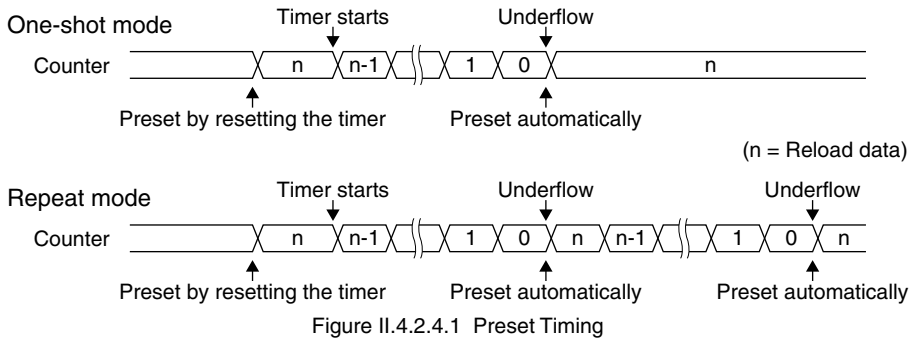


Figure II.4.2.4.1 Preset Timing

The underflow period is calculated by the expression below.

$$\text{Underflow period} = \frac{TR + 1}{clk\_in}\ [s] \qquad \text{Underflow cycle} = \frac{clk\_in}{TR + 1}\ [Hz]$$

clk_in: Count clock (prescaler output clock) frequency [Hz]
TR: Reload data (0–65535)

Note: The UART divides the 16-bit timer output by 16 to generate the sampling clock. Make sure of the division ratio when setting a transfer rate.

II.4.2.5 Resetting the 16-bit Timer

To reset the 16-bit timer, write 1 to PRESER (D1/CLG_T16U0_CTL register). This initializes the counter by presetting the reload data register value.

* **PRESER**: Timer Reset Bit in the CLG_T16U0 Control (CLG_T16U0_CTL) Register (D1/0x4206)

II.4.2.6 16-bit Timer Run/Stop Control

Before starting the 16-bit timer, set up the conditions as shown below.
(1) Select a count mode (one-shot or repeat). See Section II.4.2.2.
(2) Select the count clock (prescaler output clock). See Section II.4.2.3.
(3) Calculate the counter initial value and set it to the reload data register. See Section II.4.2.4.
(4) Reset the timer to preset the initial value to the counter. See Section II.4.2.5.
(5) Set up the interrupt level and enable the interrupt of the timer channel if the timer interrupt is used. See Section II.4.2.8.

To start the 16-bit timer, write 1 to PRUN (D0/CLG_T16U0_CTL register).

* **PRUN**: Timer Run/Stop Control Bit in the CLG_T16U0 Control (CLG_T16U0_CTL) Register (D0/0x4206)

The timer starts counting down from the initial value or the current counter value if the initial value has not been preset. When the counter underflows, the timer outputs an underflow pulse and presets the initial value again. At the same time, an interrupt request is sent to the interrupt controller (ITC).
If the timer is set in one-shot mode, the timer stops counting.
If the timer is set in repeat mode, the timer continues counting from the reloaded initial value.

To stop the 16-bit timer from the application program, write 0 to the PRUN bit. The counter stops counting and holds the current counter value until the timer is reset or restarted. To restart counting from the initial value, reset the timer before writing 1 to the PRUN bit.

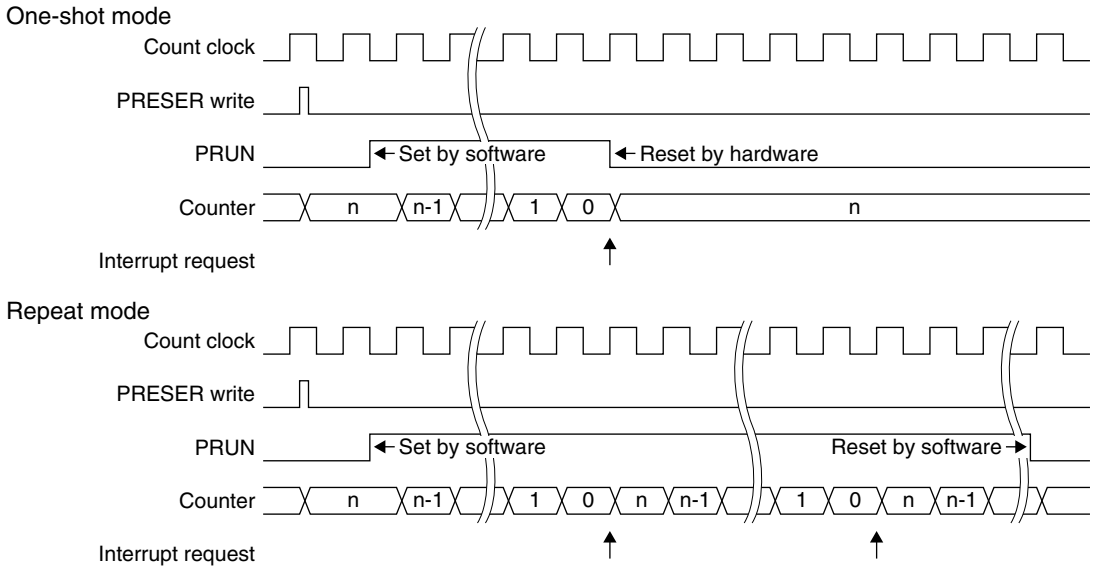


Figure II.4.2.6.1 Count Operation

II.4.2.7 16-bit Timer Output Signal

The 16-bit timer outputs an underflow pulse when the counter underflows.
This pulse is used to request a timer interrupt.
Also this pulse is used to generate the serial transfer clock for UART CH.0.

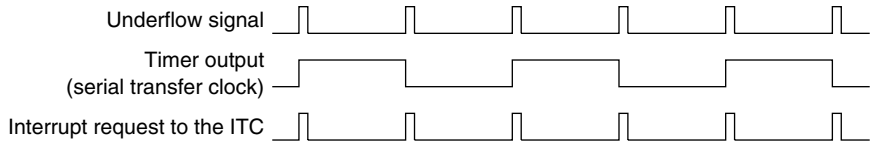


Figure II.4.2.7.1 Timer Output Clock

The reload data register value to obtain a desired transfer rate is calculated by the expression below.

$$bps = \frac{clk\_in}{(TR + 1) \times 16}$$

$$TR = \left(\frac{clk\_in}{bps} - 16\right) \div 16$$

clk_in: Count clock (prescaler output clock) frequency [Hz]
TR: Reload data (0–65535)
bps: Transfer rate (bits/second)

II.4.2.8 16-bit Timer Interrupt

The 16-bit timer outputs an interrupt request signal to the interrupt controller (ITC) when the counter underflows.
To generate a timer underflow interrupt, set up the interrupt level and enable the interrupt using the ITC registers.

ITC registers for timer interrupts

The following shows the control bits of the ITC provided for the 16-bit timer:

Interrupt flag IIFT0
- **IIFT0**: CLG_T16U0 Timer Interrupt Flag Bit in the Interrupt Flag (ITC_IFLG) Register (D8/0x4300)

Interrupt enable bit IIEN0
- **IIEN0**: CLG_T16U0 Timer Interrupt Enable Bit in the Interrupt Enable (ITC_EN) Register (D8/0x4302)

Interrupt level setup bits IILV0
- **IILV0[2:0]**: CLG_T16U0 Timer Interrupt Level Bits in the Internal Interrupt Level Setup (ITC_ILV0) Register 0 (D[2:0]/0x430e)

When an underflow occurs in the timer, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the timer interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the timer underflow pulse, regardless of how the interrupt enable bit is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the timer interrupt.
An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The timer interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vector

The following shows the vector number and vector address for the timer interrupt:

Vector number: 12 (0x0c)
Vector address: TTBR + 0x30

II.4.2.9 Details of Control Registers

Table II.4.2.9.1 List of 16-bit Timer Registers

Address	Register name		Function
0x4200	CLG_T16U0_CLK	CLG_T16U0 Input Clock Select Register	Selects a prescaler output clock.
0x4202	CLG_T16U0_TR	CLG_T16U0 Reload Data Register	Sets reload data.
0x4204	CLG_T16U0_TC	CLG_T16U0 Counter Data Register	Counter data
0x4206	CLG_T16U0_CTL	CLG_T16U0 Control Register	Sets the timer mode and starts/stops the timer.

The following describes each 16-bit timer register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x4200: CLG_T16U0 Input Clock Select Register (CLG_T16U0_CLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
CLG_T16U0 Input Clock Select Register (CLG_T16U0 _CLK)	0x4200 (16 bits)	D15–4	–	reserved	–		–	–	0 when being read.
		D3–0	DF[3:0]	Timer input clock select (Prescaler output clock)	DF[3:0]	Clock	0x0	R/W	
					0xf	reserved			
					0xe	PCLK•1/16384			
					0xd	PCLK•1/8192			
					0xc	PCLK•1/4096			
					0xb	PCLK•1/2048			
					0xa	PCLK•1/1024			
					0x9	PCLK•1/512			
					0x8	PCLK•1/256			
					0x7	PCLK•1/128			
					0x6	PCLK•1/64			
					0x5	PCLK•1/32			
					0x4	PCLK•1/16			
					0x3	PCLK•1/8			
					0x2	PCLK•1/4			
					0x1	PCLK•1/2			
					0x0	PCLK•1/1			

D[15:4] Reserved

D[3:0] DF[3:0]: Timer Input Clock Select Bits

These bits select the count clock of the 16-bit timer from 15 prescaler output clocks.

Table II.4.2.9.2 Selecting the Count Clock

DF[3:0]	Prescaler output clock	DF[3:0]	Prescaler output clock
0xf	Reserved	0x7	PCLK•1/128
0xe	PCLK•1/16384	0x6	PCLK•1/64
0xd	PCLK•1/8192	0x5	PCLK•1/32
0xc	PCLK•1/4096	0x4	PCLK•1/16
0xb	PCLK•1/2048	0x3	PCLK•1/8
0xa	PCLK•1/1024	0x2	PCLK•1/4
0x9	PCLK•1/512	0x1	PCLK•1/2
0x8	PCLK•1/256	0x0	PCLK•1/1

(Default: 0x0)

Note: When setting the count clock, make sure the 16-bit timer counter is stopped.

0x4202: CLG_T16U0 Reload Data Register (CLG_T16U0_TR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T16U0 Reload Data Register (CLG_T16U0 _TR)	**0x4202** (16 bits)	D15–0	**TR[15:0]**	16-bit timer reload data TR15 = MSB TR0 = LSB	0x0 to 0xffff	0x0	R/W	

D[15:0] TR[15:0]: 16-bit Timer Reload Data Bits

Set the initial value for the counter. (Default: 0x0)

The reload data written in this register is preset to the respective counter when the timer is reset or when the counter underflows.

When starting the 16-bit timer after resetting, the timer counts down from the reload value. So the reload value and the input clock frequency determine the period of time from starting the timer until an underflow occurs (and between underflows). This makes it possible to obtain a desired wait time, a periodical interrupt interval, or programmable transfer clock for the serial interface.

0x4204: CLG_T16U0 Counter Data Register (CLG_T16U0_TC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T16U0 Counter Data Register (CLG_T16U0 _TC)	**0x4204** (16 bits)	D15–0	**TC[15:0]**	16-bit timer counter data TC15 = MSB TC0 = LSB	0x0 to 0xffff	0xffff	R	

D[15:0] TC[15:0]: 16-bit Timer Counter Data Bits

The counter data can be read from this register. (Default: 0xffff)
This is a read-only register, so the writing operation is invalid.

0x4206: CLG_T16U0 Control Register (CLG_T16U0_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T16U0 Control Register (CLG_T16U0_CTL)	0x4206 (16 bits)	D15–5	–	reserved	–	–	–	0 when being read.
		D4	TRMD	Count mode select	1 One shot 0 Repeat	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	PRESER	Timer reset	1 Reset 0 Ignored	0	W	
		D0	PRUN	Timer run/stop control	1 Run 0 Stop	0	R/W	

D[15:5] Reserved

D4 TRMD: Count Mode Select Bit

Selects the count mode of the 16-bit timer.
1 (R/W): One-shot mode
0 (R/W): Repeat mode (default)

The 16-bit timer is set in repeat mode when TRMD is set to 0. In this mode, the 16-bit timer does not stop after it starts counting until the application program stops the timer. When the counter underflows, the timer presets the reload data register value to the counter and continues counting. The timer outputs the underflow pulses periodically. Set the 16-bit timer in this mode when generating periodical interrupts with a given interval or generating the serial transfer clock.

The 16-bit timer is set in one-shot mode when TRMD is set to 1. In this mode, the 16-bit timer automatically stops counting when the counter underflows, so only one interrupt can be generated after starting the timer. When an underflow occurs, the counter is preset with the reload data register value before the timer operation stops. Set the 16-bit timer in this mode when a certain waiting time must be generated.

Note: When setting the count mode, make sure the 16-bit timer counter is stopped.

D[3:2] Reserved

D1 PRESER: Timer Reset Bit

Resets the 16-bit timer.
1 (W): Reset
0 (W): Has no effect
0 (R): Always 0 when read (default)

Writing 1 to this bit presets the reload data in the counter.

D0 PRUN: Timer Run/Stop Control Bit

Controls the timer's Run/Stop state.
1 (R/W): Run
0 (R/W): Stop (default)

The timer starts counting by writing 1 to PRUN and stops counting by writing 0.
In the stop state, the counter data is retained until the timer is reset or placed in a run state.

II.4.2.10 Precautions

- Before the 16-bit timer can start counting, the prescaler must be run.
- When setting the count clock or count mode, make sure the 16-bit timer is turned off.

II.4.3 8-bit Timers (CLG_T8S and CLG_T8I)

II.4.3.1 Outline of the 8-bit Timers

The S1C17501 CLG incorporates two channels of 8-bit timers (CLG_T8S and CLG_T8I).
The 8-bit timer includes an 8-bit presettable down counter and an 8-bit reload data register for setting the preset value. The timer counts down from the initial value set in the reload data register and outputs an underflow signal when the counter underflows. The underflow signal is used to generate an interrupt and an internal serial interface clock. The underflow period can be programmed by selecting a prescaler clock and setting reload data. This allows the application program to get any desired time intervals and programmable serial transfer rates.
Normally, CLG_T8S is used to generate the SPI CH.0 operating clock and CLG_T8I is used to generate the I^2C operating clock.
Figure II.4.3.1.1 shows the structure of the 8-bit timers.

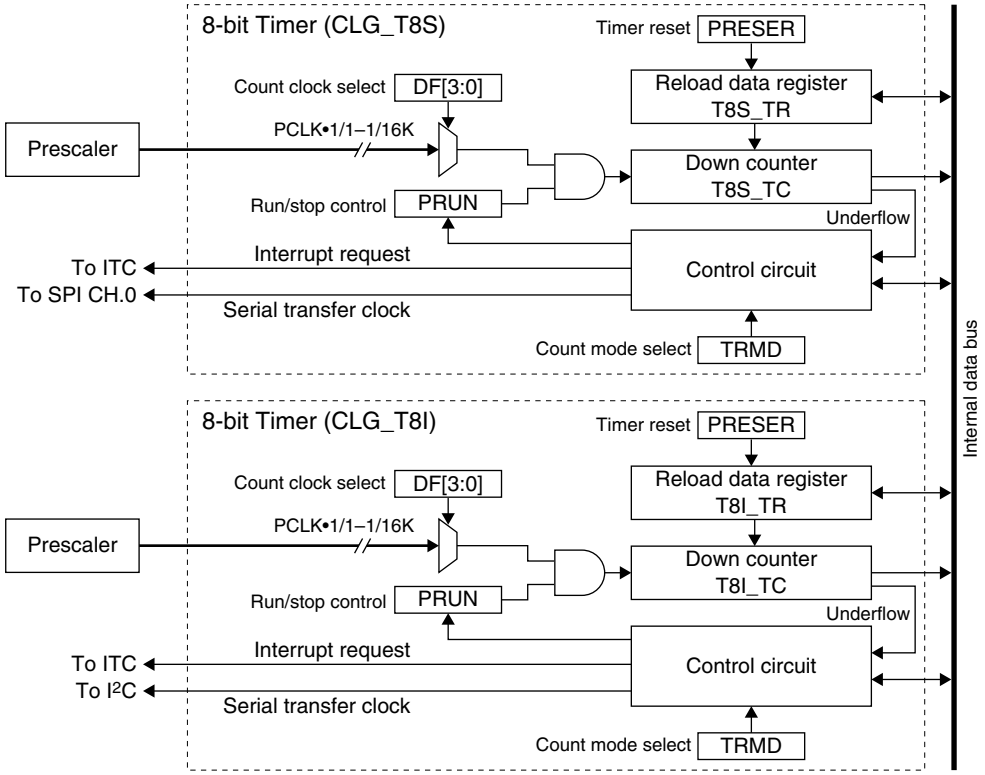


Figure II.4.3.1.1 Structure of 8-bit Timers

Note: The descriptions in this section apply to all 8-bit timers because the they have the same functions except for the control register addresses. The '*x*' in the register names denotes a timer channel (S or I) and the register addresses are described as (CLG_T8S/CLG_T8I).
Example: CLG_T8*x*_CTL register (0x4246/0x4266)
x = S: CLG_T8S_CTL register (0x4246)
x = I: CLG_T8I_CTL register (0x4266)

II.4.3.2 Count Mode of the 8-bit Timer

The 8-bit timer has two count modes: repeat mode and one-shot mode. It can be selected using TRMD (D4/CLG_T8*x*_CTL register).

∗ **TRMD**: Count Mode Select Bit in the CLG_T8*x* Control (CLG_T8*x*_CTL) Registers (D4/0x4246/0x4266)

Repeat mode (TRMD = 0, default)

The 8-bit timer is set in repeat mode when TRMD is set to 0.
In this mode, the 8-bit timer does not stop after it starts counting until the application program stops the timer. When the counter underflows, the timer presets the reload data register value to the counter and continues counting. The timer outputs the underflow pulses periodically. Set the 8-bit timer in this mode when generating periodical interrupts with a given interval or generating the serial transfer clock.

One-shot mode (TRMD = 1)

The 8-bit timer is set in one-shot mode when TRMD is set to 1.
In this mode, the 8-bit timer automatically stops counting when the counter underflows, so only one interrupt can be generated after starting the timer. When an underflow occurs, the counter is preset with the reload data register value before the timer operation stops. Set the 8-bit timer in this mode when a certain waiting time must be generated.

Note: When setting the count mode, make sure the 8-bit timer counter is stopped.

II.4.3.3 Count Clock

The 8-bit timer uses a prescaler output clock as the count clock. The prescaler divides PCLK by 1 to 16K to generate 15 clocks. Select one of the prescaler output clocks using DF[3:0] (D[3:0]/CLG_T8*x*_CLK register).

∗ **DF[3:0]**: Timer Input Clock Select Bits in the CLG_T8*x* Input Clock Select (CLG_T8*x*_CLK) Registers (D[3:0]/0x4240/0x4260)

Table II.4.3.3.1 Selecting the Count Clock

DF[3:0]	Prescaler output clock	DF[3:0]	Prescaler output clock
0xf	Reserved	0x7	PCLK•1/128
0xe	PCLK•1/16384	0x6	PCLK•1/64
0xd	PCLK•1/8192	0x5	PCLK•1/32
0xc	PCLK•1/4096	0x4	PCLK•1/16
0xb	PCLK•1/2048	0x3	PCLK•1/8
0xa	PCLK•1/1024	0x2	PCLK•1/4
0x9	PCLK•1/512	0x1	PCLK•1/2
0x8	PCLK•1/256	0x0	PCLK•1/1

(Default: 0x0)

Notes: • Before the 8-bit timer can start counting, the prescaler must be run.
• When setting the count clock, make sure the 8-bit timer counter is stopped.

For controlling the prescaler, refer to Section II.3, “Prescaler (PSC).”

II.4.3.4 8-bit Timer Reload Register and Underflow Period

The Reload Data (CLG_T8*x*_TR) Register (0x4242/0x4262) is used to set the initial value to the down counter.
The counter initial value set in the reload data register is preset to the down counter when the 8-bit timer is reset or when the counter underflows. When starting the 8-bit timer after resetting, the timer counts down from the reload value. So the reload value and the input clock frequency determine the period of time from starting the timer until an underflow occurs (and between underflows). This makes it possible to obtain a desired wait time, a periodical interrupt interval, or programmable transfer clock for the serial interface.

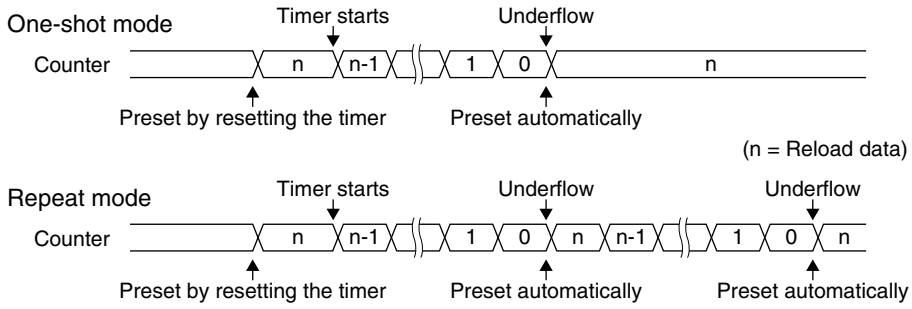


Figure II.4.3.4.1 Preset Timing

The underflow period is calculated by the expression below.

$$\text{Underflow period} = \frac{TR + 1}{clk\_in}\ [s] \qquad \text{Underflow cycle} = \frac{clk\_in}{TR + 1}\ [Hz]$$

clk_in: Count clock (prescaler output clock) frequency [Hz]
TR: Reload data (0–255)

II.4.3.5 Resetting the 8-bit Timer

To reset the 8-bit timer, write 1 to PRESER (D1/CLG_T8*x*_CTL register). This initializes the counter by presetting the Reload Data Register value.

* **PRESER**: Timer Reset Bit in the CLG_T8*x* Control (CLG_T8*x*_CTL) Registers (D1/0x4246/0x4266)

II.4.3.6 8-bit Timer Run/Stop Control

Before starting the 8-bit timer, set up the conditions as shown below.
(1) Select a count mode (one-shot or repeat). See Section II.4.3.2.
(2) Select the count clock (prescaler output clock). See Section II.4.3.3.
(3) Calculate the counter initial value and set it to the reload data register. See Section II.4.3.4
(4) Reset the timer to preset the initial value to the counter. See Section II.4.3.5.
(5) Set up the interrupt level and enable the interrupt of the timer channel if the timer interrupt is used. See Section II.4.3.8.

To start the 8-bit timer, write 1 to PRUN (D0/CLG_T8*x*_CTL register).

∗ **PRUN**: Timer Run/Stop Control Bit in the CLG_T8*x* Control (CLG_T8*x*_CTL) Registers (D0/0x4246/0x4266)

The timer starts counting down from the initial value or the current counter value if the initial value has not been preset. When the counter underflows, the timer outputs an underflow pulse and presets the initial value again. At the same time, an interrupt request is sent to the interrupt controller (ITC).
If the timer is set in one-shot mode, the timer stops counting.
If the timer is set in repeat mode, the timer continues counting from the reloaded initial value.

To stop the 8-bit timer from the application program, write 0 to the PRUN bit. The counter stops counting and holds the current counter value until the timer is reset or restarted. To restart counting from the initial value, reset the timer before writing 1 to the PRUN bit.

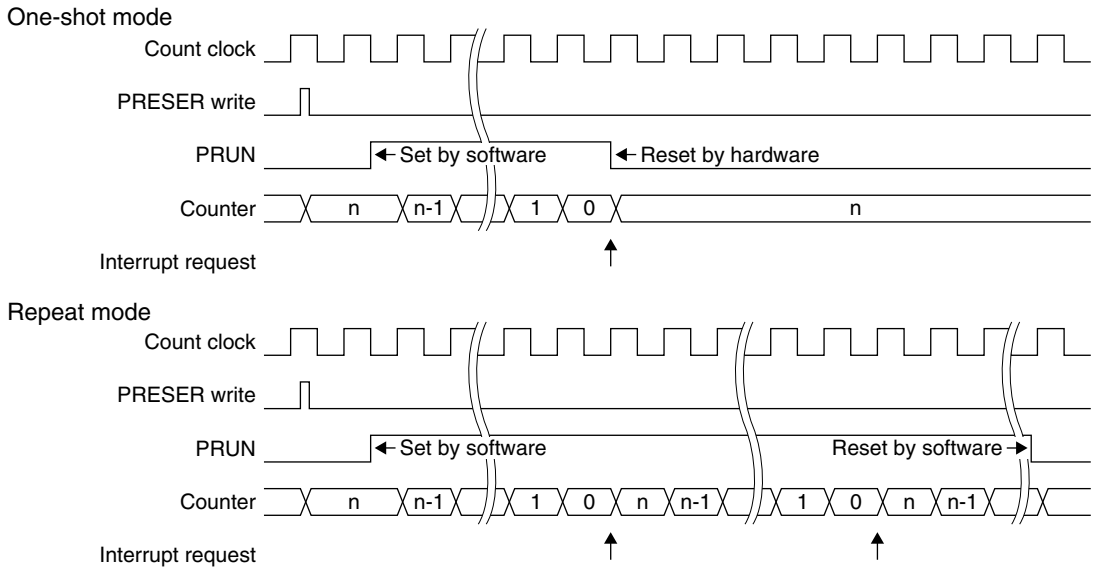


Figure II.4.3.6.1 Count Operation

II.4.3.7 8-bit Timer Output Signal

The 8-bit timer outputs an underflow pulse when the counter underflows.
This pulse is used to request a timer interrupt.
Also this pulse is used to generate a serial transfer clock for the internal serial interface.

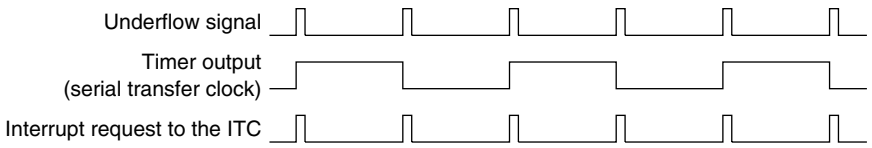


Figure II.4.3.7.1 Timer Output Clock

The generated clocks are sent to the internal serial interfaces as below.
CLG_T8S output clock → SPI
CLG_T8I output clock → I^2C

The reload data register value to obtain a desired transfer rate is calculated by the expression below.

$$TR = \frac{clk\_in}{bps \times 2} - 1$$

clk_in: Count clock (prescaler output clock) frequency [Hz]
TR: Reload data (0–255)
bps: Transfer rate (bits/second)

II.4.3.8 8-bit Timer Interrupt

The 8-bit timer outputs an interrupt request signal to the interrupt controller (ITC) when the counter underflows. To generate a timer underflow interrupt, set up the interrupt level and enable the interrupt using the ITC registers.

ITC registers for timer interrupts

Table II.4.3.8.1 shows the control registers of the ITC provided for each timer channel.

Table II.4.3.8.1 ITC Registers

Timer	Interrupt flag	Interrupt enable bit	Interrupt level setup bits
CLG_T8S	IIFT2 (D10/ITC_IFLG)	IIEN2 (D10/ITC_EN)	IILV2[2:0] (D[2:0]/ITC_ILV1)
CLG_T8I	IIFT3 (D11/ITC_IFLG)	IIEN3 (D11/ITC_EN)	IILV3[2:0] (D[10:8]/ITC_ILV1)

ITC_IFLG register (0x4300)
ITC_EN register (0x4302)
ITC_ILV1 register (0x4310)

When an underflow occurs in the timer, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the timer interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the timer underflow pulse, regardless of how the interrupt enable bit is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the timer interrupt. If the same interrupt level is set, timer Ch.0 has highest priority and timer Ch.2 has lowest priority.
An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The timer interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vectors

The following shows the vector numbers and vector addresses for the timer interrupt:

Table II.4.3.8.2 Timer Interrupt Vectors

Timer	Vector number	Vector address
CLG_T8S	14 (0x0e)	TTBR + 0x38
CLG_T8I	15 (0x0f)	TTBR + 0x3c

II.4.3.9 Details of Control Registers

Table II.4.3.9.1 List of 8-bit Timer Registers

Address	Register name		Function
0x4240	CLG_T8S_CLK	CLG_T8S Input Clock Select Register	Selects a prescaler output clock.
0x4242	CLG_T8S_TR	CLG_T8S Reload Data Register	Sets reload data.
0x4244	CLG_T8S_TC	CLG_T8S Counter Data Register	Counter data
0x4246	CLG_T8S_CTL	CLG_T8S Control Register	Sets the timer mode and starts/stops the timer.
0x4260	CLG_T8I_CLK	CLG_T8I Input Clock Select Register	Selects a prescaler output clock.
0x4262	CLG_T8I_TR	CLG_T8I Reload Data Register	Sets reload data.
0x4264	CLG_T8I_TC	CLG_T8I Counter Data Register	Counter data
0x4266	CLG_T8I_CTL	CLG_T8I Control Register	Sets the timer mode and starts/stops the timer.

The following describes each 8-bit timer register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x4240/0x4260: CLG_T8*x* Input Clock Select Registers (CLG_T8*x*_CLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
CLG_T8*x* Input Clock Select Register (CLG_T8*x*_CLK)	0x4240 0x4260 (16 bits)	D15–4	–	reserved	–		–	–	0 when being read.
		D3–0	DF[3:0]	Timer input clock select (Prescaler output clock)	DF[3:0]	Clock	0x0	R/W	
					0xf	reserved			
					0xe	PCLK•1/16384			
					0xd	PCLK•1/8192			
					0xc	PCLK•1/4096			
					0xb	PCLK•1/2048			
					0xa	PCLK•1/1024			
					0x9	PCLK•1/512			
					0x8	PCLK•1/256			
					0x7	PCLK•1/128			
					0x6	PCLK•1/64			
					0x5	PCLK•1/32			
					0x4	PCLK•1/16			
					0x3	PCLK•1/8			
					0x2	PCLK•1/4			
					0x1	PCLK•1/2			
					0x0	PCLK•1/1			

Note: The letter '*x*' in register names, etc., denotes a timer channel (S or I).

0x4240: CLG_T8S Input Clock Select Register (CLG_T8S_CLK)
0x4260: CLG_T8I Input Clock Select Register (CLG_T8I_CLK)

D[15:4] Reserved

D[3:0] DF[3:0]: Timer Input Clock Select Bits

These bits select the count clock of the 8-bit timer from 15 prescaler output clocks.

Table II.4.3.9.2 Selecting the Count Clock

DF[3:0]	Prescaler output clock	DF[3:0]	Prescaler output clock
0xf	Reserved	0x7	PCLK•1/128
0xe	PCLK•1/16384	0x6	PCLK•1/64
0xd	PCLK•1/8192	0x5	PCLK•1/32
0xc	PCLK•1/4096	0x4	PCLK•1/16
0xb	PCLK•1/2048	0x3	PCLK•1/8
0xa	PCLK•1/1024	0x2	PCLK•1/4
0x9	PCLK•1/512	0x1	PCLK•1/2
0x8	PCLK•1/256	0x0	PCLK•1/1

(Default: 0x0)

Note: When setting the count clock, make sure the 8-bit timer counter is stopped.

0x4242/0x4262: CLG_T8*x* Reload Data Registers (CLG_T8*x*_TR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T8*x*	**0x4242**	D15–8	**–**	reserved	–	–	–	0 when being read.
Reload Data Register (CLG_T8*x*_TR)	**0x4262** (16 bits)	D7–0	**TR[7:0]**	8-bit timer reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	

Note: The letter '*x*' in register names, etc., denotes a timer channel (S or I).

0x4242: CLG_T8S Reload Data Register (CLG_T8S_TR)
0x4262: CLG_T8I Reload Data Register (CLG_T8I_TR)

D[15:8] Reserved

D[7:0] TR[7:0]: 8-bit Timer Reload Data Bits

Set the initial value for the counter. (Default: 0x0)

The reload data written in this register is preset to the respective counter when the timer is reset or when the counter underflows.

When starting the 8-bit timer after resetting, the timer counts down from the reload value. So the reload value and the input clock frequency determine the period of time from starting the timer until an underflow occurs (and between underflows). This makes it possible to obtain a desired wait time, a periodical interrupt interval, or programmable transfer clock for the serial interface.

0x4244/0x4264: CLG_T8*x* Counter Data Registers (CLG_T8*x*_TC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T8*x*	**0x4244**	D15–8	–	reserved	–	–	–	0 when being read.
Counter Data Register (CLG_T8x_TC)	**0x4264** (16 bits)	D7–0	**TC[7:0]**	8-bit timer counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0xff	R	

Note: The letter '*x*' in register names, etc., denotes a timer channel (S or I).

0x4244: CLG_T8S Counter Data Register (CLG_T8S_TC)
0x4264: CLG_T8I Counter Data Register (CLG_T8I_TC)

D[15:8] Reserved

D[7:0] TC[7:0]: 8-bit Timer Counter Data Bits

The counter data can be read from this register. (Default: 0xff)
This is a read-only register, so the writing operation is invalid.

0x4246/0x4266: CLG_T8*x* Control Registers (CLG_T8*x*_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T8*x* Control Register (CLG_T8*x*_CTL)	0x4246 0x4266 (16 bits)	D15–5	–	reserved	–	–	–	0 when being read.
		D4	TRMD	Count mode select	1 One shot 0 Repeat	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	PRESER	Timer reset	1 Reset 0 Ignored	0	W	
		D0	PRUN	Timer run/stop control	1 Run 0 Stop	0	R/W	

Note: The letter '*x*' in register names, etc., denotes a timer channel (S or I).

0x4246: CLG_T8S Control Register (CLG_T8S_CTL)
0x4266: CLG_T8I Control Register (CLG_T8I_CTL)

D[15:5] Reserved

D4 TRMD: Count Mode Select Bit

Selects the count mode of the 8-bit timer.
1 (R/W): One-shot mode
0 (R/W): Repeat mode (default)

The 8-bit timer is set in repeat mode when TRMD is set to 0. In this mode, the 8-bit timer does not stop after it starts counting until the application program stops the timer. When the counter underflows, the timer presets the reload data register value to the counter and continues counting. The timer outputs the underflow pulses periodically. Set the 8-bit timer in this mode when generating periodical interrupts with a given interval or generating the serial transfer clock.

The 8-bit timer is set in one-shot mode when TRMD is set to 1. In this mode, the 8-bit timer automatically stops counting when the counter underflows, so only one interrupt can be generated after starting the timer. When an underflow occurs, the counter is preset with the reload data register value before the timer operation stops. Set the 8-bit timer in this mode when a certain waiting time must be generated.

Note: When setting the count mode, make sure the 8-bit timer counter is stopped.

D[3:2] Reserved

D1 PRESER: Timer Reset Bit

Resets the 8-bit timer.
1 (W): Reset
0 (W): Has no effect
0 (R): Always 0 when read (default)

Writing 1 to this bit presets the reload data in the counter.

D0 PRUN: Timer Run/Stop Control Bit

Controls the timer's Run/Stop state.
1 (R/W): Run
0 (R/W): Stop (default)

The timer starts counting by writing 1 to PRUN and stops counting by writing 0.
In the stop state, the counter data is retained until the timer is reset or placed in a run state.

II.4.3.10 Precautions

- Before the 8-bit timer can start counting, the prescaler must be run.
- When setting the count clock or count mode, make sure the 8-bit timer is turned off.

II.5 Real-Time Clock (RTC)

II.5.1 Overview of the RTC

The S1C17501 incorporates a real-time clock (RTC) with a perpetual calendar, and an OSC1 oscillator circuit to generate the operating clock for the RTC.
The RTC and OSC1 oscillator circuit operate in SLEEP mode. Moreover, the RTC can periodically generate interrupt requests to the CPU.
The main features of the RTC are outlined below.

- Contains time counters (seconds, minutes, and hours) and calendar counters (days, days of the week, months, and year).
- BCD data can be read from and written to both counters.
- Capable of controlling the starting and stopping of time clocks.
- 24-hour or 12-hour mode can be selected.
- A 30-second correction function can be implemented in software.
- Periodic interrupts are possible.
- Interrupt period can be selected from 1/64 second, 1 second, 1 minute, or 1 hour, with selectable level/edge interrupts.
- Independent power supply, so that the RTC can continue operating even when system power is turned off.
- A built-in OSC1 oscillator circuit (crystal oscillator or external clock input) that generates a 32.768-kHz (typ.) operating clock.
- Provides the #STBY and WAKEUP pins to control the system power supply.

Figure II.5.1.1 shows a block diagram of the RTC.

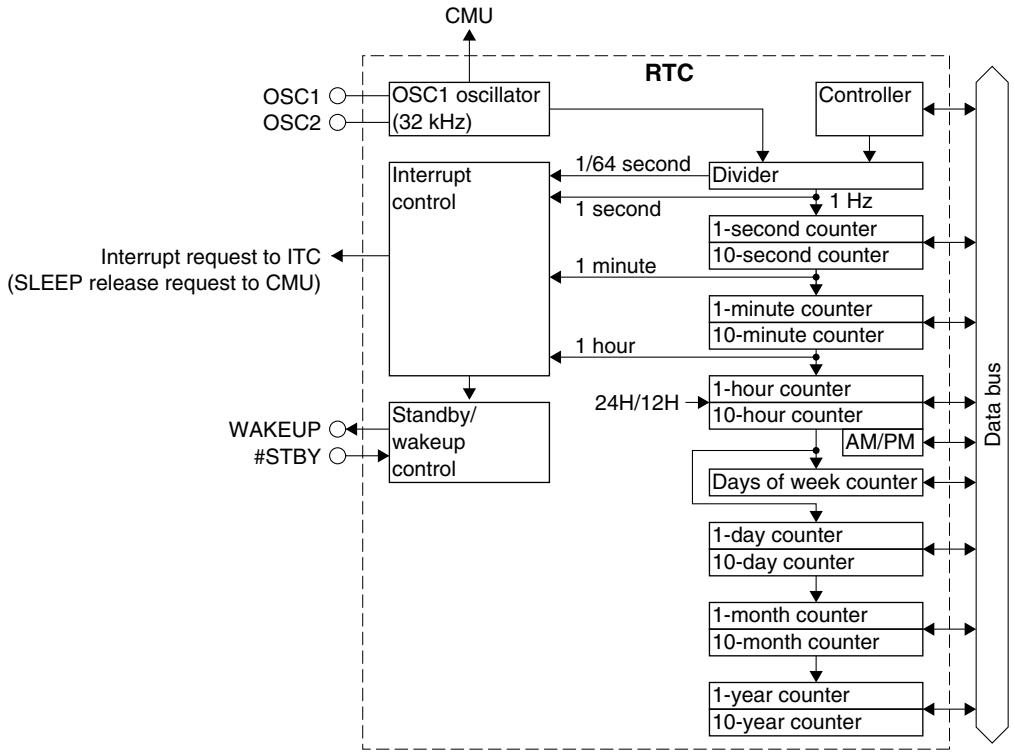


Figure II.5.1.1 RTC Block Diagram

II.5.2 RTC Counters

The RTC contains the following 13 counters, whose count values can be read out as BCD data from the respective registers. Each counter can also be set to any desired date and time by writing data to the respective register.

1-second counter

This 4-bit BCD counter counts in units of seconds. It counts from 0 to 9 synchronously with a 1-second signal derived from the 32.768-kHz OSC1 clock by dividing the clock into smaller frequencies. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-second counter. The count data is read out and written using RTCSL[3:0] (D[3:0]/RTC_SEC register).

∗ **RTCSL[3:0]**: RTC 1-second Counter Bits in the RTC Second (RTC_SEC) Register (D[3:0]/0x4614)

10-second counter

This 3-bit BCD counter counts tens of seconds. It counts from 0 to 5 with 1 carried over from the 1-second counter. This counter is reset to 0 after 5 and outputs a carry over of 1 to the 1-minute counter. The count data is read out and written using RTCSH[2:0] (D[6:4]/RTC_SEC register).

∗ **RTCSH[2:0]**: RTC 10-second Counter Bits in the RTC Second (RTC_SEC) Register (D[6:4]/0x4614)

1-minute counter

This 4-bit BCD counter counts in units of minutes. It counts from 0 to 9 with 1 carried over from the 10-second counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-minute counter. The count data is read out and written using RTCMIL[3:0] (D[3:0]/RTC_MIN register).

∗ **RTCMIL[3:0]**: RTC 1-minute Counter Bits in the RTC Minute (RTC_MIN) Register (D[3:0]/0x4615)

10-minute counter

This 3-bit BCD counter counts tens of minutes. It counts from 0 to 5 with 1 carried over from the 1-minute counter. This counter is reset to 0 after 5 and outputs a carry over of 1 to the 1-hour counter. The count data is read out and written using RTCMIH[2:0] (D[6:4]/RTC_MIN register).

∗ **RTCMIH[2:0]**: RTC 10-minute Counter Bits in the RTC Minute (RTC_MIN) Register (D[6:4]/0x4615)

1-hour counter

This 4-bit BCD counter counts in units of hours. It counts from 0 to 9 with 1 carried over from the 10-minute counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-hour counter. Depending whether 12-hour or 24-hour mode is selected, the counter is reset at 12 o'clock or 24 o'clock. The count data is read out and written using RTCHL[3:0] (D[3:0]/RTC_HOUR register).

∗ **RTCHL[3:0]**: RTC 1-hour Counter Bits in the RTC Hour (RTC_HOUR) Register (D[3:0]/0x4616)

10-hour counter

This 2-bit BCD counter counts tens of hours. With a carry over of 1 from the 1-hour counter, this counter counts from 0 to 1 (when 12-hour mode is selected) or from 0 to 2 (when 24-hour mode is selected). The counter is reset at 12 o'clock or 24 o'clock, and outputs a carry over of 1 to the 1-day counter. The count data is read out and written using RTCHH[1:0] (D[5:4]/RTC_HOUR register).

∗ **RTCHH[1:0]**: RTC 10-hour Counter Bits in the RTC Hour (RTC_HOUR) Register (D[5:4]/0x4616)

When 12-hour mode is selected, RTCAP (D6/RTC_HOUR register) that indicates A.M. or P.M. is enabled, with A.M. and P.M. represented by 0 and 1, respectively. For 24-hour mode, RTCAP is fixed to 0.

∗ **RTCAP**: AM/PM Indicator Bit in the RTC Hour (RTC_HOUR) Register (D6/0x4616)

1-day counter

This 4-bit BCD counter counts in units of days. It counts from 0 to 9 with 1 carried over from the hour counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-day counter. The number of days in each month and leap years are taken into account, so that the counter is reset to 1 when months change. The count data is read out and written using RTCDL[3:0] (D[3:0]/RTC_DAY register).

* **RTCDL[3:0]**: RTC 1-day Counter Bits in the RTC Day (RTC_DAY) Register (D[3:0]/0x4617)

10-day counter

This 2-bit BCD counter counts tens of days. It counts from 0 to 2 or 3 with 1 carried over from the 1-day counter. The number of days in each month and leap years are taken into account, so that when months change the counter is reset to 0 along with the 1-day counter, and outputs a carry over of 1 to the 1-month counter. The count data is read out and written using RTCDH[1:0] (D[5:4]/RTC_DAY register).

* **RTCDH[1:0]**: RTC 10-day Counter Bits in the RTC Day (RTC_DAY) Register (D[5:4]/0x4617)

1-month counter

This 4-bit BCD counter counts in units of months. It counts from 0 to 9 with 1 carried over from the day counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-month counter. The counter is reset to 1 when years change. The count data is read out and written using RTCMOL[3:0] (D[3:0]/RTC_MONTH register).

* **RTCMOL[3:0]**: RTC 1-month Counter Bits in the RTC Month (RTC_MONTH) Register (D[3:0]/0x4628)

10-month counter

This counter counts in units of 10 months, and is set to 1 with 1 carried over from the 1-month counter. When years change, this counter is reset to 0 along with the 1-month counter, and outputs a carry over of 1 to the 1-year counter. The count data is read out and written using RTCMOH (D4/RTC_MONTH register).

* **RTCMOH**: RTC 10-month Counter Bit in the RTC Month (RTC_MONTH) Register (D4/0x4628)

1-year counter

This 4-bit BCD counter counts in units of years. It counts from 0 to 9 with 1 carried over from the month counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-year counter. The count data is read out and written using RTCYL[3:0] (D[3:0]/RTC_YEAR register).

* **RTCYL[3:0]**: RTC 1-year Counter Bits in the RTC Year (RTC_YEAR) Register (D[3:0]/0x4629)

10-year counter

This 4-bit BCD counter counts tens of years. It counts from 0 to 9 with 1 carried over from the 1-year counter. The count data is read out and written using RTCYH[3:0] (D[7:4]/RTC_YEAR register).

* **RTCYH[3:0]**: RTC 10-year Counter Bits in the RTC Year (RTC_YEAR) Register (D[7:4]/0x4629)

Days of week counter

This is a septenary counter (that counts from 0 to 6) representing the days of the week. It counts with the same timing as the 1-day counter. The count data is read out and written using RTCWK[2:0] (D[2:0]/RTC_WEEK register).

* **RTCWK[2:0]**: RTC Days of Week Counter Bits in the RTC Days of Week (RTC_WEEK) Register (D[2:0]/0x462a)

The correspondence between the counter values and days of the week can be set in a program as desired. Table II.5.2.1 lists the basic correspondence.

Table II.5.2.1 Correspondence between Counter Values and Days of the Week

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

Initial counter values

When initially reset, the counter values are not initialized. After power-on, the counter values are indeterminate. Be sure to initialize the counters by following the procedure described in Section II.5.3.2, "Initial Sequence of the RTC."

About detection of leap years

The algorithm used in the RTC to detect leap years is for Anno Domini (A.D.) only, and can automatically identify leap years up to the year 2399.

Years (0 to 99) without a remainder when divided by 4 are considered leap years. When the 1-year and 10-year counters both are 0, a common year is assumed.

II.5.3 Control of the RTC

II.5.3.1 Controlling the Operating Clock

Counter clock

The RTC is clocked by the 32.768-kHz (typ.) OSC1 clock. The OSC1 clock is always supplied from the OSC1 oscillator circuit (even in HALT/SLEEP mode).

Register clock

The RTC_SAPB_CLK clock is used to operate the RTC control registers. To setup the registers, this clock is required. After the registers are set up, the clock supply can be stopped to reduce current consumption by setting RTC_SAPB_CLK_EN (D0/CMU_GATEDCLK2 register) to 0.

* **RTC_SAPB_CLK_EN**: RTC SAPB I/F Clock Control Bit in the Gated Clock Control 2 (CMU_GATEDCLK2) Register (D0/0x4908)

II.5.3.2 Initial Sequence of the RTC

Immediately after power-on, the contents of RTC registers are indeterminate. After powering on, follow the procedure below to let the RTC start ticking the time. Later sections detail the contents of each control.

1. Power-on
2. System initialization processing and waiting for OSC1 stabilization
   Although the OSC1 oscillator circuit starts oscillating immediately after power is switched on, a finite time of up to 3 seconds is required before the output clock stabilizes.
3. Disabling RTC interrupts
   To prevent the occurrence of unwanted RTC interrupts, the following register settings are required:
   Write 0x0 to the RTC Interrupt Mode Register (0x4601) to disable RTC interrupts.
   Write 0x1 to the RTC Interrupt Status Register (0x4600) to clear the RTC interrupt status.
   For details, see Section II.5.4, “RTC Interrupts.”
4. Starting the count
   Write 0x2 (for 12-hour mode) or 0x12 (for 24-hour mode) to the RTC Control 0 Register (0x4602) to start counting by the RTC. This operation initializes the contents of 12-hour/24-hour mode, etc. that affect count data when settings are changed, and is not the standard operation to start counting.
   For details, see Section II.5.3.3, “Selecting 12/24-hour Mode and Setting the Counters,” and Section II.5.3.4, “Starting, Stopping, and Resetting Counters.”
5. Confirming accessibility status of the RTC
   Use the RTC Control 1 Register (0x4603) to retain the counters intact and read out the busy flag to confirm that the RTC can now be accessed.
   For details, see Section II.5.3.5, “Counter Hold and Busy Flag.”
6. Stopping and resetting the count
   Write 0x1 to the RTC Control 0 Register (0x4602) to stop the count, then reset the divide-by stage of the count clock.
   For details, see Section II.5.3.4, “Starting, Stopping, and Resetting Counters.”
7. Setting the date and time
   Use the respective count registers to initialize all counters to the current date and time.
   For details, see Section II.5.3.3, “Selecting 12/24-hour Mode and Setting the Counters.”
8. Restarting count
   Release the counters from the hold state (set in step 5) and repeat step 4 to restart counting by the RTC.
   For details, see Section II.5.3.5, “Counter Hold and Busy Flag,” and Section II.5.3.4, “Starting, Stopping, and Resetting Counters.”

II.5.3.3 Selecting 12/24-hour Mode and Setting the Counters

Selecting 12-hour/24-hour mode

Whether to use the time clock in 12-hour or 24-hour mode can be selected using RTC24H (D4/RTC_CNTL0 register).
RTC24H = 1: 24-hour mode
RTC24H = 0: 12-hour mode
The count range of hour counters changes with this selection.

∗ **RTC24H**: 24H/12H Mode Select Bit in the RTC Control 0 (RTC_CNTL0) Register (D4/0x4602)

Basically, this setting should be changed while the counters are idle. RTC24H is allocated to the same address as the control bits that start the counters. Therefore, 12-hour mode or 24-hour mode can be selected at the same time the counters are started.

Note: Rewriting RTC24H may corrupt count data for the hours, days, months, years or days of the week. Therefore, once RTC24H settings are changed, be sure to set data back in these counters again.

Checking A.M./P.M. with 12-hour mode selected

When 12-hour mode is selected, RTCAP (D6/RTC_HOUR register) that indicates A.M. or P.M. is enabled.
RTCAP = 0: A.M.
RTCAP = 1: P.M.
For 24-hour mode, RTCAP is fixed to 0.

∗ **RTCAP**: AM/PM Indicator Bit in the RTC Hour (RTC_HOUR) Register (D6/0x4616)

When setting the time of day, write either of the values above to this bit to specify A.M. or P.M.

Setting the counters

Idle counters can be accessed for read or write at any time.
However, settings like those shown below should be avoided, since such settings may cause timekeeping errors.

- Settings exceeding the effective range
  Do not set count data exceeding 60 seconds, 60 minutes, 12 or 24 hours, 31 days, 12 months, or 99 years.
- Settings nonexistent in the calendar
  Do not set such nonexistent dates as April 31 or February 29, 2006. Even if such settings are made, the counters operate normally, so that when 1 is carried over from the hour counter to the 1-day counter, the day counter counts up to the first day of the next month. (For April 31, the day counter counts up to May 1; for February 29, 2006, the day counter counts up to March 1, 2006.)

If any counter must be rewritten while operating, there is a procedure that must be followed to ensure that the counter is rewritten correctly. For details, see Section II.5.3.6, "Reading from and Writing to Counters in Operation."

II.5.3.4 Starting, Stopping, and Resetting Counters

Starting and stopping counters

The RTC starts counting when RTCSTP (D1/RTC_CNTL0 register) is set to 0, and stops counting when this bit is set to 1.

∗ **RTCSTP**: Counter Run/Stop Control Bit in the RTC Control 0 (RTC_CNTL0) Register (D1/0x4602)

The RTC is stopped by writing 1 to RTCSTP at the 32-kHz input clock divide-by stage of 8,192 Hz or those stages that follow. The RTC does not stop at up to the input clock divide-by-2 stage (16,384 Hz).
If the RTC stops counting when 1 is carried over to the next-digit counter, the count value may be corrupted. Therefore, see the next section to ensure that 1 is not carried over when counters are made to stop. This is unnecessary, however, when the contents of all counters are newly set again.

Resetting the counters

RTCRST (D0/RTC_CNTL0 register) is the bit used to reset the 32 kHz to 2 Hz counters.

∗ **RTCRST**: Software Reset Bit in the RTC Control 0 (RTC_CNTL0) Register (D0/0x4602)

Setting RTCRST to 1 resets the counters above (cleared to 0), and writing 0 to this bit negates the reset.

II.5.3.5 Counter Hold and Busy Flag

If 1 is carried over when reading the counters, the correct counter value may not be read out. Moreover, if a write or stop operation is attempted, the counter value may be corrupted. Therefore, whether counters are in a carry (busy) state should be checked before reading or writing data from or to the count registers. For this purpose, control bits RTCBSY (D1/RTC_CNTL1 register) and RTCHLD (D0/RTC_CNTL1 register) are provided.

- **RTCBSY**: Counter Busy Flag Bit in the RTC Control 1 (RTC_CNTL1) Register (D1/0x4603)
- **RTCHLD**: Counter Hold Control Bit in the RTC Control 1 (RTC_CNTL1) Register (D0/0x4603)

RTCBSY is a read-only flag indicating that 1 is being carried over. RTCBSY is set to 1 when 1 is being carried over; otherwise, it is 0. RTCBSY should be confirmed as being 0 before accessing the counters to ensure that the correct value will be read or set.
Note, however, that RTCBSY is fixed to 1 while counting is in progress. To reflect the current state in the count value, RTCHLD should be set to 1.

RTCBSY = 0 (RTC accessible)

If the value of RTCBSY is 0 when this bit is read out after writing 1 to RTCHLD, it means that 1 is not being carried over. In this case, the counter hold function is actuated, with a carry over of 1 to the 1-second counter disabled in hardware. Counters that count less than seconds continue operating.
Data can be read from or written to the count registers in this state.
After reading or writing data, reset RTCHLD to 0.

When 1 must be carried over while data is being read or written with counters in the hold state, 1 second is automatically added at the time, with RTCHLD reset to 0 to correct the count value. This correction is effective for only 1 second, and the time to carry over 1 on subsequent occasions is ignored. In this case, timekeeping data gets out of order. Therefore, be sure to reset RTCHLD to 0 as soon as possible after completing the necessary read or write operation.

RTCBSY = 1 (RTC is busy)

If the value of RTCBSY is 1 when this bit is read after writing 1 to RTCHLD, it means that 1 is being carried over. The period needed for the counters to carry over 1 is 4 ms per second. In this case, reset RTCHLD to 0 as soon as possible and [A] recheck RTCBSY by following the same procedure or [B] wait 4 ms before checking RTCBSY.
If RTCBSY is found to be 1, be sure to immediately reset RTCHLD to 0. If RTCHLD is left at 1, the time of day may become incorrect.

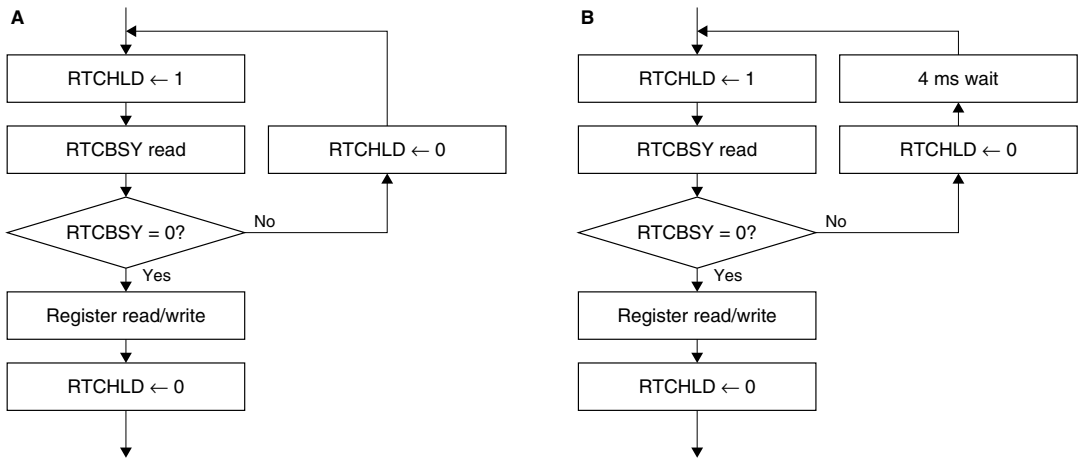


Figure II.5.3.5.1 Procedure for Checking whether the RTC is Busy

There is also a method of reading out data without using RTCHLD and RTCBSY. (See the next section.)

II.5.3.6 Reading from and Writing to Counters in Operation

As described in the previous section, the counters must be accessed for read/write when 1 is not being carried over. Follow the procedure shown in the flowchart in Figure II.5.3.5.1 to read from or write to the counters.

The counters can be read without using RTCHLD and RTCBSY, as shown in Figure II.5.3.6.1.

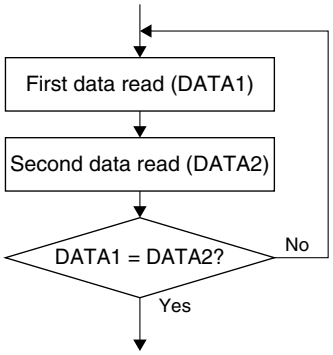


Figure II.5.3.6.1 Procedure for Reading Counters not in the Hold State

II.5.3.7 30-second Correction

The description "30-second correction" means resetting the seconds to 0 and adding 1 to the minutes when seconds of the time clock are in the range of 30 to 59 seconds. When in the range of 0 to 29 seconds, the RTC resets the seconds to 0 but it does not change the minutes. This function may be used to round up seconds to minutes when resetting seconds in an application.
This function can be executed by writing 1 to RTCADJ (D2/RTC_CNTL0 register).

∗ **RTCADJ**: 30-second Adjustment Bit in the RTC Control 0 (RTC_CNTL0) Register (D2/0x4602)

Writing 1 to RTCADJ causes the RTC to operate as follows:

- When the 10-second counter is 3 or more, the RTC generates a carry over of 1 to start counting by the 1-minute counter.
- When the 10-second counter is 2 or less, the RTC does not generate a carry over of 1.

After RTCADJ is set to 1, it remains set for the 4-ms period required for this processing, then automatically returns to 0.
Accessing the counters while RTCADJ = 1 is prohibited. Writing 0 to RTCADJ is also prohibited, because it would cause the RTC to operate erratically.

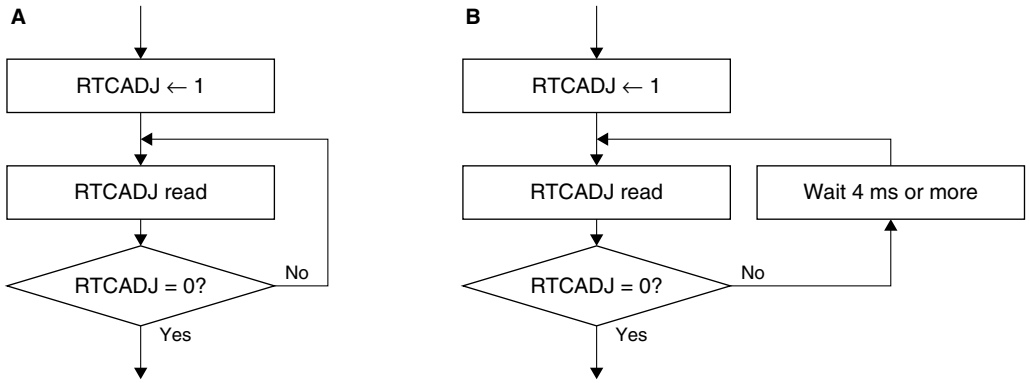


Figure II.5.3.7.1 Procedure for Executing 30-second Correction

II.5.4 RTC Interrupts

The RTC has a function to generate interrupts at given intervals.
Since the RTC is active even in standby mode, interrupts may be used to turn off SLEEP mode.
This section describes the internal interrupt control function of the RTC. To generate interrupts to the CPU, the interrupt controller (ITC) must also be set up. For details on how to control the ITC, see Section IV.1, "Interrupt Controller (ITC)." For details on how to turn off SLEEP mode using an interrupt, see Section II.2, "Clock Management Unit (CMU)."

Setting the interrupt cycle

The interrupt cycle (in which the RTC outputs interrupt requests at specific intervals) can be selected from four choices listed in Table II.5.4.1 by using RTCT[1:0] (D[3:2]/RTC_INTMODE register).

* **RTCT[1:0]**: RTC Interrupt Cycle Setup Bits in the RTC Interrupt Mode (RTC_INTMODE) Register (D[3:2]/0x4601)

Table II.5.4.1 Interrupt Cycle Settings

RTCT[1:0]	Interrupt cycle
0x3	1 hour
0x2	1 minute
0x1	1 second
0x0	1/64 second

RTCT[1:0] should be set while RTC interrupts are disabled. (See the procedure for enabling and disabling interrupts described below.)

Setting interrupt conditions

The interrupt requests sent to the ITC can be selected as edge-triggered or level-sensed interrupts by setting a register bit. RTCIMD (D1/RTC_INTMODE register) is the bit provided for this purpose.

* **RTCIMD**: RTC Interrupt Mode Select Bit in the RTC Interrupt Mode (RTC_INTMODE) Register (D1/0x4601)

Setting RTCIMD to 1 selects a level-sensed interrupt; setting it to 0 selects an edge-triggered interrupt.
When an edge-triggered interrupt has been selected, the RTC outputs an interrupt pulse to the ITC using the bus clock supplied from the CMU. If a cause of interrupt occurs when the bus clock has not been supplied such as in SLEEP mode, the RTC switches the interrupt mode to level-sensed and sets the interrupt signal to the active level from occurrence of the interrupt cause until the bus clock supply is started.

Enabling and disabling interrupts

The RTC interrupt requests output to the ITC are enabled by setting RTCIEN (D0/RTC_INTMODE register) to 1 and disabled by setting it to 0.

* **RTCIEN**: RTC Interrupt Enable Bit in the RTC Interrupt Mode (RTC_INTMODE) Register (D0/0x4601)

Interrupt status

When the RTC is up and running, RTCIRQ (D0/RTC_INTSTAT register) is set at the cyclic interrupt intervals set up by RTCT[1:0]. When RTC interrupts are enabled by RTCIEN, interrupt requests are sent to the ITC.

* **RTCIRQ**: Interrupt Status Bit in the RTC Interrupt Status (RTC_INTSTAT) Register (D0/0x4600)

Writing 1 to this status bit clears the bit. Because this bit is not cleared in hardware, be sure to clear it in software after an interrupt is generated. If this bit remains set while interrupts are re-enabled or control is returned from the interrupt handler routine by the reti instruction, the same interrupt may be generated again.

Precautions

All RTC interrupt control bits described above are indeterminate when power is turned on. Moreover, these bits are not initialized to specific values by an initial reset.
After power-on, be sure to set RTCIEN to 0 (interrupt disabled) to prevent the occurrence of unwanted RTC interrupts. Also be sure to write 1 to RTCIRQ to reset it.

II.5.5 OSC1 Oscillator Circuit

The S1C17501 contains an oscillator circuit (OSC1) used to generate a 32.768 kHz (typ.) clock as the clock source for timekeeping operation of the RTC.
The OSC1 clock can also be used as a power-saving operating clock for the core system or peripheral circuits. For details, see Section II.2, "Clock Management Unit (CMU)."

II.5.5.1 Input/Output Pins of the OSC1 Oscillator Circuit

Table II.5.5.1.1 lists the input/output pins of the OSC1 oscillator circuit.

Table II.5.5.1.1 Input/Output Pins of the OSC1 Oscillator Circuit

Pin name	I/O	Function
OSC1	I	OSC1 oscillator input pin: Crystal oscillator or external clock input
OSC2	O	OSC1 oscillator output pin: Crystal oscillator (left open when using external clock input)

II.5.5.2 Structure of the OSC1 Oscillator Circuit

The OSC1 oscillator circuit accommodates a crystal oscillator and external clock input. As for the RTC, $RTCV_{DD}$ is used to supply power to this circuit.
Figure II.5.5.2.1 shows the structure of the OSC1 oscillator circuit.

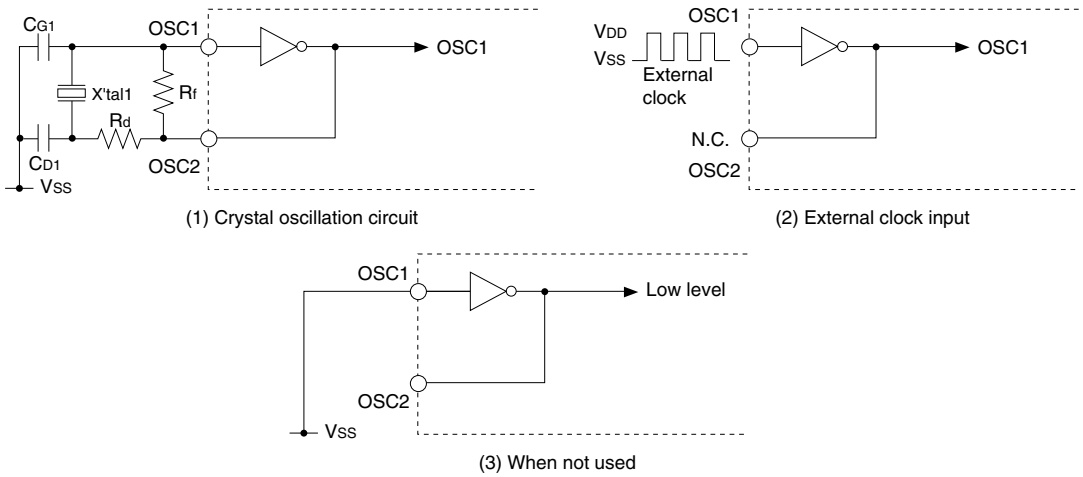


Figure II.5.5.2.1 OSC1 Oscillator Circuit

For use as a crystal oscillator circuit, connect a crystal resonator X'tal1 (32.768 kHz, typ.), feedback resistor (R_f), two capacitors (C_{G1}, C_{D1}), and, if necessary, a drain resistor (R_d) to the OSC1 and OSC2 pins and V_{SS}, as shown in Figure II.5.5.2.1 (1).
To use an external clock, leave the OSC2 pin open and input a $RTCV_{DD}$ level clock (whose duty cycle is 50%) to the OSC1 pin.
The oscillator frequency/input clock frequency is 32.768 kHz (typ.). Make sure the crystal resonator or external clock used in the RTC has this clock frequency. With any other clock frequencies, the RTC cannot be used for timekeeping purposes.
For details of oscillation characteristics and the input characteristics of external clock, see "Electrical Characteristics."
When not using the OSC1 oscillator circuit, connect the OSC1 pin to V_{SS} and leave the OSC2 pin open.

The OSC1 oscillator always operates without controlling using a register.

Note: When the oscillator is made to start oscillating at power-on, a finite time (of up to 3 seconds) is required until oscillation stabilizes. To prevent system malfunction, do not use the oscillator-derived clock until this oscillation stabilization time elapses.

II.5.6 WAKEUP and #STBY Pins

The S1C17501 has a battery backup function that allows the system to turn the system power (V_{DD}, AV_{DD}) off with the RTC (including the OSC1 oscillator circuit) kept active and the IRAM2 data maintained by supplying $RTCV_{DD}$. The RTC provides the WAKEUP and #STBY pins used for controlling this function.
The #STBY pin is used to disconnect the circuits driven with $RTCV_{DD}$ (RTC, OSC1, and IRAM2) from the other circuits driven with V_{DD} or AV_{DD} (including the control registers for RTC and OSC1). The #STBY pin must be set to a high level during normal operation. Setting the #STBY pin to a low level from outside the IC disconnects the $RTCV_{DD}$ circuits from the system allowing the system power (V_{DD}, AV_{DD}) turned off.
The WAKEUP pin is an output pin of which the output can be controlled by the RTC interrupt or software. This output can control the external regulator to turn the system power (V_{DD}, AV_{DD}) on and off.
Note that leakage currents flow from the $RTCV_{DD}$ system to the V_{DD}/AV_{DD} system if the system power is turned off when the #STBY pin is set to a high level. Therefore, the #STBY pin must be set to a low level before the system power is turned off.
Figure II.5.6.1 shows an example of system standby/wakeup circuit using the WAKEUP and #STBY pins.

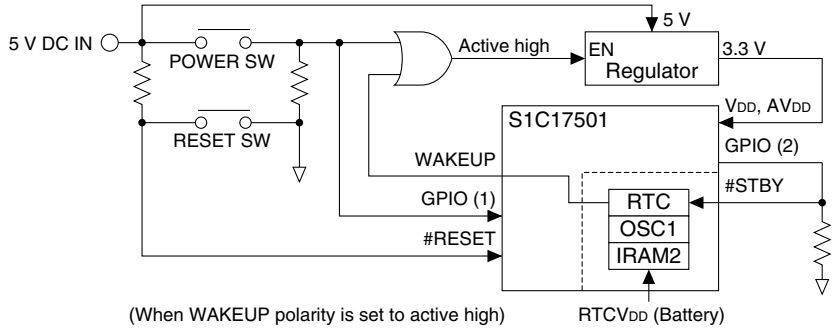


Figure II.5.6.1 Example of System Standby/Wakeup Circuit

Selecting the WAKEUP signal polarity

Use WUP_POL (D0/RTC_WAKEUP register) to select the WAKEUP output level when it is asserted by an RTC interrupt or software control.

* **WUP_POL**: WAKEUP Polarity Selection Bit in the RTC Wakeup Configuration (RTC_WAKEUP) Register (D0/0x460f)

The WAKEUP output is configured to active high signal when WUP_POL is set to 0 or active low signal when WUP_POL is set to 1. WUP_POL is not initialized at initial reset, therefore, it must be initialized with software when using the WAKEUP output.

Controlling the WAKEUP output

Controlling by an RTC interrupt

When the cause of RTC interrupt that has been selected with software (see Section II.5.4) occurs, the WAKEUP signal is asserted similar to the interrupt request signal. The RTC maintains the WAKEUP signal at the active level until the system resumes operating and clears the RTC interrupt status bit RTCIRQ (D0 /RTC_INTSTAT register). The WAKEUP signal will be negated after RTCIRQ is cleared.

Software control

The WAKEUP output can also be controlled using WUP_CTL (D1/RTC_WAKEUP register).

* **WUP_CTL**: WAKEUP Control Bit in the RTC Wakeup Configuration (RTC_WAKEUP) Register (D1/0x460f)

The WAKEUP signal is asserted by setting WUP_CTL to 1 and is negated by setting WUP_CTL to 0. WUP_CTL is not initialized at initial reset, therefore, it must be set to 1 (active) at the beginning with the initialize routine.
The table below shows the WAKEUP signal status according to the control bit.

Table II.5.6.1 WAKEUP Signal Status

Control bit settings			WAKEUP pin status
WUP_POL	WUP_CTL	RTCIRQ	
1	1	1	0 (Low)
1	1	0	0 (Low)
1	0	1	0 (Low)
1	0	0	1 (High)
0	1	1	1 (High)
0	1	0	1 (High)
0	0	1	1 (High)
0	0	0	0 (Low)

Control procedures

The following shows some power control procedures using the system standby/wakeup circuit shown in Figure II.5.6.1. The description below assumes that the power (5 V) is supplied to the regulator and the WAKEUP signal polarity is set to active high.

Power On using the POWER SW

(1) Press the POWER SW. The switch must be held down until Step (5) has completed.
(2) The regulator is enabled to output voltage and the 3.3 V voltage is supplied to the S1C17501 V_{DD} and AV_{DD} pins.
(3) The CPU starts operating and executes the initialize routine after power-on reset.
(4) Configure GPIO (2) as an output port and set the port output level to 1 (high). This signal is fed to the #STBY pin resulting that the $RTCV_{DD}$ system circuits will be connected to the system.
(5) Write 0x2 to the RTC_WAKEUP register to set the WAKEUP polarity to active high and enable the WAKEUP pin to output 1 (high). This control fixes the regulator output to be enabled, thus the POWER SW can be released (turned off).
(6) Read the key from the specific IRAM2 location and check whether the backup data is valid or not (e.g. valid if 0xaa). Then if valid, read the backup data from the IRAM2.
(7) Clear the key located in the IRAM2 (e.g. write a value such as 0x00).
(8) Execute other processing.

Keep the #STBY input = 1 and WAKEUP output = 1 conditions while the IC is operating.

Power Off using the POWER SW

The following procedure should be started under the above condition (#STBY input = 1 and WAKEUP output = 1).

(1) Press the POWER SW.
(2) The GPIO (1) port inputs 1 (high). Detect this status by reading the input data or using an interrupt from the port, and execute the sequence to place the S1C17501 into battery backup mode.
(3) Copy the data required to be saved into the IRAM2. In addition to this, write a key for indicating that the backup data is valid (e.g. 0xaa) to the specific location in the IRAM2.
(4) Set the RTC interrupt conditions and enable the interrupt. (when restarting the system using an RTC interrupt)
(5) Set the GPIO (2) port to output 0 (low). This signal is fed to the #STBY pin resulting that the $RTCV_{DD}$ system circuits will be disconnected from the system.
(6) Write 0x0 to the RTC_WAKEUP register to set the WAKEUP pin to output 0 (low).
(7) (After the POWER SW is turned off if it is still on,) The regulator stops generating 3.3 V and the power of the S1C17501 except the $RTCV_{DD}$ is turned off.

When automatically turning the system power off by software control, start the above procedure from Step (3).

Power On using an RTC interrupt

(1) When an RTC interrupt occurs, the WAKEUP output level goes 1 (high).
(2) The regulator is enabled to output voltage and the 3.3 V voltage is supplied to the S1C17501 V_{DD} and AV_{DD} pins.
(3) The CPU starts operating and executes the initialize routine after power-on reset.
(4) Configure GPIO (2) as an output port and set the port output level to 1 (high). This signal is fed to the #STBY pin resulting that the $RTCV_{DD}$ system circuits will be connected to the system.
(5) Write 0x2 to the RTC_WAKEUP register to set the WAKEUP polarity to active high and enable the WAKEUP pin to output 1 (high). This control fixes the regulator output to be enabled.
(6) Reset RTCIRQ (D0 /RTC_INTSTAT register) to 0.
(7) Read the key from the specific IRAM2 location and check whether the backup data is valid or not (e.g. valid if 0xaa). Then if valid, read the backup data from the IRAM2.
(8) Clear the key located in the IRAM2 (e.g. write a value such as 0x00).
(9) Execute other processing.

II.5.7 Details of Control Registers

Table II.5.7.1 RTC Register List

Address	Register name		Function
0x4600	RTC_INTSTAT	RTC Interrupt Status Register	Indicates RTC interrupt status.
0x4601	RTC_INTMODE	RTC Interrupt Mode Register	Sets up RTC interrupt modes.
0x4602	RTC_CNTL0	RTC Control 0 Register	Controls the RTC.
0x4603	RTC_CNTL1	RTC Control 1 Register	
0x460f	RTC_WAKEUP	RTC Wakeup Configuration Register	Sets up RTC wakeup conditions.
0x4614	RTC_SEC	RTC Second Register	Second counter data
0x4615	RTC_MIN	RTC Minute Register	Minute counter data
0x4616	RTC_HOUR	RTC Hour Register	Hour counter data
0x4617	RTC_DAY	RTC Day Register	Day counter data
0x4628	RTC_MONTH	RTC Month Register	Month counter data
0x4629	RTC_YEAR	RTC Year Register	Year counter data
0x462a	RTC_WEEK	RTC Days of Week Register	Days of week counter data

The following describes each RTC register. These are all 8-bit registers.

Notes:
- When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”
- The contents of all RTC control registers are indeterminate when power is turned on, and are not initialized to specific values by initial reset. These registers should be initialized in software.
- If 1 is being carried over when the counters are accessed for read, the correct counter value may not be read out. Moreover, attempting to write to a counter or other control register may corrupt the counter value. Therefore, do not write to counters while 1 is being carried over. For the correct method of operation, see Section II.5.3.5, “Counter Hold and Busy Flag,” and Section II.5.3.6, “Reading from and Writing to Counters in Operation.”

0x4600: RTC Interrupt Status Register (RTC_INTSTAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Interrupt Status Register (RTC_INTSTAT)	0x4600 (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	RTCIRQ	Interrupt status	1 Occurred 0 Not occurred	X	R/W	Reset by writing 1.

D[7:1] Reserved

D0 RTCIRQ: Interrupt Status Bit

This bit indicates whether a cause of RTC interrupt occurred as follows:

1 (R): Cause of interrupt occurred
0 (R): No cause of interrupt occurred
1 (W): Resets this bit to 0
0 (W): Has no effect

This bit is set at cyclic interrupt intervals set up by RTCT[1:0] (D[3:2]/RTC_INTMODE register). When RTC interrupts have been enabled by RTCIEN (D0/RTC_INTMODE register) at this time, an interrupt request is sent to the ITC. This bit is always set, even when RTC interrupts are disabled.

Note: Writing 1 to this status bit clears it. Because this bit is not cleared in hardware, be sure to clear it in software after an interrupt is generated. If this bit remains set while interrupts are re-enabled or control is returned from the interrupt handler routine by the reti instruction, the same interrupt may be generated again.
Moreover, the value of this bit is indeterminate after power-on, and is not initialized to 0 by initial reset. To prevent the occurrence of unwanted RTC interrupts, be sure to reset this bit in software after power-on and initial reset.

0x4601: RTC Interrupt Mode Register (RTC_INTMODE)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
RTC Interrupt Mode Register (RTC_INTMODE)	0x4601 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3–2	RTCT[1:0]	RTC interrupt cycle setup	RTCT[1:0]	Cycle			X	R/W	
					0x3	1 hour					
					0x2	1 minute					
					0x1	1 second					
					0x0	1/64 second					
		D1	RTCIMD	RTC interrupt mode select	1	Level sense	0	Edge trigger	X	R/W	
		D0	RTCIEN	RTC interrupt enable	1	Enable	0	Disable	X	R/W	

D[7:4] Reserved

D[3:2] RTCT[1:0]: RTC Interrupt Cycle Setup Bits

These bits select the RTC interrupt cycle.

Table II.5.7.2 Interrupt Cycle Settings

RTCT[1:0]	Interrupt cycle
0x3	1 hour
0x2	1 minute
0x1	1 second
0x0	1/64 second

(Default: indeterminate)

RTCIRQ (D0/RTC_INTSTAT register) is set by a count-up pulse of the interrupt cycle counter selected. When RTC interrupts are enabled by RTCIEN (D0), an interrupt request is sent to the ITC.
RTCT[1:0] should be set while RTC interrupts are disabled. (These bits may also be set simultaneously when RTC interrupts are enabled.)

D1 RTCIMD: RTC Interrupt Mode Select Bit

This bit specifies whether RTC interrupts are to be generated by an edge or level of the interrupt request signal.
1 (R/W): Level sensed
0 (R/W): Edge triggered

When an edge-triggered interrupt is selected and used to turn off SLEEP mode via the CMU, note that no interrupts will be generated because the ITC is inactive. When an RTC interrupt handler routine must be executed after exiting SLEEP mode, select a level-sensed interrupt.

D0 RTCIEN: RTC Interrupt Enable Bit

This bit enables or disables RTC interrupt request output to the ITC.
1 (R/W): Enable interrupts
0 (R/W): Disable interrupts

To generate an RTC interrupt or use an RTC interrupt request signal to turn off SLEEP mode, set this bit to 1. When this bit is 0, no interrupts are generated even when RTCIRQ (D0/RTC_INTSTAT register) is set and SLEEP mode cannot be turned off.

Note: The value of RTCIEN is indeterminate after power-on, and not initialized to 0 by initial reset. To prevent the occurrence of unwanted RTC interrupts, be sure to clear this bit in software after power-on and initial reset.

0x4602: RTC Control 0 Register (RTC_CNTL0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Control 0 Register (RTC_CNTL0)	0x4602 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	RTC24H	24H/12H mode select	1 24H 0 12H	X	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2	RTCADJ	30-second adjustment	1 Adjust 0 –	X	R/W	
		D1	RTCSTP	Counter run/stop control	1 Stop 0 Run	X	R/W	
		D0	RTCRST	Software reset	1 Reset 0 –	X	R/W	

D[7:5] Reserved

D4 RTC24H: 24H/12H Mode Select Bit

This bit selects whether to use the hour counter in 24-hour or 12-hour mode.
1 (R/W): 24-hour mode
0 (R/W): 12-hour mode

The count range of hour counters changes with this selection.
Basically, this setting should be changed while the counters are idle. Since this register is assigned a control bit (D1) to start the counters, 12-hour or 24-hour mode may be selected when starting the counters.

Note: Rewriting RTC24H may corrupt the count data for hours, days, months, years, or days of the week. Therefore, after changing the RTC24H setting, be sure to set data back in these counters again.

D3 Reserved

D2 RTCADJ: 30-second Adjustment Bit

This bit executes 30-second correction.
1 (W): Execute 30-second correction
0 (W): Has no effect
1 (R): 30-second correction being executed
0 (R): 30-second correction completed (not being executed)

The description "30-second correction" means adding 1 to the minutes when seconds of the time clock are in the 30-to-59 second range, and doing nothing in the 0-to-29 second range. This function may be used to round up seconds to minutes when resetting seconds in an application.
Writing 1 to this bit causes the RTC to operate as follows:

- When the 10-second counter is 3 or more, the RTC generates a carry over of 1 to start counting by the 1-minute counter.
- When the 10-second counter is 2 or less, the RTC does not generate a carry over of 1.

After being set to 1, this bit remains set for the 4-ms period needed for the processing above, then is automatically reset to 0.

Note: Accessing the counters while RTCADJ = 1 is prohibited. Writing 0 to this bit during such time is also prohibited, because it would cause the RTC to operate erratically.

D1 RTCSTP: Counter Run/Stop Control Bit

This bit starts or stops the counters. It also indicates counter operating status.
1 (R/W): Stops counters/Counters idle
0 (R/W): Starts counters/Counters operating

Setting this bit to 0 starts the counters; setting it to 1 stops the counters.
The value read from this bit is 0 when the counters are operating, and 1 when the counters are idle.
Writing 1 to this bit stops the counters at the 32-kHz input clock divide-by stage of 8,192 Hz or stages that follow. The counters do not stop at up to the input clock divide-by-2 stage (16,384 Hz).
If the counters stop while 1 is being carried over, the count value may be corrupted. Therefore, see Section II.5.3.5 to ensure that 1 is not being carried over when the counters are stopped. This is unnecessary when, for example, the contents of all counters are newly set again.

D0 RTCRST: Software Reset Bit

This bit resets the counters currently at divide-by stages.
1 (R/W): Reset counters
0 (R/W): Negate reset

Setting this bit to 1 resets the 32 kHz to 2 Hz counters (cleared to 0). Writing 0 to this bit negates the reset.

0x4603: RTC Control 1 Register (RTC_CNTL1)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
RTC Control 1 Register (RTC_CNTL1)	0x4603 (8 bits)	D7–2	–	reserved	–				–	–	0 when being read.
		D1	RTCBSY	Counter busy flag	1	Busy	0	R/W possible	X	R	
		D0	RTCHLD	Counter hold control	1	Hold	0	Running	X	R/W	

D[7:2] Reserved

D1 RTCBSY: Counter Busy Flag Bit

This flag indicates whether 1 is being carried over to the next-digit counter.

1 (R): Busy (while 1 is being carried over)

0 (R): Accessible for read/write

1/0 (W): Has no effect

If 1 is being carried over while the counters are being read, correct counter values may not be read. Moreover, attempting a write or stop operation may corrupt the counter values. Therefore, this bit should be checked to confirm that the counters are not in a carry (busy) state before reading or writing data from or to the count registers.
However, because this bit is fixed to 1 while the counters are operating, RTCHLD (D0) should be set to 1 so that the count value reflects the current state.

When a value of 0 is read from this bit after writing 1 to RTCHLD (D0), it means that 1 is not now being carried over. In this case, the counter hold function is also actuated, with a carry over of 1 to the 1-second counter disabled in hardware. Counters for less than seconds continue operating. In this state, data can be read from or written to the count registers. After reading or writing data, reset RTCHLD (D0) to 0.
If 1 is being carried over when data is being read from or written to counters in the hold state, 1 second is automatically added at that time, with RTCHLD (D0) reset to 0 for correcting the count value. This correction is only effective for 1 second, thus ignoring the time needed to carry over 1 on subsequent occasions. In this case, the timekeeping data gets out of order. Therefore, be sure to reset RTCHLD (D0) to 0 as soon as possible after completing the required read or write operation.

When a value of 1 is read from this bit after writing 1 to RTCHLD (D0), it means that 1 is now being carried over. A period of 4 ms per second is required for a carry over of 1 to the counters. In this case, reset RTCHLD (D0) to 0 as soon as possible and check this bit again by following the same procedure, or wait 4 ms before checking this bit. If this bit is set to 1, always reset RTCHLD (D0) to 0 immediately. Leaving RTCHLD (D0) set to 1 may result in an incorrect time of day.

D0 RTCHLD: Counter Hold Control Bit

This bit allows the busy state of counters to be checked and the counters held intact.

1 (R/W): Checks for busy state/Holds counters

0 (R/W): Normal operation

For the operation of this bit, see the description of RTCBSY (D1) above.

0x460f: RTC Wakeup Configuration Register (RTC_WAKEUP)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
RTC Wakeup Configuration Register (RTC_WAKEUP)	0x460f (8 bits)	D7–2	–	reserved	–				–	–	0 when being read.
		D1	WUP_CTL	WAKEUP control	1	Active	0	Inactive	X	R/W	
		D0	WUP_POL	WAKEUP polarity selection	1	Active low	0	Active high	X	R/W	

D[7:2] Reserved

D1 WUP_CTL: WAKEUP Control Bit

This bit controls the WAKEUP output.

1 (R/W): Active

0 (R/W): Inactive

This bit is used to control the WAKEUP output with software. The WAKEUP signal will also be asserted when a cause of RTC interrupt occurs.

D0 WUP_POL: WAKEUP Polarity Selection Bit

This bit select the active level of the WAKEUP output signal.

1 (R/W): Active low

0 (R/W): Active high

0x4614: RTC Second Register (RTC_SEC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Second Register (RTC_SEC)	0x4614 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–4	RTCSH[2:0]	RTC 10-second counter	0 to 5	X	R/W	
		D3–0	RTCSL[3:0]	RTC 1-second counter	0 to 9	X	R/W	

Note: Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, “Counter Hold and Busy Flag,” and Section II.5.3.6, “Reading from and Writing to Counters in Operation.”)

D7 Reserved

D[6:4] RTCSH[2:0]: RTC 10-second Counter Bits

These bits comprise a 3-bit BCD counter used to count tens of seconds.
The counter counts from 0 to 5 with a carry over of 1 from the 1-second counter. This counter is reset to 0 after 5 and outputs a carry over of 1 to the 1-minute counter.

D[3:0] RTCSL[3:0]: RTC 1-second Counter Bits

These bits comprise a 4-bit BCD counter used to count units of seconds.
The counter counts from 0 to 9 synchronously with a 1-second signal derived from the 32.768-kHz OSC1 clock. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-second counter.

0x4615: RTC Minute Register (RTC_MIN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Minute Register (RTC_MIN)	0x4615 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–4	RTCMIH[2:0]	RTC 10-minute counter	0 to 5	X	R/W	
		D3–0	RTCMIL[3:0]	RTC 1-minute counter	0 to 9	X	R/W	

Note: Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation.")

D7 Reserved

D[6:4] RTCMIH[2:0]: RTC 10-minute Counter Bits

These bits comprise a 3-bit BCD counter used to count tens of minutes.
The counter counts from 0 to 5 with a carry over of 1 from the 1-minute counter. This counter is reset to 0 after 5 and outputs a carry over of 1 to the 1-hour counter.

D[3:0] RTCMIL[3:0]: RTC 1-minute Counter Bits

These bits comprise a 4-bit BCD counter used to count units of minutes.
The counter counts from 0 to 9 with a carry over of 1 from the 10-second counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-minute counter.

0x4616: RTC Hour Register (RTC_HOUR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Hour Register (RTC_HOUR)	0x4616 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	RTCAP	AM/PM indicator	1 PM 0 AM	X	R/W	
		D5–4	RTCHH[1:0]	RTC 10-hour counter	0 to 2 or 0 to 1	X	R/W	
		D3–0	RTCHL[3:0]	RTC 1-hour counter	0–9	X	R/W	

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation.")

• Rewriting RTC24H (D4/RTC_CNTL0 register) may corrupt the count data in this register. Therefore, after changing the RTC24H setting, be sure to set up this register again.

D7 Reserved

D6 RTCAP: AM/PM Indicator Bit

When 12-hour mode is selected, this bit indicates A.M. or P.M.
1 (R/W): P.M.
0 (R/W): A.M.

This bit is only effective when RTC24H (D4/RTC_CNTL0 register) is set to 0 (12-hour mode).
When 24-hour mode is selected, this bit is fixed to 0. In this case, do not write 1 to RTCAP.

Note: The RTCAP bit keeps the current set value even if RTC24H (D4/RTC_CNTL0 register) is changed from 12-hour mode to 24-hour mode, and will be fixed at 0 after the hour counter is updated (or reset in software).

D[5:4] RTCHH[1:0]: RTC 10-hour Counter Bits

These bits comprise a 2-bit BCD counter used to count tens of hours.
With a carry over of 1 from the 1-hour counter, the counter counts from 0 to 1 when 12-hour mode is selected, or from 0 to 2 when 24-hour mode is selected. The counter is reset at 12 o'clock or 24 o'clock, and outputs a carry over of 1 to the 1-day counter.

D[3:0] RTCHL[3:0]: RTC 1-hour Counter Bits

These bits comprise a 4-bit BCD counter used to count units of hours.
The counter counts from 0 to 9 with a carry over of 1 from the 10-minute counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-hour counter. Depending on whether 12-hour mode or 24-hour mode is selected, the counter is reset at 12 o'clock or 24 o'clock.

0x4617: RTC Day Register (RTC_DAY)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Day Register (RTC_DAY)	0x4617 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5–4	RTCDH[1:0]	RTC 10-day counter	0 to 3	X	R/W	
		D3–0	RTCDL[3:0]	RTC 1-day counter	0 to 9	X	R/W	

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation.")

• Rewriting RTC24H (D4/RTC_CNTL0 register) may corrupt the count data in this register. Therefore, after changing the RTC24H setting, be sure to set up this register again.

D[7:6] Reserved

D[5:4] RTCDH[1:0]: RTC 10-day Counter Bits

These bits comprise a 2-bit BCD counter used to count tens of days. The counter counts from 0 to 2 or 3 with a carry over of 1 from the 1-day counter. The number of days in each month and leap years are taken into account, so that when months change the counter is reset to 0 along with the 1-day counter, and a carry over of 1 is output to the 1-month counter.

D[3:0] RTCDL[3:0]: RTC 1-day Counter Bits

These bits comprise a 4-bit BCD counter used to count units of days.

The counter counts from 0 to 9 with a carry over of 1 from the hour counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-day counter. The number of days in each month and leap years are taken into account, so that the counter is reset to 1 when months change.

0x4628: RTC Month Register (RTC_MONTH)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Month Register (RTC_MONTH)	0x4628 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	RTCMOH	RTC 10-month counter	0 to 1	X	R/W	
		D3–0	RTCMOL[3:0]	RTC 1-month counter	0 to 9	X	R/W	

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation.")

• Rewriting RTC24H (D4/RTC_CNTL0 register) may corrupt the count data in this register. Therefore, after changing the RTC24H setting, be sure to set up this register again.

D[7:5] Reserved

D4 RTCMOH: RTC 10-month Counter Bit

This is a tens of months count bit.

This bit is set to 1 with a carry over of 1 from the 1-month counter. When years change, this bit is reset to 0 along with the 1-month counter, and a carry over of 1 is output to the 1-year counter.

D[3:0] RTCMOL[3:0]: RTC 1-month Counter Bits

These bits comprise a 4-bit BCD counter used to count units of months.

The counter counts from 0 to 9 with a carry over of 1 from the day counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-month counter. The counter is reset to 1 when years change.

0x4629: RTC Year Register (RTC_YEAR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Year Register (RTC_YEAR)	**0x4629** (8 bits)	D7–4	**RTCYH[3:0]**	RTC 10-year counter	0 to 9	X	R/W	
		D3–0	**RTCYL[3:0]**	RTC 1-year counter	0 to 9	X	R/W	

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation.")

• Rewriting RTC24H (D4/RTC_CNTL0 register) may corrupt the count data in this register. Therefore, after changing the RTC24H setting, be sure to set up this register again.

D[7:4] RTCYH[3:0]: RTC 10-year Counter Bits

These bits comprise a 4-bit BCD counter used to count tens of years. The counter counts from 0 to 9 with a carry over of 1 from the 1-year counter.

D[3:0] RTCYL[3:0]: RTC 1-year Counter Bits

These bits comprise a 4-bit BCD counter used to count units of years.
The counter counts from 0 to 9 with a carry over of 1 from the month counter. This counter is reset to 0 after 9 and outputs a carry over of 1 to the 10-year counter.

0x462a: RTC Days of Week Register (RTC_WEEK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
RTC Days of Week Register (RTC_WEEK)	0x462a (8 bits)	D7–3	–	reserved	–		–	–	0 when being read.
		D2–0	RTCWK[2:0]	RTC days of week counter	RTCWK[2:0]	Days of week	X	R/W	
					0x7	–			
					0x6	Saturday			
					0x5	Friday			
					0x4	Thursday			
					0x3	Wednesday			
					0x2	Tuesday			
					0x1	Monday			
					0x0	Sunday			

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation.")

• Rewriting RTC24H (D4/RTC_CNTL0 register) may corrupt the count data in this register. Therefore, after changing the RTC24H setting, be sure to set up this register again.

D[7:3] Reserved

D[2:0] RTCWK[2:0]: RTC Days of Week Counter Bits

This is a septenary counter (that counts from 0 to 6) representing days of the week. This counter counts at the same timing as the 1-day counter.

The correspondence between the counter values and days of the week can be set in a program as desired. Table II.5.7.3 lists the basic correspondence.

Table II.5.7.3 Correspondence between Counter Values and Days of the Week

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

(Default: indeterminate)

II.5.8 Precautions

- The contents of all RTC control registers are indeterminate when power is turned on and are not initialized to specific values by initial reset. Be sure to initialize these registers in software.
- While 1 is being carried over to the next-digit counter, the correct counter value may not be read out. Moreover, attempting to write to the counters or other control registers may corrupt the counter value. Therefore, do not write to the counters while 1 is being carried over. For the correct method of operation, see Section II.5.3.5, "Counter Hold and Busy Flag," and Section II.5.3.6, "Reading from and Writing to Counters in Operation."
- Note that rewriting RTC24H (D4/RTC_CNTL0 register) to switch between 12-hour mode and 24-hour mode may corrupt the count data for hours, days, months, years, or days of the week. Therefore, after changing the RTC24H setting, be sure to set data in these counters back again.
- Avoid the settings below that may cause timekeeping errors.

  - Settings exceeding the effective range
    Do not set count data exceeding 60 seconds, 60 minutes, 12 or 24 hours, 31 days, 12 months, or 99 years.

  - Settings nonexistent in the calendar
    Do not set nonexistent dates such as April 31 or February 29, 2006. Even if such settings are made, the counters operate normally, so that when 1 is carried over from the hour counter to the 1-day counter, the day counter counts up to the first day of the next month. (For April 31, the day counter counts up to May 1; for February 29, 2006, the day counter counts up to March 1, 2006.)

- The contents of all RTC interrupt control bits are indeterminate when power is turned on, and are not initialized to specific values by initial reset.
  After power-on, be sure to set RTCIEN (D0/RTC_INTMODE register) to 0 (interrupt disabled) for preventing the occurrence of unwanted RTC interrupts. Also be sure to write 1 to RTCIRQ (D0/RTC_INTSTAT register) to reset it.
- Immediately after the OSC1 oscillator circuit is activated (as at power-on), a finite time (of about 3 seconds) is required for OSC1 oscillation to stabilize. Do not let the RTC start counting until this time elapses.

III S1C17501 MEMORY CONTROLLER MODULES

III.1 Flash Controller (FLASHC)

III.1.1 Overview of the FLASHC

The S1C17501 incorporates a 128K-byte or 96K-byte Flash memory and a FLASHC (Flash Controller) for accessing the Flash memory. In addition to reading from the Flash memory, the FLASHC supports erasing and programming of the Flash memory from the application program.
The following shows the main features of the on-chip Flash memory and FLASHC.

Flash memory

Memory size	128K bytes (64K × 16 bits) or 96K bytes (48K × 16 bits)
Sector size	512 words (16 bits) / sector
Erase/program time	Chip erase time: 100 ms (typ.) Sector erase time: 20 ms (typ.) Word program time: 15 µs (typ.)
Read access time	50 ns (typ.)
Erase/program interface	Write pulse input type
Reliability	Endurance: 1000 cycles (min.) Data retention: 10 years (min.)

FLASHC

Writing	Supports 16-bit writing only.
Reading	Supports 8-, 16-, and 32-bit reading. Supports burst reading. The S1C17 Core can access the Flash memory with no wait cycle (minimum of one access cycle). If the system clock is faster than the access time, it is necessary to insert a wait cycle (configurable with software).

Notes: • The FLASHC Control Register (0x5800) and Trap Table Base Registers (0x5814–0x5816) are write-protected. Before these registers can be rewritten, their write protection must be removed by writing data 0x96 to the FLASHC Protect Register (0x5810). Note that since unnecessary rewrites to addresses 0x5800 and 0x5814–0x5816 could lead to erratic system operation, the FLASHC Protect Register (0x5810) should be set to other than 0x96 unless the FLASHC Control Register and Trap Table Base Address Low/High Registers must be rewritten.

• The S1C17 Core accesses the stack for writing in 32-bit unit when calling/returning to/from interrupt handlers or subroutines, therefore, a stack cannot be located in the Flash memory area to which data is always written in 16-bit units.

III.1.2 Flash Memory Map

In the 128K-byte Flash model, the Flash memory is located from address 0x20000 to address 0x3ffff.
In the 96K-byte Flash model, the Flash memory is located from address 0x20000 to address 0x37fff.

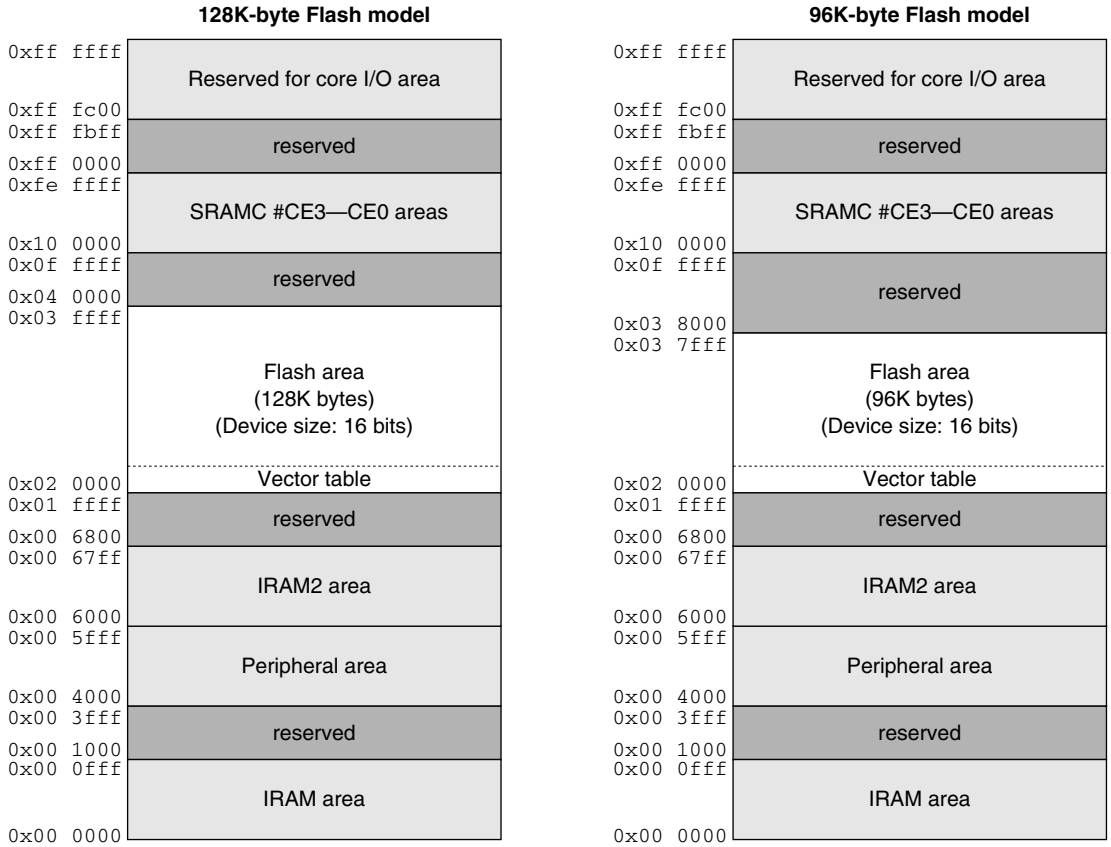


Figure III.1.2.1 Flash Memory Map

III.1.3 Boot Address and TTBR

After an initial reset, the vector table is located at the beginning of the Flash memory (address 0x20000). Therefore, the boot vector must be written to address 0x20000. See Section I.5.4 for the vector table.
The S1C17501 allows the base (starting) address of the vector table to be set using the TTBR_LOW and TTBR_HIGH registers (0x5814, 0x5816) in the FLASHC module. This makes it possible to change the vector table location after the system has booted from address 0x20000 once.
The TTBR_LOW register specifies the low-order 16 bits of the vector table address and the TTBR_HIGH register specifies the high-order 8 bits. Bits 7 to 0 in the TTBR_LOW register are fixed at 0, so the trap table starting address always begins with a 256-byte boundary address.

III.1.4 Programming the Flash Memory

This section explains how to erase and program the Flash memory.

Precautions on Flash programming

(1) The FLASHC Control Register (0x5800) and Trap Table Base Registers (0x5814–0x5816) are write-protected. Before these registers can be rewritten, their write protection must be removed by writing data 0x96 to the FLASHC Protect Register (0x5810). Note that since unnecessary rewrites to addresses 0x5800 and 0x5814–0x5816 could lead to erratic system operation, the FLASHC Protect Register (0x5810) should be set to other than 0x96 unless the FLASHC Control Register and Trap Table Base Registers must be rewritten.

(2) Disable the watchdog timer reset function before starting Flash erase/program operations.

(3) Disable interrupts before starting Flash erase/program operations.

(4) Be sure to fix the #RESET pin at high (inactive) during erasing/programming the Flash memory.

(5) Use the internal RAM or external ROM to store and execute the instructions for erasing/programming the Flash memory.

(6) Make sure that the Flash memory is not busy by reading a FLASHC status flag before starting Flash erase/program operations.

(7) The FLASHC supports only 16-bit writing for programming the Flash memory. The FLASHC will do nothing if an attempt is made to write data to Flash memory in an 8-bit, 24-bit, or 32-bit write instruction.

(8) The S1C17 Core accesses the stack for writing in 32-bit unit when calling/returning to/from interrupt handlers or subroutines, therefore, a stack cannot be located in the Flash memory area to which data is always written in 16-bit units.

(9) Both the minimum and maximum values of each erase/program cycle parameter listed in the table below must be guaranteed with software.

Table III.1.4.1 Erase/Program Cycle Timing Parameters

Parameter	Timing specs		
	Min.	Typ.	Max.
Chip erase operation mode hold time	12 µs	–	–
Sector erase operation mode hold time	6 µs	–	–
Program operation mode hold time	4 µs	–	–
Chip erase pulse width	70 ms	100 ms	500 ms
Sector erase pulse width	15 ms	20 ms	50 ms
Program pulse width	10 µs	15 µs	20 µs

III.1.4.1 Chip Erase Procedure

The following shows a chip erase procedure:

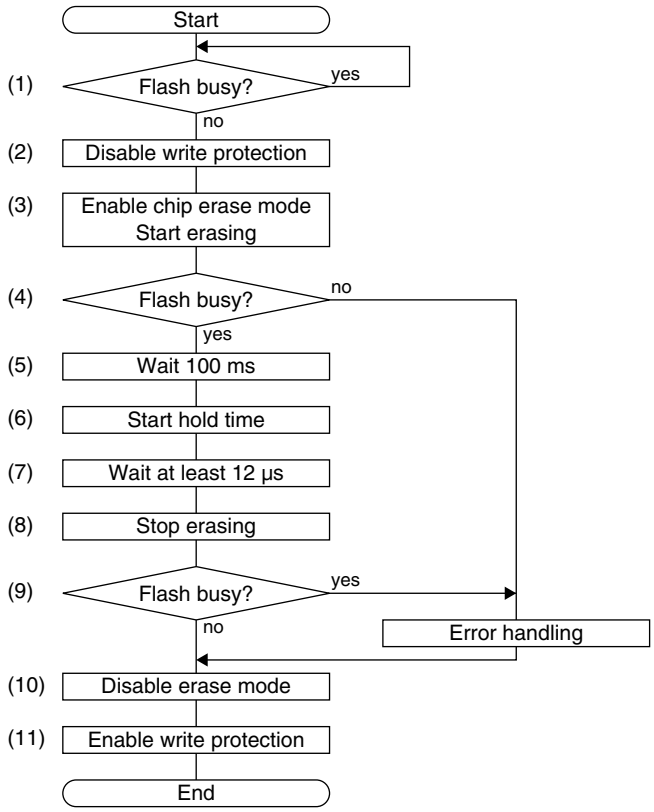


Figure III.1.4.1.1 Chip Erase Flow

(1) Read FLS_STAT (D15/FLASH_CTL register) to check whether the Flash memory is in busy status or not.
If FLS_STAT is 0, the Flash memory is ready to erase. Go to Step (2).
If FLS_STAT is 1, the Flash memory is in busy status such as during reading, erasing, programming, or hardware reset status. In this case, wait until FLS_STAT goes 0.

∗ **FLS_STAT**: Flash Status Flag Bit in the FLASHC Control (FLASH_CTL) Register (D15/0x5800)

(2) Write 0x96 to the FLASHC Protect Register (0x5810) to remove write protection of the FLASHC Control Register (0x5800).

(3) Write 0x104 to the FLASHC Control Register (0x5800). This sets CHIP_ERS_EN (D2/FLASH_CTL register) to 1 to enable chip erase mode and sets START_ERASE (D8/FLASH_CTL register) to 1 to start chip erase operation.

∗ **CHIP_ERS_EN**: Flash Chip Erase Enable Bit in the FLASHC Control (FLASH_CTL) Register (D2/0x5800)
∗ **START_ERASE**: Flash Erasing Start Bit in the FLASHC Control (FLASH_CTL) Register (D8/0x5800)

(4) Read FLS_STAT to check whether the Flash memory is in busy status or not.
When the FLASHC starts erasing, FLS_STAT goes 1. Go to Step (5).
If FLS_STAT is 0, the FLASHC could not start erasing. It may caused by enabling both chip and sector erase modes at the same time or by occurrence of an error in the system. Execute an error recovery routine and go to Step (10).

(5) Wait for 100 ms to generate an appropriate chip erase pulse width.

(6) Write 0x204 to the FLASHC Control Register (0x5800). This sets START_HOLD (D9/FLASH_CTL register) to 1 to start the chip erase operation mode hold period (the #CE signal for the Flash memory becomes inactive). At this time, keep CHIP_ERS_EN (D2/FLASH_CTL register) to 1 to continue chip erase mode.

∗ **START_HOLD**: Hold Period Start Bit in the FLASHC Control (FLASH_CTL) Register (D9/0x5800)

(7) Wait at least 12 µs to generate the chip erase operation mode hold time.

(8) Write 0x404 to the FLASHC Control Register (0x5800). This sets STOP (D10/FLASH_CTL register) to 1 to finish chip erasing.

∗ **STOP**: Flash Erase/Program Stop Bit in the FLASHC Control (FLASH_CTL) Register (D10/0x5800)

(9) Read FLS_STAT to check whether the Flash memory is in busy status or not.
If FLS_STAT is 0, the chip erase operation has finished normally. Go to Step (10).
If FLS_STAT is 1, an error has occurred during erasing. Execute an error recovery routine and go to Step (10).

(10) Write 0x0 to the FLASHC Control Register (0x5800). This resets CHIP_ERS_EN (D2/FLASH_CTL register) to 0 to disable chip erase mode.

(11) Write a value other than 0x96 to the FLASHC Protect Register (0x5810) to enable write protection of the FLASHC Control Register (0x5800).

III.1.4.2 Sector Erase Procedure

The following shows a sector erase procedure:

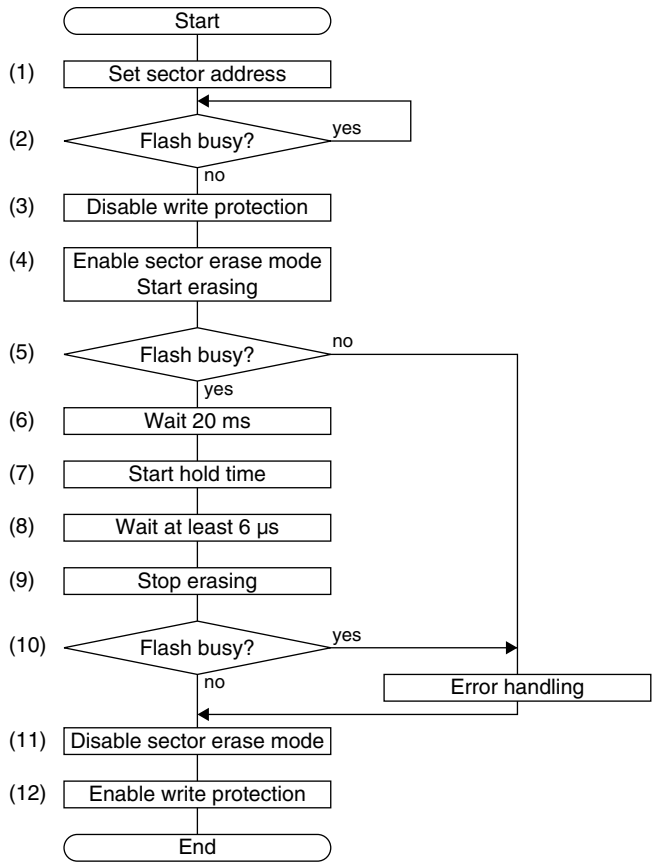


Figure III.1.4.2.1 Sector Erase Flow

(1) Set the sector address to be erased to the FLASHC Sector Address Register (0x5802).
Address[16:10] should be set to this register as the sector address.

Table III.1.4.2.1 Corresponding Between Memory Address and Flash Sector

Memory address	Sector address	Memory address	Sector address	Memory address	Sector address	Memory address	Sector address
0x20000–0x203ff	0x0	0x28000–0x283ff	0x20	0x30000–0x303ff	0x40	0x38000–0x383ff	0x60
0x20400–0x207ff	0x1	0x28400–0x287ff	0x21	0x30400–0x307ff	0x41	0x38400–0x387ff	0x61
0x20800–0x20bff	0x2	0x28800–0x28bff	0x22	0x30800–0x30bff	0x42	0x38800–0x38bff	0x62
0x20c00–0x20fff	0x3	0x28c00–0x28fff	0x23	0x30c00–0x30fff	0x43	0x38c00–0x38fff	0x63
0x21000–0x213ff	0x4	0x29000–0x293ff	0x24	0x31000–0x313ff	0x44	0x39000–0x393ff	0x64
0x21400–0x217ff	0x5	0x29400–0x297ff	0x25	0x31400–0x317ff	0x45	0x39400–0x397ff	0x65
0x21800–0x21bff	0x6	0x29800–0x29bff	0x26	0x31800–0x31bff	0x46	0x39800–0x39bff	0x66
0x21c00–0x21fff	0x7	0x29c00–0x29fff	0x27	0x31c00–0x31fff	0x47	0x39c00–0x39fff	0x67
0x22000–0x223ff	0x8	0x2a000–0x2a3ff	0x28	0x32000–0x323ff	0x48	0x3a000–0x3a3ff	0x68
0x22400–0x227ff	0x9	0x2a400–0x2a7ff	0x29	0x32400–0x327ff	0x49	0x3a400–0x3a7ff	0x69
0x22800–0x22bff	0xa	0x2a800–0x2abff	0x2a	0x32800–0x32bff	0x4a	0x3a800–0x3abff	0x6a
0x22c00–0x22fff	0xb	0x2ac00–0x2afff	0x2b	0x32c00–0x32fff	0x4b	0x3ac00–0x3afff	0x6b
0x23000–0x233ff	0xc	0x2b000–0x2b3ff	0x2c	0x33000–0x333ff	0x4c	0x3b000–0x3b3ff	0x6c
0x23400–0x237ff	0xd	0x2b400–0x2b7ff	0x2d	0x33400–0x337ff	0x4d	0x3b400–0x3b7ff	0x6d
0x23800–0x23bff	0xe	0x2b800–0x2bbff	0x2e	0x33800–0x33bff	0x4e	0x3b800–0x3bbff	0x6e
0x23c00–0x23fff	0xf	0x2bc00–0x2bfff	0x2f	0x33c00–0x33fff	0x4f	0x3bc00–0x3bfff	0x6f
0x24000–0x243ff	0x10	0x2c000–0x2c3ff	0x30	0x34000–0x343ff	0x50	0x3c000–0x3c3ff	0x70
0x24400–0x247ff	0x11	0x2c400–0x2c7ff	0x31	0x34400–0x347ff	0x51	0x3c400–0x3c7ff	0x71
0x24800–0x24bff	0x12	0x2c800–0x2cbff	0x32	0x34800–0x34bff	0x52	0x3c800–0x3cbff	0x72
0x24c00–0x24fff	0x13	0x2cc00–0x2cfff	0x33	0x34c00–0x34fff	0x53	0x3cc00–0x3cfff	0x73
0x25000–0x253ff	0x14	0x2d000–0x2d3ff	0x34	0x35000–0x353ff	0x54	0x3d000–0x3d3ff	0x74
0x25400–0x257ff	0x15	0x2d400–0x2d7ff	0x35	0x35400–0x357ff	0x55	0x3d400–0x3d7ff	0x75
0x25800–0x25bff	0x16	0x2d800–0x2dbff	0x36	0x35800–0x35bff	0x56	0x3d800–0x3dbff	0x76
0x25c00–0x25fff	0x17	0x2dc00–0x2dfff	0x37	0x35c00–0x35fff	0x57	0x3dc00–0x3dfff	0x77
0x26000–0x263ff	0x18	0x2e000–0x2e3ff	0x38	0x36000–0x363ff	0x58	0x3e000–0x3e3ff	0x78
0x26400–0x267ff	0x19	0x2e400–0x2e7ff	0x39	0x36400–0x367ff	0x59	0x3e400–0x3e7ff	0x79
0x26800–0x26bff	0x1a	0x2e800–0x2ebff	0x3a	0x36800–0x36bff	0x5a	0x3e800–0x3ebff	0x7a
0x26c00–0x26fff	0x1b	0x2ec00–0x2efff	0x3b	0x36c00–0x36fff	0x5b	0x3ec00–0x3efff	0x7b
0x27000–0x273ff	0x1c	0x2f000–0x2f3ff	0x3c	0x37000–0x373ff	0x5c	0x3f000–0x3f3ff	0x7c
0x27400–0x277ff	0x1d	0x2f400–0x2f7ff	0x3d	0x37400–0x377ff	0x5d	0x3f400–0x3f7ff	0x7d
0x27800–0x27bff	0x1e	0x2f800–0x2fbff	0x3e	0x37800–0x37bff	0x5e	0x3f800–0x3fbff	0x7e
0x27c00–0x27fff	0x1f	0x2fc00–0x2ffff	0x3f	0x37c00–0x37fff	0x5f	0x3fc00–0x3ffff	0x7f

(2) Read FLS_STAT (D15/FLASH_CTL register) to check whether the Flash memory is in busy status or not.
If FLS_STAT is 0, the Flash memory is ready to erase. Go to Step (3).
If FLS_STAT is 1, the Flash memory is in busy status such as during reading, erasing, programming, or hardware reset status. In this case, wait until FLS_STAT goes 0.

* **FLS_STAT**: Flash Status Flag Bit in the FLASHC Control (FLASH_CTL) Register (D15/0x5800)

(3) Write 0x96 to the FLASHC Protect Register (0x5810) to remove write protection of the FLASHC Control Register (0x5800).

(4) Write 0x102 to the FLASHC Control Register (0x5800). This sets SCT_ERS_EN (D1/FLASH_CTL register) to 1 to enable sector erase mode and sets START_ERASE (D8/FLASH_CTL register) to 1 to start sector erase operation.

* **SCT_ERS_EN**: Flash Sector Erase Enable Bit in the FLASHC Control (FLASH_CTL) Register (D1/0x5800)
* **START_ERASE**: Flash Erasing Start Bit in the FLASHC Control (FLASH_CTL) Register (D8/0x5800)

(5) Read FLS_STAT to check whether the Flash memory is in busy status or not.
When the FLASHC starts erasing, FLS_STAT goes 1. Go to Step (6).
If FLS_STAT is 0, the FLASHC could not start erasing. It may caused by enabling both chip and sector erase modes at the same time or by occurrence of an error in the system. Execute an error recovery routine and go to Step (11).

(6) Wait for 20 ms to generate an appropriate sector erase pulse width.

(7) Write 0x202 to the FLASHC Control Register (0x5800). This sets START_HOLD (D9/FLASH_CTL register) to 1 to start the sector erase operation mode hold period (the #CE signal for the Flash memory becomes inactive). At this time, keep SCT_ERS_EN (D1/FLASH_CTL register) to 1 to continue sector erase mode.

∗ **START_HOLD**: Hold Period Start Bit in the FLASHC Control (FLASH_CTL) Register (D9/0x5800)

(8) Wait at least 6 µs to generate the sector erase operation mode hold time.

(9) Write 0x402 to the FLASHC Control Register (0x5800). This sets STOP (D10/FLASH_CTL register) to 1 to finish sector erasing.

∗ **STOP**: Flash Erase/Program Stop Bit in the FLASHC Control (FLASH_CTL) Register (D10/0x5800)

(10) Read FLS_STAT to check whether the Flash memory is in busy status or not.
If FLS_STAT is 0, the sector erase operation has finished normally. Go to Step (11).
If FLS_STAT is 1, an error has occurred during erasing. Execute an error recovery routine and go to Step (11).

(11) Write 0x0 to the FLASHC Control Register (0x5800). This resets SCT_ERS_EN (D1/FLASH_CTL register) to 0 to disable sector erase mode.

(12) Write a value other than 0x96 to the FLASHC Protect Register (0x5810) to enable write protection of the FLASHC Control Register (0x5800).

III.1.4.3 Flash Programming Procedure

The following shows a Flash programming (writing) procedure:

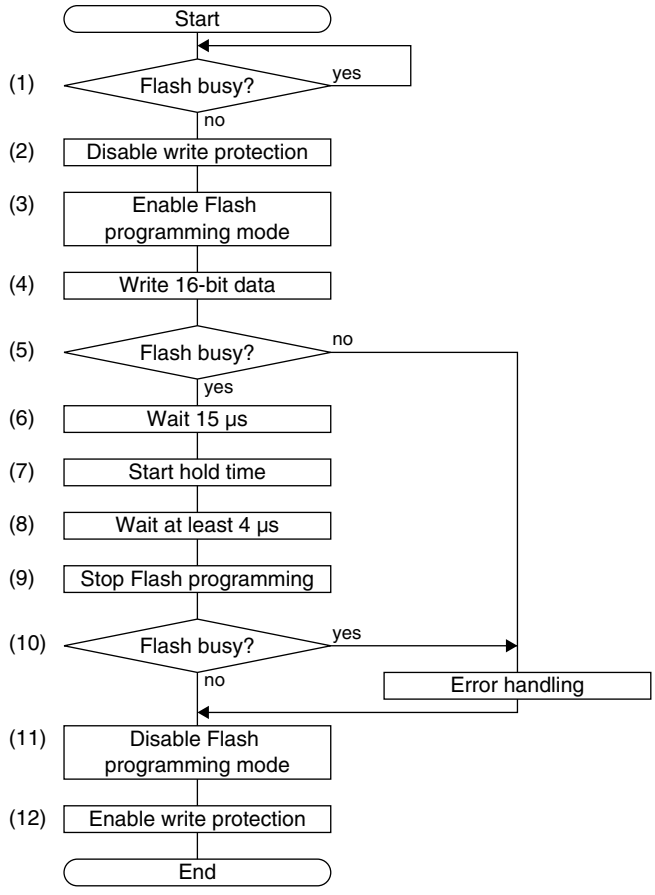


Figure III.1.4.3.1 Flash Programming Flow

(1) Read FLS_STAT (D15/FLASH_CTL register) to check whether the Flash memory is in busy status or not.
If FLS_STAT is 0, the Flash memory is ready to program. Go to Step (2).
If FLS_STAT is 1, the Flash memory is in busy status such as during reading, erasing, programming, or hardware reset status. In this case, wait until FLS_STAT goes 0.

∗ **FLS_STAT**: Flash Status Flag Bit in the FLASHC Control (FLASH_CTL) Register (D15/0x5800)

(2) Write 0x96 to the FLASHC Protect Register (0x5810) to remove write protection of the FLASHC Control Register (0x5800).

(3) Write 0x1 to the FLASHC Control Register (0x5800). This sets WR_EN (D0/FLASH_CTL register) to 1 to enable Flash programming mode.

∗ **WR_EN**: Flash Programming Enable Bit in the FLASHC Control (FLASH_CTL) Register (D0/0x5800)

(4) Write 16-bit data using a 16-bit data transfer instruction.
Do not to use an 8-bit, 24-bit, or 32-bit data transfer instruction.

(5) Read FLS_STAT to check whether the Flash memory is in busy status or not.
When the FLASHC starts programming, FLS_STAT goes 1. Go to Step (6).
If FLS_STAT is 0, the FLASHC could not start programming. It may caused by an error in the system. Execute an error recovery routine and go to Step (11).

(6) Wait for 15 μs to generate an appropriate Flash program pulse width.

(7) Write 0x201 to the FLASHC Control Register (0x5800). This sets START_HOLD (D9/FLASH_CTL register) to 1 to start the program operation mode hold period (the #CE signal for the Flash memory becomes inactive). At this time, keep WR_EN (D0/FLASH_CTL register) to 1 to continue Flash programming mode.

∗ **START_HOLD**: Hold Period Start Bit in the FLASHC Control (FLASH_CTL) Register (D9/0x5800)

(8) Wait at least 4 μs to generate the program operation mode hold time.

(9) Write 0x401 to the FLASHC Control Register (0x5800). This sets STOP (D10/FLASH_CTL register) to 1 to finish Flash programming.

∗ **STOP**: Flash Erase/Program Stop Bit in the FLASHC Control (FLASH_CTL) Register (D10/0x5800)

(10) Read FLS_STAT to check whether the Flash memory is in busy status or not.
If FLS_STAT is 0, the Flash programming operation has finished normally. Go to Step (11).
If FLS_STAT is 1, an error has occurred during programming. Execute an error recovery routine and go to Step (11).

(11) Write 0x0 to the FLASHC Control Register (0x5800). This resets WR_EN (D0/FLASH_CTL register) to 0 to disable Flash programming mode.

(12) Write a value other than 0x96 to the FLASHC Protect Register (0x5810) to enable write protection of the FLASHC Control Register (0x5800).

III.1.5 Read Access Control

In order to read data from the Flash memory properly even if the system operates with a high-speed clock, the FLASHC can insert a wait cycle in the Flash read cycle. The number of system clock cycles to be inserted as a wait cycle can be specified using FLS_WAIT[2:0] (D[2:0]/FLASH_WAIT register). Set the appropriate number of cycles according to the system clock frequency.

* **FLS_WAIT[2:0]**: Flash Read Access Wait Cycle Setup Bits in the FLASHC Wait (FLASH_WAIT) Register (D[2:0]/0x5804)

Table III.1.5.1 Setting Read Access Wait Cycle

FLS_WAIT[2:0]	Number of wait cycles	Number of read access cycles	System clock frequency
0x7	7 cycles	8 cycles	48 MHz or less
0x6	6 cycles	7 cycles	48 MHz or less
0x5	5 cycles	6 cycles	48 MHz or less
0x4	4 cycles	5 cycles	48 MHz or less
0x3	3 cycles	4 cycles	48 MHz or less
0x2	2 cycles	3 cycles	48 MHz or less
0x1	1 cycle	2 cycles	40 MHz or less
0x0	0 cycles	1 cycle	20 MHz or less

(Default: 0x7)

Note: The variation of the Flash read access cycles with the different FLS_WAIT[2:0] settings cannot be monitored from outside the IC as the FLASHC includes a Flash memory pre-fetch circuit. Therefore, be sure to avoid setting of FLS_WAIT[2:0] to a value that is not supported in the system clock frequency.

III.1.6 Details of Control Registers

Table III.1.6.1 List of FLASHC Registers

Address	Register name		Function
0x5800	FLASH_CTL	FLASHC Control Register	Controls Flash erase/program operations.
0x5802	FLASH_ADDR	FLASHC Sector Address Register	Sets the Flash address for erasing a sector.
0x5804	FLASH_WAIT	FLASHC Wait Register	Sets the wait cycle for Flash read.
0x5810	FLASH_PROT	FLASHC Protect Register	Enables Flash control registers for writing.
0x5814	TTBR_LOW	Trap Table Base Register 0	Sets the vector table address.
0x5816	TTBR_HIGH	Trap Table Base Register 1	

The following describes each FLASHC control register. These are all 16-bit registers.

Note: When setting the registers, be sure to write 0, and not 1, for all “reserved bits.”

0x5800: FLASHC Control Register (FLASH_CTL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FLASHC Control Register (FLASH_CTL)	0x5800 (16 bits)	D15	FLS_STAT	Flash status flag	1	Busy	0	Idle	1	R	
		D14–11	–	reserved	–				–	–	0 when being read.
		D10	STOP	Flash erase/program stop	1	Stop	0	Ignored	0	W	
		D9	START_HOLD	Hold period start	1	Start	0	Ignored	0	W	
		D8	START_ERASE	Flash erasing start	1	Start	0	Ignored	0	W	
		D7–3	–	reserved	–				–	–	
		D2	CHIP_ERS_EN	Flash chip erase enable	1	Enable	0	Disable	0	R/W	
		D1	SCT_ERS_EN	Flash sector erase enable	1	Enable	0	Disable	0	R/W	
		D0	WR_EN	Flash programming enable	1	Enable	0	Disable	0	R/W	

Note: The FLASHC Control Register is write-protected. Before this register can be rewritten, write protection must be removed by writing data 0x96 to the FLASHC Protect Register (0x5810). Note that since unnecessary rewrites to the FLASHC Control Register could lead to erratic system operation, the FLASHC Protect Register (0x5810) should be set to other than 0x96 unless the FLASHC Control Register must be rewritten.

D15 FLS_STAT: Flash Status Flag Bit

Indicates whether the Flash memory is idle or in busy status.

1 (R): Busy (default)
0 (R): Idle
1/0 (W): Has no effect

This flag is set to 1 while the Flash memory is being read, erased, programmed, or the hardware reset sequence is in progress.
Before the Flash memory can be erased or programmed, this flag must be 0. To place the Flash memory in Idle state, the CPU must run with the instructions stored in a memory other than Flash (e.g. internal RAM or external ROM).

D[14:11] Reserved

D10 STOP: Flash Erase/Program Stop Bit

Stops chip erase, sector erase, or Flash programming operation.

1 (W): Stop
0 (W): Has no effect
0 (R): Always 0 when read (default)

Writing 1 to STOP terminates the chip erase, sector erase, or program operation. This bit should be set to 1 when the operation mode hold time has elapsed after setting START_HOLD (D9) to 1.
This bit must be fixed at 0 while the Flash memory is being read.

D9 START_HOLD: Hold Period Start Bit

Starts the operation mode hold period.

1 (W): Start
0 (W): Has no effect
0 (R): Always 0 when read (default)

Write 1 to START_HOLD when the chip erase, sector erase, or program time (see Table III.1.6.2) has elapsed after setting 1 to START_ERASE (D10) or writing data to the Flash memory.
Writing 1 to START_HOLD starts the operation mode hold period by setting the #CE signal for the Flash memory inactive. After that, wait for operation mode hold time (see Table III.1.6.2) before terminating chip erase, sector erase, or program mode.
This bit must be fixed at 0 while the Flash memory is being read.

Table III.1.6.2 Erase/Program Cycle Timing Parameters

Parameter	Timing specs		
	Min.	Typ.	Max.
Chip erase operation mode hold time	12 µs	–	–
Sector erase operation mode hold time	6 µs	–	–
Program operation mode hold time	4 µs	–	–
Chip erase pulse width	70 ms	100 ms	500 ms
Sector erase pulse width	15 ms	20 ms	50 ms
Program pulse width	10 µs	15 µs	20 µs

D8 START_ERASE: Flash Erasing Start Bit

Starts chip erase or sector erase operation.
1 (W): Start
0 (W): Has no effect
0 (R): Always 0 when read (default)

When performing chip erasing, set CHIP_ERS_EN (D2) and START_ERASE to 1.
When performing sector erasing, set SCT_ERS_EN (D1) and START_ERASE to 1 after setting the sector address to be erased to the FLASHC Sector Address Register (0x5802).
If CHIP_ERS_EN and SCT_ERS_EN are both set to 0 or 1, writing 1 to START_ERASE is ignored and it does not start erasing.
Do not set this bit to 1 when the Flash memory is in busy status (FLS_STAT (D15) = 1).

D[7:3] Reserved

D2 CHIP_ERS_EN: Flash Chip Erase Enable Bit

Enables chip erase mode.
1 (R/W): Enable
0 (R/W): Disable (default)

Set CHIP_ERS_EN and START_ERASE (D8) to 1 to start chip erasing. After the chip erase operation is terminated by setting STOP (D10) to 1, reset CHIP_ERS_EN to 0. See Section III.1.4.1, "Chip Erase Procedure," for controlling chip erase operation.

D1 SCT_ERS_EN: Flash Sector Erase Enable Bit

Enables sector erase mode.
1 (R/W): Enable
0 (R/W): Disable (default)

Set SCT_ERS_EN and START_ERASE (D8) to 1 to start sector erasing after setting the sector address to be erased to the FLASHC Sector Address Register (0x5802). After the sector erase operation is terminated by setting STOP (D10) to 1, reset SCT_ERS_EN to 0. See Section III.1.4.2, "Sector Erase Procedure," for controlling sector erase operation.

D0 WR_EN: Flash Programming Enable Bit

Enables Flash programming mode.
1 (R/W): Enable
0 (R/W): Disable (default)

Set WR_EN to 1 to start Flash programming mode. After that write data using a 16-bit data transfer instruction. After the Flash programming operation is terminated by setting STOP (D10) to 1, reset WR_EN to 0. See Section III.1.4.3, "Flash Programming Procedure," for controlling Flash programming operation.

0x5802: FLASHC Sector Address Register (FLASH_ADDR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FLASHC Sector Address Register (FLASH_ADDR)	**0x5802** (16 bits)	D15–7	**–**	reserved	–	–	–	0 when being read.
		D6–0	**FLS_ADDR [6:0]**	Erase sector address	Address[16:10]	0x0	R/W	

D[15:7] Reserved

D[6:0] FLS_ADDR[6:0]: Erase Sector Address Bits

Specifies the sector to be erased. (Default: 0x0)

Specify bit 16 to bit 10 of the memory address as the sector address.

Table III.1.6.3 Corresponding Between Memory Address and Flash Sector

Memory address	Sector address	Memory address	Sector address	Memory address	Sector address	Memory address	Sector address
0x20000–0x203ff	0x0	0x28000–0x283ff	0x20	0x30000–0x303ff	0x40	0x38000–0x383ff	0x60
0x20400–0x207ff	0x1	0x28400–0x287ff	0x21	0x30400–0x307ff	0x41	0x38400–0x387ff	0x61
0x20800–0x20bff	0x2	0x28800–0x28bff	0x22	0x30800–0x30bff	0x42	0x38800–0x38bff	0x62
0x20c00–0x20fff	0x3	0x28c00–0x28fff	0x23	0x30c00–0x30fff	0x43	0x38c00–0x38fff	0x63
0x21000–0x213ff	0x4	0x29000–0x293ff	0x24	0x31000–0x313ff	0x44	0x39000–0x393ff	0x64
0x21400–0x217ff	0x5	0x29400–0x297ff	0x25	0x31400–0x317ff	0x45	0x39400–0x397ff	0x65
0x21800–0x21bff	0x6	0x29800–0x29bff	0x26	0x31800–0x31bff	0x46	0x39800–0x39bff	0x66
0x21c00–0x21fff	0x7	0x29c00–0x29fff	0x27	0x31c00–0x31fff	0x47	0x39c00–0x39fff	0x67
0x22000–0x223ff	0x8	0x2a000–0x2a3ff	0x28	0x32000–0x323ff	0x48	0x3a000–0x3a3ff	0x68
0x22400–0x227ff	0x9	0x2a400–0x2a7ff	0x29	0x32400–0x327ff	0x49	0x3a400–0x3a7ff	0x69
0x22800–0x22bff	0xa	0x2a800–0x2abff	0x2a	0x32800–0x32bff	0x4a	0x3a800–0x3abff	0x6a
0x22c00–0x22fff	0xb	0x2ac00–0x2afff	0x2b	0x32c00–0x32fff	0x4b	0x3ac00–0x3afff	0x6b
0x23000–0x233ff	0xc	0x2b000–0x2b3ff	0x2c	0x33000–0x333ff	0x4c	0x3b000–0x3b3ff	0x6c
0x23400–0x237ff	0xd	0x2b400–0x2b7ff	0x2d	0x33400–0x337ff	0x4d	0x3b400–0x3b7ff	0x6d
0x23800–0x23bff	0xe	0x2b800–0x2bbff	0x2e	0x33800–0x33bff	0x4e	0x3b800–0x3bbff	0x6e
0x23c00–0x23fff	0xf	0x2bc00–0x2bfff	0x2f	0x33c00–0x33fff	0x4f	0x3bc00–0x3bfff	0x6f
0x24000–0x243ff	0x10	0x2c000–0x2c3ff	0x30	0x34000–0x343ff	0x50	0x3c000–0x3c3ff	0x70
0x24400–0x247ff	0x11	0x2c400–0x2c7ff	0x31	0x34400–0x347ff	0x51	0x3c400–0x3c7ff	0x71
0x24800–0x24bff	0x12	0x2c800–0x2cbff	0x32	0x34800–0x34bff	0x52	0x3c800–0x3cbff	0x72
0x24c00–0x24fff	0x13	0x2cc00–0x2cfff	0x33	0x34c00–0x34fff	0x53	0x3cc00–0x3cfff	0x73
0x25000–0x253ff	0x14	0x2d000–0x2d3ff	0x34	0x35000–0x353ff	0x54	0x3d000–0x3d3ff	0x74
0x25400–0x257ff	0x15	0x2d400–0x2d7ff	0x35	0x35400–0x357ff	0x55	0x3d400–0x3d7ff	0x75
0x25800–0x25bff	0x16	0x2d800–0x2dbff	0x36	0x35800–0x35bff	0x56	0x3d800–0x3dbff	0x76
0x25c00–0x25fff	0x17	0x2dc00–0x2dfff	0x37	0x35c00–0x35fff	0x57	0x3dc00–0x3dfff	0x77
0x26000–0x263ff	0x18	0x2e000–0x2e3ff	0x38	0x36000–0x363ff	0x58	0x3e000–0x3e3ff	0x78
0x26400–0x267ff	0x19	0x2e400–0x2e7ff	0x39	0x36400–0x367ff	0x59	0x3e400–0x3e7ff	0x79
0x26800–0x26bff	0x1a	0x2e800–0x2ebff	0x3a	0x36800–0x36bff	0x5a	0x3e800–0x3ebff	0x7a
0x26c00–0x26fff	0x1b	0x2ec00–0x2efff	0x3b	0x36c00–0x36fff	0x5b	0x3ec00–0x3efff	0x7b
0x27000–0x273ff	0x1c	0x2f000–0x2f3ff	0x3c	0x37000–0x373ff	0x5c	0x3f000–0x3f3ff	0x7c
0x27400–0x277ff	0x1d	0x2f400–0x2f7ff	0x3d	0x37400–0x377ff	0x5d	0x3f400–0x3f7ff	0x7d
0x27800–0x27bff	0x1e	0x2f800–0x2fbff	0x3e	0x37800–0x37bff	0x5e	0x3f800–0x3fbff	0x7e
0x27c00–0x27fff	0x1f	0x2fc00–0x2ffff	0x3f	0x37c00–0x37fff	0x5f	0x3fc00–0x3ffff	0x7f

0x5804: FLASHC Wait Register (FLASH_WAIT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FLASHC Wait Register (FLASH_WAIT)	0x5804 (16 bits)	D15–3	–	reserved	–		–	–	0 when being read.
		D2–0	FLS_WAIT [2:0]	Flash read access wait cycle setup	FLS_WAIT[2:0]	Wait cycle	0x7	R/W	
					0x7 : 0x0	7 cycles : 0 cycles			

D[15:3] Reserved

D[2:0] FLS_WAIT[2:0]: Flash Read Access Wait Cycle Setup Bits

These bits set the number of wait cycles to be inserted when the Flash memory is read.

Table III.1.6.4 Setting Read Access Wait Cycle

FLS_WAIT[2:0]	Number of wait cycles	Number of read access cycles	System clock frequency
0x7	7 cycles	8 cycles	48 MHz or less
0x6	6 cycles	7 cycles	48 MHz or less
0x5	5 cycles	6 cycles	48 MHz or less
0x4	4 cycles	5 cycles	48 MHz or less
0x3	3 cycles	4 cycles	48 MHz or less
0x2	2 cycles	3 cycles	48 MHz or less
0x1	1 cycle	2 cycles	40 MHz or less
0x0	0 cycles	1 cycle	20 MHz or less

(Default: 0x7)

The number of wait cycles should be set according to the system clock frequency.

Note: The variation of the Flash read access cycles with the different FLS_WAIT[2:0] settings cannot be monitored from outside the IC as the FLASHC includes a Flash memory pre-fetch circuit. Therefore, be sure to avoid setting of FLS_WAIT[2:0] to a value that is not supported in the system clock frequency.

0x5810: FLASHC Protect Register (FLASH_PROT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FLASHC Protect Register (FLASH_PROT)	**0x5810** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**FLS_PROT [7:0]**	FLASHC register protect flag	Writing 10010110 (0x96) removes the write protection of the FLASHC Control Register (0x5800) and Trap Table Base Registers (0x5814–0x5816). Writing another value set the write protection.	0x0	R/W	

D[15:8] Reserved

D[7:0] FLS_PROT[7:0]: FLASHC Register Protect Flag Bits

Enables/disables write protection of the FLASHC registers (0x5800, 0x5814–0x5816).
0x96 (R/W): Disable write protection
Other than 0x96 (R/W): Write-protect the register (default: 0x0)

Before altering the FLASHC Control Register (0x5800) or Trap Table Base Registers (0x5814–0x5816), write data 0x96 to the FLASHC Protect Register to disable write protection. If this register is set to other than 0x96, even if an attempt is made to alter the FLASHC Control Register or Trap Table Base Registers by executing a write instruction, the content of said register will not be altered even though the instruction may have been executed without a problem. Once this register is set to 0x96, the FLASHC Control Register or Trap Table Base Registers can be rewritten any number of times until being reset to other than 0x96. When rewriting the FLASHC Control Register or Trap Table Base Registers has finished, this register should be set to other than 0x96 to prevent accidental writing to the FLASHC Control Register or Trap Table Base Registers.

0x5814–0x5816: Trap Table Base Registers (TTBR_LOW, TTBR_HIGH)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Trap Table Base Register 0 (TTBR_LOW)	**0x5814** (16 bits)	D15–8	**TTBR[15:8]**	Trap table base address A[15:8]	0x0–0xff	0x0	R/W	
		D7–0	**TTBR[7:0]**	Trap table base address A[7:0] (fixed at 0)	0x0	0x0	R	
Trap Table Base Register 1 (TTBR_HIGH)	**0x5816** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	**TTBR[23:16]**	Trap table base address A[23:16]	0x0–0xff	0x2	R/W	

Note: The Trap Table Base Registers are write-protected. Before these registers can be rewritten, write protection must be removed by writing data 0x96 to the FLASHC Protect Register (0x5810). Note that since unnecessary rewrites to the Trap Table Base Registers could lead to erratic system operation, the FLASHC Protect Register (0x5810) should be set to other than 0x96 unless the Trap Table Base Registers must be rewritten.

D[15:0]/0x5814, D[7:0]/0x5816 TTBR[23:16]: Trap Table Base Address Bits

These registers are used to set the starting address of the vector table.

After an initial reset, the TTBR_LOW/HIGH registers are set to 0x20000. Therefore, even when the trap table position is changed, it is necessary that at least the reset vector be written to the above address. Bits 7 to 0 in the TTBR_LOW register are fixed at 0, so the trap table starting address always begins with a 256-byte boundary address.

III.1.7 Precautions

(1) The FLASHC Control Register (0x5800) and Trap Table Base Registers (0x5814–0x5816) are write-protected. Before these registers can be rewritten, their write protection must be removed by writing data 0x96 to the FLASHC Protect Register (0x5810). Note that since unnecessary rewrites to addresses 0x5800 and 0x5814–0x5816 could lead to erratic system operation, the FLASHC Protect Register (0x5810) should be set to other than 0x96 unless the FLASHC Control Register and Trap Table Base Registers must be rewritten.

(2) Disable the watchdog timer reset function before starting Flash erase/program operations.

(3) Disable interrupts before starting Flash erase/program operations.

(4) Be sure to fix the #RESET pin at high (inactive) during erasing/programming the Flash memory.

(5) Use the internal RAM or external ROM to store and execute the instructions for erasing/programming the Flash memory.

(6) Make sure that the Flash memory is not busy by reading a FLASHC status flag before starting Flash erase/program operations.

(7) The FLASHC supports only 16-bit writing for programming the Flash memory. The FLASHC will do nothing if an attempt is made to write data to Flash memory in an 8-bit, 24-bit, or 32-bit write instruction.

(8) The S1C17 Core accesses the stack for writing in 32-bit unit when calling/returning to/from interrupt handlers or subroutines, therefore, a stack cannot be located in the Flash memory area to which data is always written in 16-bit units.

(9) Both the minimum and maximum values of each erase/program cycle parameter listed in the table below must be guaranteed with software.

Table III.1.7.1 Erase/Program Cycle Timing Parameters

Parameter	Timing specs		
	Min.	Typ.	Max.
Chip erase operation mode hold time	12 µs	–	–
Sector erase operation mode hold time	6 µs	–	–
Program operation mode hold time	4 µs	–	–
Chip erase pulse width	70 ms	100 ms	500 ms
Sector erase pulse width	15 ms	20 ms	50 ms
Program pulse width	10 µs	15 µs	20 µs

(10) When the halt instruction located in the Flash memory is executed, the S1C17501 enters HALT mode with the Flash area chip select signal asserted. This increases current consumption, as the Flash memory stays active during HALT status. Therefore, when setting the S1C17501 into HALT mode, the halt instruction should be executed in IRAM.

(11) The variation of the Flash read access cycles with the different FLS_WAIT[2:0] settings cannot be monitored from outside the IC as the FLASHC includes a Flash memory pre-fetch circuit. Therefore, be sure to avoid setting of FLS_WAIT[2:0] to a value that is not supported in the system clock frequency.

∗ **FLS_WAIT[2:0]**: Flash Read Access Wait Cycle Setup Bits in the FLASHC Wait (FLASH_WAIT) Register (D[2:0]/0x5804)

III.2 SRAM Controller (SRAMC)

III.2.1 Overview of the SRAMC

The SRAM Controller (SRAMC) is a bus controller module for accessing internal extended peripheral modules, IRAM2, and external devices.
The SRAMC functions and features are outlined below.

Extended internal peripheral interface

- 8-bit/16-bit selectable data bus
- Supports 8-bit, 16-bit, or 24-bit access.
- 0 wait access (4-cycle bus access)
- Little endian (fixed)

IRAM2 interface

- 16-bit data bus
- Supports 8-bit, 16-bit, or 24-bit access.
- Allows the CPU to fetch instructions from the IRAM2 in addition to read/write data.
- Write pulse width: 1 cycle
- Read pulse width: 1 or 2 cycles (software selectable)
- Little endian (fixed)

External memory interface

- 23-bit address bus and 16-bit data bus.
- Outputs four chip-enable signals (#CE0 to #CE3) for external devices.
- Supports two access types for each #CE area: A0 and BSL.
- Supports two device size for each #CE area: 8 bits and 16 bits.
- Supports static wait cycle insertion (0 to 15 cycles, software selectable) for each #CE area.
- Allows SRAM, ROM, or Flash memory to be connected directly to the external bus.
- Allows wait states to be inserted from the external #WAIT pin (for SRAM type only).
- Little endian (fixed)

III.2.2 SRAMC Pins

Table III.2.2.1 lists the pins used by the SRAMC.

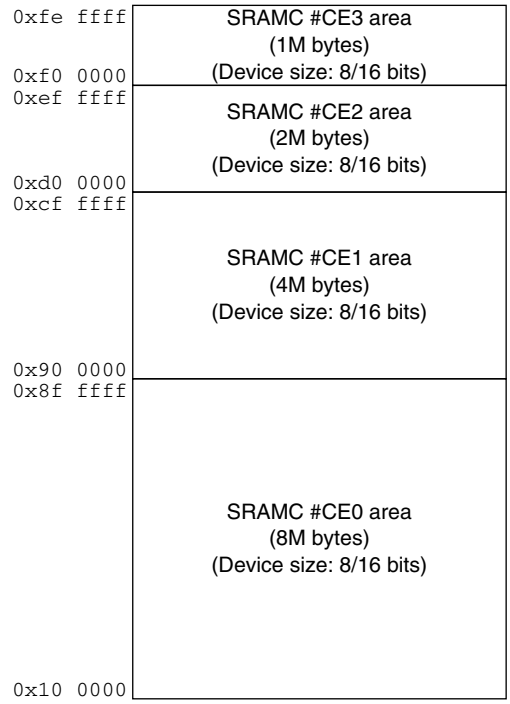
Table III.2.2.1 SRAMC Pin List

Pin name	I/O	Function
D[15:0]	I/O	External data bus D[15:0]
A0/#BSL	O	External address bus A0 / Bus strobe (low byte) signal output
A[22:1]	O	External address bus A[22:1]
#CE[3:0]	O	Chip enable signal outputs
#RD	O	Read signal output
#WRL	O	Write (low byte) signal output
#WRH/#BSH	O	Write (high byte) signal / Bus strobe (high byte) signal output

Note: The external bus control pins above are shared with general-purpose I/O ports and they are configured for I/O ports at initial reset. Before the SRAMC signals assigned to these pins can be used, the functions of these pins must be switched for the SRAMC by setting each corresponding Port Function Select Register.
For details on how to switch over the pin functions, see Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

III.2.3 External Memory Area

The SRAMC supports an external memory space, which is divided into four areas as shown in Figure III.2.3.1.



∗1 A NAND Flash can be connected to #CE2 area only.

Figure III.2.3.1 External Memory Space of the S1C17501

III.2.3.1 Chip Enable Signals

The S1C17501 provides 23 bits of an external address bus, 16 bits of an external data bus, and four chip-enable pins (#CE0 to #CE3), allowing access to the 15MB address space.

Table III.2.3.1.1 Chip Enable Signals

#CE pin	Address range	Size
#CE0	0x100000–0x8fffff	8MB
#CE1	0x900000–0xcfffff	4MB
#CE2	0xd00000–0xefffff	2MB
#CE3	0xf00000–0xfeffff	1MB

When using an external area, enable the corresponding #CE output by setting the corresponding Port Function Select Register (see Section I.3.3).

III.2.3.2 Area Condition Settings

Bus access conditions can be set by area for each #CE*x* signal.
This section describes the parameters to be set individually for each area and the relevant control bits.

Endian mode

The S1C17501 supports little endian mode only. When using an 8-bit external device, its data lines should be connected to the D[7:0] pins.

Device type

The SRAMC incorporates an SRAM-type bus interface, allowing A0 (default) or BSL to be selected as the device type. To use a BSL-type device in the #CE area, set CE*x*TYPE (D*x*/EXTMEM_A0_BSL register) to 1.

- **CE3TYPE**: #CE3 Device Type Selection Bit in the External Memory Device Type Setup (EXTMEM_A0_BSL) Register (D3/0x5008)
- **CE2TYPE**: #CE2 Device Type Selection Bit in the External Memory Device Type Setup (EXTMEM_A0_BSL) Register (D2/0x5008)
- **CE1TYPE**: #CE1 Device Type Selection Bit in the External Memory Device Type Setup (EXTMEM_A0_BSL) Register (D1/0x5008)
- **CE0TYPE**: #CE0 Device Type Selection Bit in the External Memory Device Type Setup (EXTMEM_A0_BSL) Register (D0/0x5008)

Table III.2.3.2.2 lists the bus control signal pins used in each device type.

Table III.2.3.2.2 Bus Control Signal Pins Used in A0 and BSL Modes

Pin name	A0 (default)	BSL
#CE*x*	#CE*x*	#CE*x*
#RD	#RD	#RD
A0/#BSL	Unused	#BSL
#WRL/#WR	#WRL	#WR
#WRH/#BSH	#WRH	#BSH

Device size

Use CE*x*SIZE (D*x*/EXTMEM_SIZE register) to select a device size.

- **CE3SIZE**: #CE3 Device Size Selection Bit in the External Memory Device Size Setup (EXTMEM_SIZE) Register (D3/0x5004)
- **CE2SIZE**: #CE2 Device Size Selection Bit in the External Memory Device Size Setup (EXTMEM_SIZE) Register (D2/0x5004)
- **CE1SIZE**: #CE1 Device Size Selection Bit in the External Memory Device Size Setup (EXTMEM_SIZE) Register (D1/0x5004)
- **CE0SIZE**: #CE0 Device Size Selection Bit in the External Memory Device Size Setup (EXTMEM_SIZE) Register (D0/0x5004)

At an initial reset, the device size is initialized to 16 bits (CE*x*SIZE = 1). When an 8-bit device is used, set CE*x*SIZE to 0.

Static wait cycle

If the number of static wait cycles is specified, the chip enable and read/write signals are always prolonged for the number of specified cycles when the area is accessed. Set up the wait cycle according to the specifications of the device connected to the area using CE*x*WAIT[3:0] (EXTMEM_SWAIT register).

- **CE3WAIT[3:0]**: #CE3 Static Wait Cycle Setup Bits in the External Memory Device Static Wait Control (EXTMEM_SWAIT) Register (D[15:12]/0x5000)
- **CE2WAIT[3:0]**: #CE2 Static Wait Cycle Setup Bits in the External Memory Device Static Wait Control (EXTMEM_SWAIT) Register (D[11:8]/0x5000)
- **CE1WAIT[3:0]**: #CE1 Static Wait Cycle Setup Bits in the External Memory Device Static Wait Control (EXTMEM_SWAIT) Register (D[7:4]/0x5000)
- **CE0WAIT[3:0]**: #CE0 Static Wait Cycle Setup Bits in the External Memory Device Static Wait Control (EXTMEM_SWAIT) Register (D[3:0]/0x5000)

Table III.2.3.2.3 Setting the Static Wait Cycle

CE*x*WAIT[3:0]	Number of wait cycles
0xf	15 cycles
0xe	14 cycles
0xd	13 cycles
0xc	12 cycles
0xb	11 cycles
0xa	10 cycles
0x9	9 cycles
:	:
0x2	2 cycles
0x1	1 cycle
0x0	0 cycles

(Default: 0xf)

At initial reset, the static wait conditions for all external areas are set to 15 cycles.
The area to which an SRAM device is connected allows dynamic wait control using the #WAIT pin in addition to the static wait control.

III.2.4 Connection of External Devices and Bus Operation

III.2.4.1 Connecting External Devices

The following shows an example of connecting the S1C17501 and SRAM.

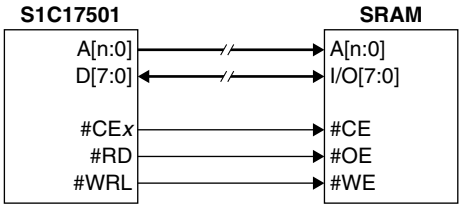


Figure III.2.4.1.1 Example of 8-bit SRAM Connection with 8-bit Device Size

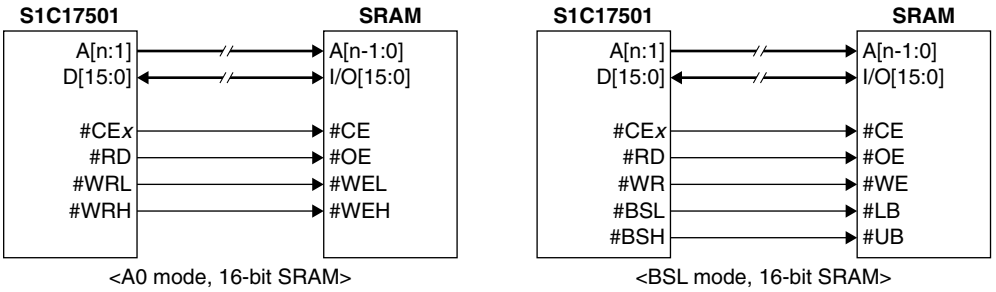


Figure III.2.4.1.2 Example of 16-bit SRAM Connection with 16-bit Device Size

III.2.4.2 Data Configuration in Memory

The S1C17501 SRAMC handles 8-bit, 16-bit, and 24/32-bit data. To access data in memory, addresses aligned to the boundary of the data size must be specified. Specifying other addresses generates address misaligned interrupts. Instructions (e.g., stack manipulating and branch instructions) that rewrite the content of the Stack Pointer (SP) or Program Counter (PC) forcibly alter the address specified to a boundary address to prevent address misaligned interrupts. For details of address misaligned exceptions, refer to the S1C17 Core Manual.
Table III.2.4.2.1 shows where each type of data is located in memory.

Table III.2.4.2.1 Data Locations in Memory

Data type	Location
8-bit data	8-bit boundary (all addresses)
16-bit data	16-bit boundary (A[0] = 0)
24/32-bit data	32-bit boundary (A[1:0] = 0b00)

All 16-bit and 24/32-bit data in memory are accessed in little endian mode. To increase memory efficiency, try locating the same type of data at contiguous addresses to reduce blank areas created by positioning at boundary addresses as much as possible.

III.2.4.3 External Bus Operation

The internal data bus size in the S1C17501 is 32 bits. Note, however, that it has 16 external bus pins D[15:0]. Depending on the device size and data size of the instruction executed, two or more bus operations may occur. Table III.2.4.3.1 shows bus operation in A0 and BSL modes.
For details on how to connect memory, see Section III.2.4.1, "Connecting External Devices."

Table III.2.4.3.1 Bus Operation

Device size	Data size	R/W	A1	A0	A0 mode			BSL mode			Access count
					Valid signal	D[15:8] pins	D[7:0] pins	Valid signal	D[15:8] pins	D[7:0] pins	
8 bits	8 bits	W	*	*	#WRL	–	D[7:0]	–	–	–	1
		R	*	*	#RD	–	D[7:0]	–	–	–	1
	16 bits	W	*	0	#WRL	–	D[7:0]	–	–	–	1st
			*	1		–	D[15:8]		–	–	2nd
		R	*	0	#RD	–	D[7:0]	–	–	–	1st
			*	1		–	D[15:8]		–	–	2nd
	24/32 bits	W	0	0	#WRL	–	D[7:0]	–	–	–	1st
			0	1		–	D[15:8]		–	–	2nd
			1	0		–	D[23:16]		–	–	3rd
			1	1		–	D[31:24]		–	–	4th
		R	0	0	#RD	–	D[7:0]	–	–	–	1st
			0	1		–	D[15:8]		–	–	2nd
			1	0		–	D[23:16]		–	–	3rd
			1	1		–	D[31:24]		–	–	4th
16 bits	8 bits	W	*	0	#WRL	–	D[7:0]	#WR #BSL	–	D[7:0]	1
			*	1	#WRH	D[7:0]	–	#WR #BSH	D[7:0]	–	1
		R	*	0	#RD	–	D[7:0]	#RD #BSL	–	D[7:0]	1
			*	1		D[7:0]	–	#RD #BSH	D[7:0]	–	1
	16 bits	W	*	0	#WRH #WRL	D[15:0]		#WR #BSH #BSL	D[15:0]		1
		R	*	0	#RD	D[15:0]		#RD #BSH #BSL	D[15:0]		1
	24/32 bits	W	0	0	#WRH #WRL	D[15:0]		#WR #BSH #BSL	D[15:0]		1st
			1	0		D[31:16]			D[31:16]		2nd
		R	0	0	#RD	D[15:0]		#RD #BSH #BSL	D[15:0]		1st
			1	0		D[31:16]			D[31:16]		2nd

Handling the eight high-order bits during 32-bit memory accesses

During writing, the eight high-order bits are written as 0. During reading from a memory, the eight high-order bits are ignored. However, the eight high-order bits are effective as the PSR value only in the stack operation when an interrupt occurs.

III.2.5 SRAMC Operating Clock

The SRAMC is clocked by the SRAMC_CLK clock (= system clock) generated by the CMU.
The bus control signals are generated synchronously with SRAMC_CLK.
The SRAMC provides SRAMC_CLK_EN (D7/CMU_GATEDCLK1 register) for controlling the SRAMC clock (SRAMC_CLK). However, the SRAMC controls the internal peripheral bus (SAPB) and external bus, so SRAMC_CLK cannot be stopped while the IC is running. In other words, SRAMC_CLK does not stop in normal operation mode (except when the halt or slp instruction is executed) even if SRAMC_CLK_EN is set to 0. However, SRAMC_CLK can be automatically turned off in HALT mode (after the halt instruction is executed) by setting SRAMC_CLK_EN to 0 (default: on).

* **SRAMC_CLK_EN**: SRAMC Clock Control (in HALT mode) Bit in the Gated Clock Control 1 (CMU_GATEDCLK1) Register (D7/0x4907)

When initially reset, SRAMC_CLK_EN is set to 1 (on) to enable the SRAMC_CLK supply. If SRAMC_CLK is unused in HALT mode, set SRAMC_CLK_EN to 0 (off). The SRAMC_CLK supply will be stopped after the CPU executes the halt instruction. After that, the SRAMC_CLK supply will be enabled when HALT mode is cancelled by an interrupt or an other cause. SRAMC_CLK will stop in SLEEP mode regardless of how the CMU registers are set (see Section II.2.8.2 for more information).

The same clock can also be output to the external device from the CMU_CLK pin.

For details on how to set and control the SRAMC operating clock and CMU_CLK output, see Section II.2, “Clock Management Unit (CMU).”

III.2.6 Bus Access Timing Chart

III.2.6.1 SRAM Read/Write Timings with No External #WAIT

1. SRAM read/write timings with no static wait cycles

[Example settings]
Device size: 16 bits
Access size: 16 bits
Number of static wait cycles: 0 cycles

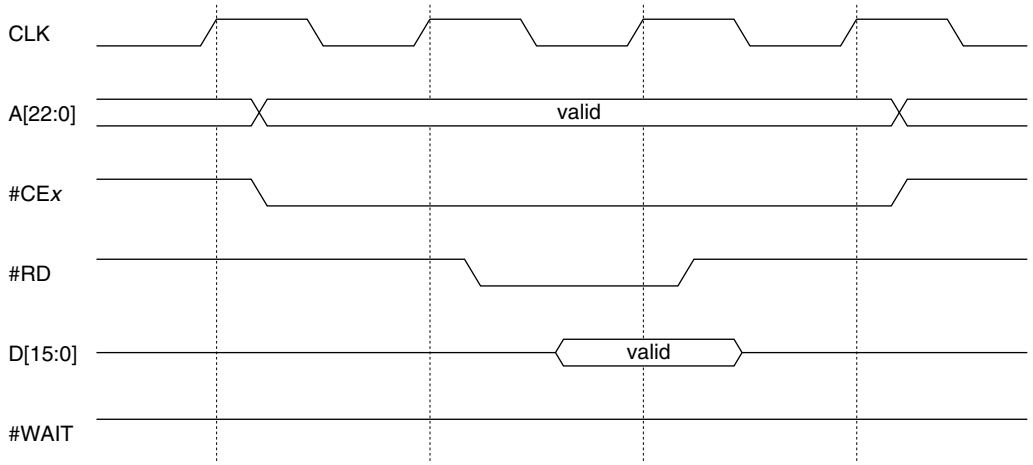


Figure III.2.6.1.1 SRAM Read Timing with No Static Wait Cycle

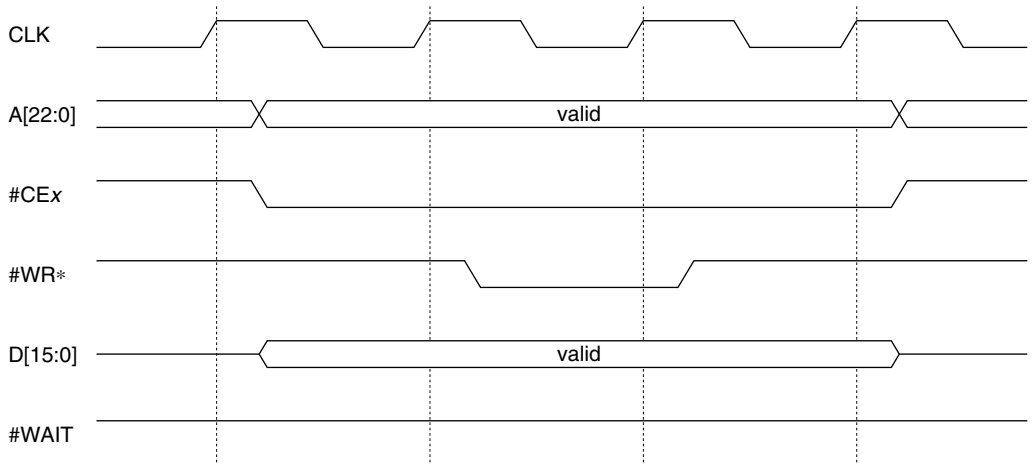


Figure III.2.6.1.2 SRAM Write Timing with No Static Wait Cycle

2. SRAM read/write timings with static wait cycles

[Example settings]
Device size: 16 bits
Access size: 16 bits
Number of static wait cycles: 2 cycles

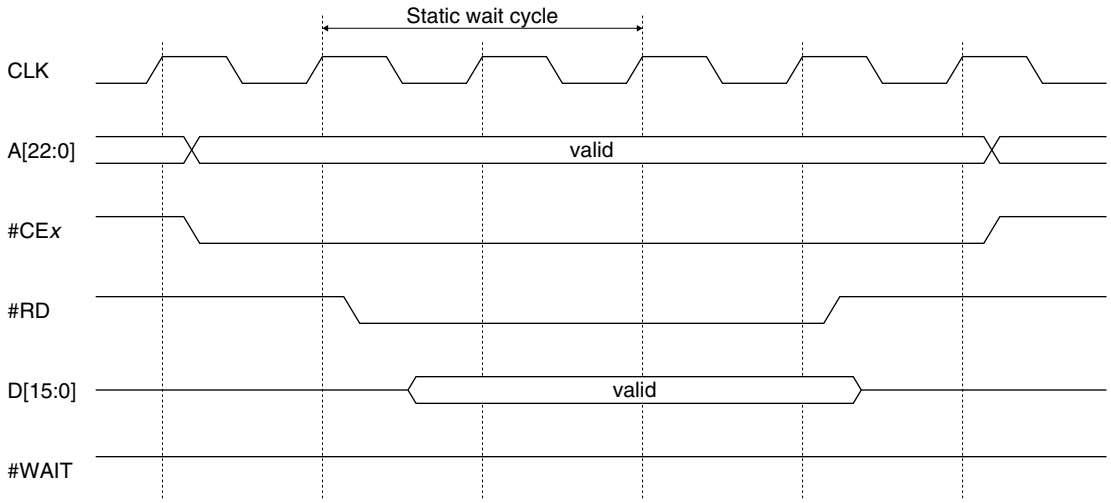


Figure III.2.6.1.3 SRAM Read Timing with Static Wait Cycle

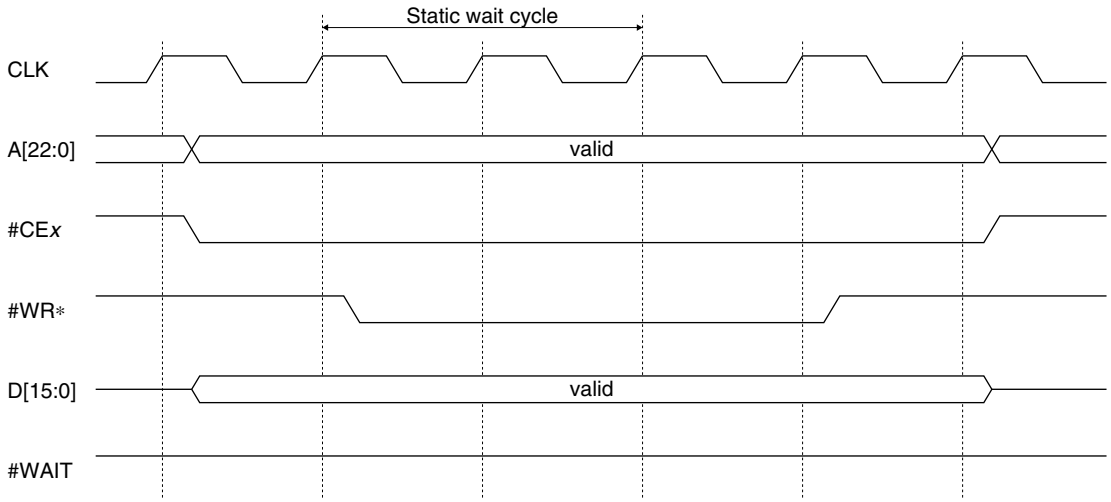


Figure III.2.6.1.4 SRAM Write Timing with Static Wait Cycle

III.2.6.2 SRAM Read/Write Timings with External #WAIT

A wait cycle can be inserted from the #WAIT pin only for SRAM-type devices.
The external #WAIT signal is sampled on the rising edges of the clock at one clock before the read or write signal goes high. A wait state is entered while the #WAIT signal is sampled active (low), and subsequent operation resumes when the #WAIT signal is sampled inactive (high).

[Example settings]
Device size: 16 bits
Access size: 16 bits
Number of static wait cycles: 0 cycles

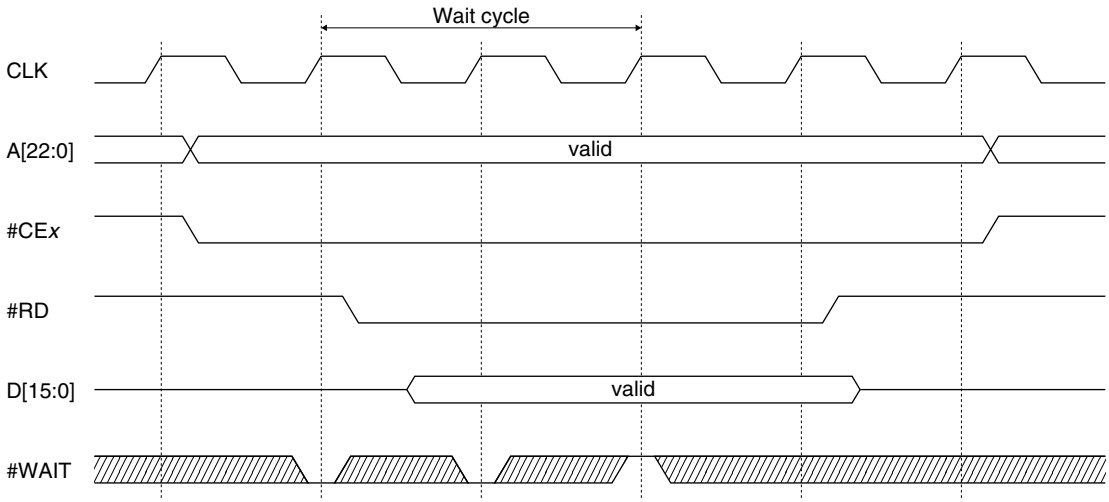


Figure III.2.6.2.1 SRAM Read Timing with External #WAIT

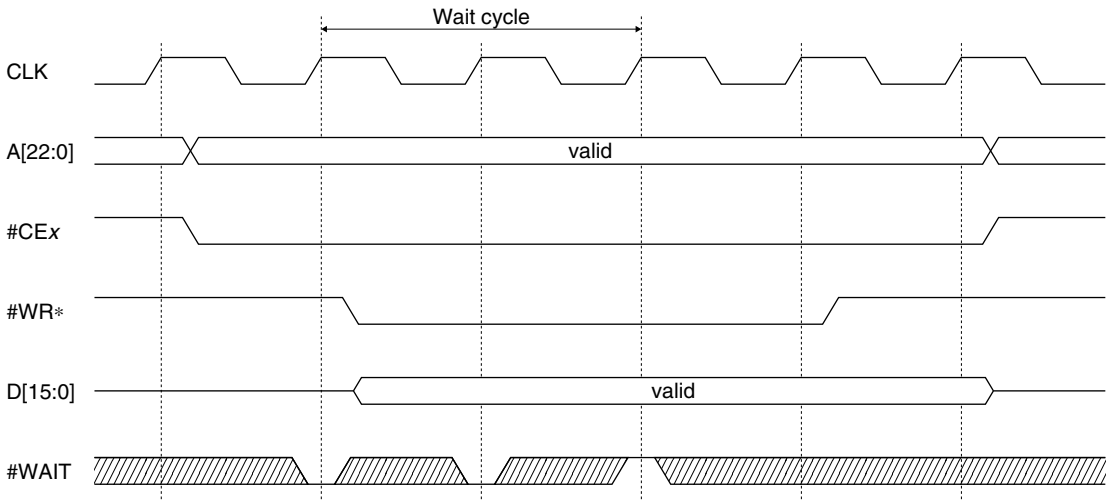


Figure III.2.6.2.2 SRAM Write Timing with External #WAIT

III.2.7 Setting Wait Cycles for Internal Devices

The SRAMC provides wait cycle setup registers for accessing the IRAM2, MAC operation module, and RTC module.

Setting the wait cycle for reading the IRAM2

The CPU can write data to the 2KB IRAM2 with 0 wait cycles (8-bit or 16-bit write needs 4 cycles). On the other hand, the IRAM2 can be read from the CPU with 0 or 1 wait cycle inserted (16-bit read needs 4 + wait cycles).

By default, the IRAM2 is read with 1 wait cycle. If the system clock frequency is lower than 48 MHz, select 0 wait cycles by setting IRAM2_WAIT (D0/IRAM2_SWAIT register) to 0.

* **IRAM2_WAIT**: IRAM2 Static Wait Cycle Setup Bit in the Internal Memory Static Wait Control (IRAM2_SWAIT) Register (D0/0x5010)

Setting the wait cycles for accessing the MAC module

The MAC module can be accessed with 0 or 1 wait cycle (1 or 2 access cycles).

By default, the MAC module is read/write with 1 wait cycle. If the system clock frequency is lower than 24 MHz, select 0 wait cycles by setting MAC_WAIT (D0/MAC_WAIT register) to 0.

* **MAC_WAIT**: MAC Wait Cycle Setup Bit in the MAC Wait Control (MAC_WAIT) Register (D0/0x5014)

Setting the wait cycles for accessing the RTC module

In order to access the RTC registers properly even if the system operates with a high-speed clock, the SRAMC can insert a wait cycle in the RTC access cycle. The number of system clock cycles to be inserted as a wait cycle can be specified using RTC_WAIT[2:0] (D[2:0]/RTC_WAIT register).

* **RTC_WAIT[2:0]**: RTC Access Wait Cycle Setup Bits in the RTC Wait Control (RTC_WAIT) Register (D[2:0]/0x5018)

Table III.2.7.1 Number of Wait Cycles during RTC Access

RTC_WAIT[2:0]	Number of wait cycles
0x7	7 cycles
0x6	6 cycles
0x5	5 cycles
0x4	4 cycles
0x3	3 cycles
0x2	2 cycles
0x1	1 cycle
0x0	0 cycles

(Default: 0x7)

The S1C17501 is able to operate with RTC_WAIT[2:0] ≥ 1.

III.2.8 Control Register Details

Table III.2.8.1 SRAMC Register List

Address	Register name		Function
0x5000	EXTMEM_SWAIT	External Memory Static Wait Control Register	Sets up static wait cycles.
0x5004	EXTMEM_SIZE	External Memory Device Size Setup Register	Selects the device size (8/16 bits).
0x5008	EXTMEM_A0_BSL	External Memory Device Type Setup Register	Selects the device type (A0/BSL).
0x5010	IRAM2_SWAIT	Internal Memory Static Wait Control Register	Sets up IRAM2 read cycle.
0x5014	MAC_WAIT	MAC Wait Control Register	Sets up the MAC wait cycle.
0x5018	RTC_WAIT	RTC Wait Control Register	Sets up RTC access cycle.

The following describes each SRAMC control register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x5000: External Memory Static Wait Control Register (EXTMEM_SWAIT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
External Memory Static Wait Control Register (EXTMEM_SWAIT)	0x5000 (16 bits)	D15–12	CE3WAIT[3:0]	#CE3 static wait cycle setup	CE*x*WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
		D11–8	CE2WAIT[3:0]	#CE2 static wait cycle setup	0xe	14 cycles	0xf	R/W	
					0xd	13 cycles			
		D7–4	CE1WAIT[3:0]	#CE1 static wait cycle setup	:	:	0xf	R/W	
					0x2	2 cycles			
		D3–0	CE0WAIT[3:0]	#CE0 static wait cycle setup	0x1	1 cycle	0xf	R/W	
					0x0	0 cycles			

D[15:12] CE3WAIT[3:0]: #CE3 Static Wait Cycle Setup Bits

These bits set the static wait cycle for accessing the #CE3 area.

Table III.2.8.2 Setting the Static Wait Cycle

CE*x*WAIT[3:0]	Number of wait cycles
0xf	15 cycles
0xe	14 cycles
0xd	13 cycles
0xc	12 cycles
0xb	11 cycles
0xa	10 cycles
0x9	9 cycles
:	:
0x2	2 cycles
0x1	1 cycle
0x0	0 cycles

(Default: 0xf)

D[11:8] CE2WAIT[3:0]: #CE2 Static Wait Cycle Setup Bits

These bits set the static wait cycle for accessing the #CE2 area.

D[7:4] CE1WAIT[3:0]: #CE1 Static Wait Cycle Setup Bits

These bits set the static wait cycle for accessing the #CE1 area.

D[3:0] CE0WAIT[3:0]: #CE0 Static Wait Cycle Setup Bits

These bits set the static wait cycle for accessing the #CE0 area.

0x5004: External Memory Device Size Setup Register (EXTMEM_SIZE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Memory Device Size Setup Register (EXTMEM_SIZE)	0x5004 (16 bits)	D15–4	–	reserved	–	–	–	0 when being read.
		D3	CE3SIZE	#CE3 device size selection	1 16 bits 0 8 bits	1	R/W	
		D2	CE2SIZE	#CE2 device size selection	1 16 bits 0 8 bits	1	R/W	
		D1	CE1SIZE	#CE1 device size selection	1 16 bits 0 8 bits	1	R/W	
		D0	CE0SIZE	#CE0 device size selection	1 16 bits 0 8 bits	1	R/W	

D[15:4] Reserved

D3 CE3SIZE: #CE3 Device Size Selection Bit

This bit selects the device size for the #CE3 area.
1 (R/W): 16 bits (default)
0 (R/W): 8 bits

D2 CE2SIZE: #CE2 Device Size Selection Bit

This bit selects the device size for the #CE2 area.
1 (R/W): 16 bits (default)
0 (R/W): 8 bits

D1 CE1SIZE: #CE1 Device Size Selection Bit

This bit selects the device size for the #CE1 area.
1 (R/W): 16 bits (default)
0 (R/W): 8 bits

D0 CE0SIZE: #CE0 Device Size Selection Bit

This bit selects the device size for the #CE0 area.
1 (R/W): 16 bits (default)
0 (R/W): 8 bits

0x5008: External Memory Device Type Setup Register (EXTMEM_A0_BSL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Memory Device Type Setup Register (EXTMEM_A0_BSL)	0x5008 (16 bits)	D15–4	–	reserved	–	–	–	0 when being read.
		D3	CE3TYPE	#CE3 device type selection	1 BSL 0 A0	0	R/W	
		D2	CE2TYPE	#CE2 device type selection	1 BSL 0 A0	0	R/W	
		D1	CE1TYPE	#CE1 device type selection	1 BSL 0 A0	0	R/W	
		D0	CE0TYPE	#CE0 device type selection	1 BSL 0 A0	0	R/W	

D[15:4] Reserved

D3 CE3TYPE: #CE3 Device Type Selection Bit

This bit selects a device type (A0 or BSL) for the #CE3 area.
1 (R/W): BSL
0 (R/W): A0 (default)

Table III.2.8.3 Bus Control Signal Pin Functions in A0/BSL Mode

Pin name	A0 (default)	BSL
#CE*x*	#CE*x*	#CE*x*
#RD	#RD	#RD
A0/#BSL	Unused	#BSL
#WRL/#WR	#WRL	#WR
#WRH/#BSH	#WRH	#BSH

D2 CE2TYPE: #CE2 Device Type Selection Bit

This bit selects a device type (A0 or BSL) for the #CE2 area.
1 (R/W): BSL
0 (R/W): A0 (default)

D1 CE1TYPE: #CE1 Device Type Selection Bit

This bit selects a device type (A0 or BSL) for the #CE1 area.
1 (R/W): BSL
0 (R/W): A0 (default)

D0 CE0TYPE: #CE0 Device Type Selection Bit

This bit selects a device type (A0 or BSL) for the #CE0 area.
1 (R/W): BSL
0 (R/W): A0 (default)

0x5010: Internal Memory Static Wait Control Register (IRAM2_SWAIT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Internal Memory Static Wait Control Register (IRAM2_SWAIT)	0x5010 (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	IRAM2_WAIT	IRAM2 static wait cycle setup	1 1 cycle 0 0 cycles	1	R/W	

D[15:1] Reserved

D0 IRAM2_WAIT: IRAM2 Static Wait Cycle Setup Bit

This bit selects the number of wait cycles to be inserted when the IRAM2 is read.
1 (R/W): 1 cycle (default)
0 (R/W): 0 cycles

If the system clock frequency is lower than 48 MHz, 0 wait cycles can be selected.

0x5014: MAC Wait Control Register (MAC_WAIT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MAC Wait Control Register (MAC_WAIT)	0x5014 (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	MAC_WAIT	MAC wait cycle setup	1 1 cycle 0 0 cycles	1	R/W	

D[15:1] Reserved

D0 MAC_WAIT: MAC Wait Cycle Setup Bit

This bit selects the number of wait cycles to be inserted when the MAC module is accessed.
1 (R/W): 1 cycle (default)
0 (R/W): 0 cycles

If the system clock frequency is lower than 24 MHz, 0 wait cycles can be selected.

0x5018: RTC Wait Control Register (RTC_WAIT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
RTC Wait Control Register (RTC_WAIT)	0x5018 (16 bits)	D15–3	–	reserved	–		–	–	0 when being read.
		D2–0	RTC_WAIT [2:0]	RTC access wait cycle setup	RTC_WAIT[2:0]	Wait cycle	0x7	R/W	
					0x7 : 0x0	7 cycles : 0 cycles			

D[15:3] Reserved

D[2:0] RTC_WAIT[2:0]: RTC Access Wait Cycle Setup Bits

These bits set the number of wait cycles to be inserted when an RTC register is accessed.

Table III.2.8.5 Number of Wait Cycles during RTC Access

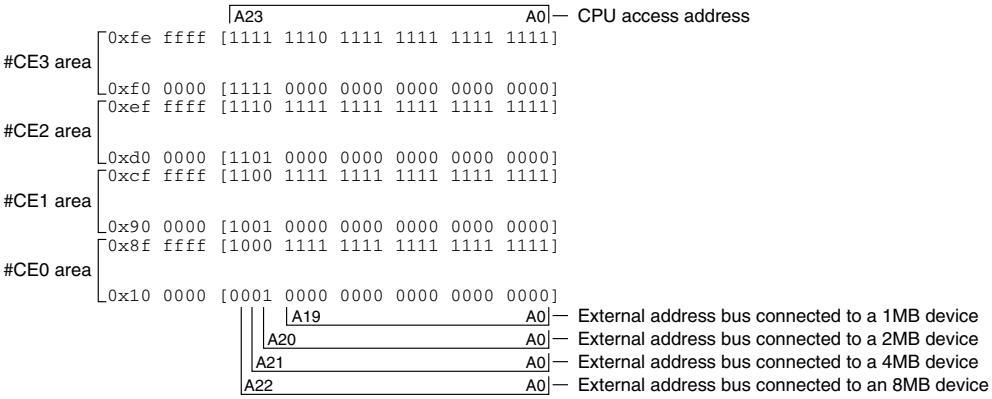
RTC_WAIT[2:0]	Number of wait cycles
0x7	7 cycles
0x6	6 cycles
0x5	5 cycles
0x4	4 cycles
0x3	3 cycles
0x2	2 cycles
0x1	1 cycle
0x0	0 cycles

(Default: 0x7)

The S1C17501 is able to operate with RTC_WAIT[2:0] ≥ 1.

III.2.9 Precautions

- When the CPU accesses an external memory area (#CE0 to #CE3 areas, addresses 0x100000 to 0xfefff) in the S1C17501 memory space, the external bus outputs the address as follows:



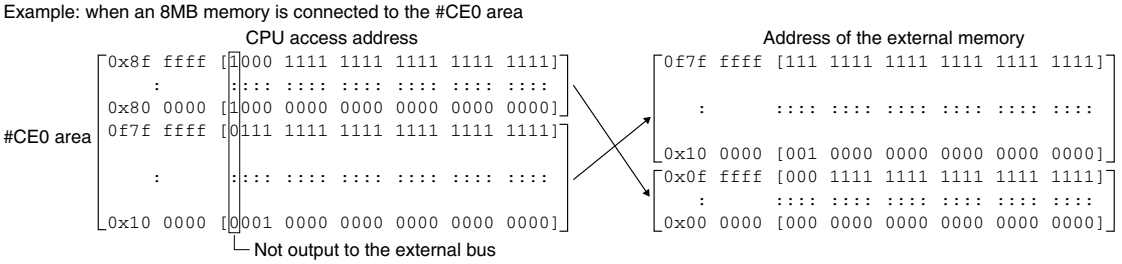


Figure III.2.9.1 CPU Access Address and External Bus Output Address

For example, the address to be accessed will be directly output on the external bus when the CPU accesses an address within the range from 0x100000 to 0x7fffff in the #CE0 area. When the CPU accesses an address within the range from 0x800000 to 0x8fffff (the high-order 1MB of the #CE0 area), the external bus outputs an address within the range from 0x000000 to 0x0fffff (the low-order 1MB of the external memory device will be accessed), as the external address bus does not have A23. In other words, the relative address from beginning of the #CE area may not correspond to the address of the external memory device.

There is no problem when an SRAM is connected to the external bus. Take this into consideration if a ROM in which data has been written externally is connected. Furthermore, when a Flash memory is connected to the external bus, pay attention to the correspondence between the sector to be accessed and the address accessed by the CPU, as the sector size is not the same in all sectors.

IV S1C17501 INTERRUPT SYSTEM

IV.1 Interrupt Controller (ITC)

IV.1.1 Configuration of ITC

The S1C17501 provides 28 interrupt systems listed below.

1. Port input interrupts (8 types)
2. PWM control capture timer/counter interrupt (1 type)
3. A/D converter interrupts (2 types)
4. 16-bit clock generator timer interrupt (1 type)
5. 8-bit clock generator timer interrupts (2 types)
6. UART interrupt (1 type)
7. SPI (SPI CH.0) interrupt (1 type)
8. I^2C interrupt (1 type)
9. RTC interrupt (1 type)
10. 8-bit programmable timer interrupts (4 types)
11. Extended SPI (SPI CH.1) interrupt (1 type)
12. USB function controller interrupt (1 type)
13. I^2S interrupt (2 types)
14. Remote controller interrupt (1 type)

Each interrupt system provides an interrupt flag that indicates the occurrence of an interrupt request from the peripheral module and an interrupt enable bit that enables/disables interrupts. In addition, the ITC allows the application program to set the interrupt level (priority) of each interrupt system that determines the order of handling when two or more interrupts occur at the same time.
( ) in the list above represents the number of interrupt causes supported in each interrupt system. Use the control register in the peripheral module to select the interrupt causes for generating an interrupt request. For more information on interrupt causes and control, see the description for each peripheral module.
Figure IV.1.1.1 shows the structure of the interrupt system.

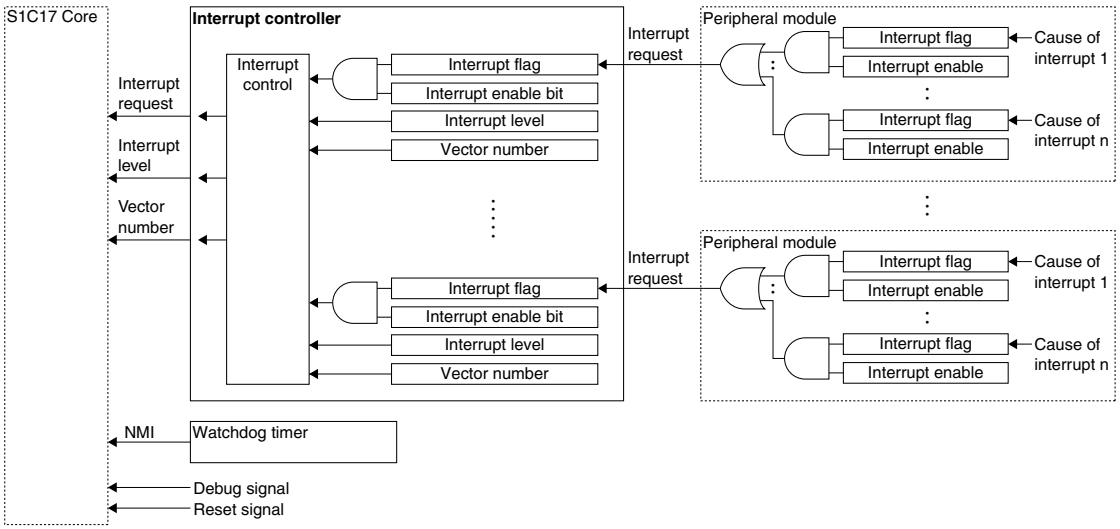


Figure IV.1.1.1 Interrupt System

IV.1.2 Vector Table

The vector table contains the vectors to the interrupt handler routines (handler routine start address) that will be read by the S1C17 Core to execute the handler when an interrupt occurs. The S1C17501 allows the base (starting) address of the vector table to be set using the TTBR_LOW and TTBR_HIGH registers (0x5814, 0x5816). "TTBR" described in Table IV.1.2.1 means the value set to these registers. After an initial reset, the TTBR_LOW/HIGH registers are set to 0x20000. Therefore, even when the vector table position is changed, it is necessary that at least the reset vector be written to the above address. Table IV.1.2.1 shows the vector table of the S1C17501.

Table IV.1.2.1 Vector Table

Vector No.	Vector address	Hardware interrupt name	Cause of hardware interrupt	Priority
0 (0x00)	TTBR + 0x00	Reset	• Low input to the #RESET pin • Watchdog timer overflow *2	1
1 (0x01)	TTBR + 0x04	Address misaligned interrupt	Memory access instruction	2
–	(0xfffc00)	Debugging interrupt	`brk` instruction, etc.	3
2 (0x02)	TTBR + 0x08	NMI	• Low input to the #NMI pin • Watchdog timer overflow *2	4
3 (0x03)	TTBR + 0x0c	C compiler (reserved)	Used in emulation library for C compiler	5
4 (0x04)	TTBR + 0x10	Port input interrupt 0	P*x*0 input (rising/falling edge or high/low level)	High *1
5 (0x05)	TTBR + 0x14	Port input interrupt 1	P*x*1 input (rising/falling edge or high/low level)	↑
6 (0x06)	TTBR + 0x18	Port input interrupt 2	P*x*2 input (rising/falling edge or high/low level)	
7 (0x07)	TTBR + 0x1c	Port input interrupt 3	P*x*3 input (rising/falling edge or high/low level)	
8 (0x08)	TTBR + 0x20	MFT interrupt	• Compare-match • Period-match • ADC protection input • Port protection input	
9 (0x09)	TTBR + 0x24	reserved	–	
10 (0x0a)	TTBR + 0x28	A/D converter	Out of range results (upper- and lower-limit)	
11 (0x0b)	TTBR + 0x2c		End of conversion	
12 (0x0c)	TTBR + 0x30	CLG_T16U0 timer interrupt	Timer underflow	
		Port input interrupt 4	P*x*4 input (rising/falling edge or high/low level)	
13 (0x0d)	TTBR + 0x34	Port input interrupt 5	P*x*5 input (rising/falling edge or high/low level)	
14 (0x0e)	TTBR + 0x38	CLG_T8S timer interrupt	Timer underflow	
		Port input interrupt 6	P*x*6 input (rising/falling edge or high/low level)	
15 (0x0f)	TTBR + 0x3c	CLG_T8I timer interrupt	Timer underflow	
		Port input interrupt 7	P*x*7 input (rising/falling edge or high/low level)	
16 (0x10)	TTBR + 0x40	UART with IrDA CH.0 interrupt	• Transmit buffer empty • Receive buffer full • Receive error	
		Port input interrupt 4	P*x*4 input (rising/falling edge or high/low level)	
17 (0x11)	TTBR + 0x44	Port input interrupt 5	P*x*5 input (rising/falling edge or high/low level)	
18 (0x12)	TTBR + 0x48	SPI CH.0 interrupt	• Transmit buffer empty • Receive buffer full	
		Port input interrupt 6	P*x*6 input (rising/falling edge or high/low level)	
19 (0x13)	TTBR + 0x4c	I^2C interrupt	• Transmit buffer empty • Receive buffer full	
		Port input interrupt 7	P*x*7 input (rising/falling edge or high/low level)	
20 (0x14)	TTBR + 0x50	RTC interrupt	1/64 second, 1 second, 1 minute, or 1 hour count up	
21 (0x15)	TTBR + 0x54	PT8 CH.0 interrupt	Timer 0 underflow	
22 (0x16)	TTBR + 0x58	PT8 CH.1 interrupt	Timer 1 underflow	
23 (0x17)	TTBR + 0x5c	PT8 CH.2 interrupt	Timer 2 underflow	
24 (0x18)	TTBR + 0x60	PT8 CH.3 interrupt	Timer 3 underflow	
25 (0x19)	TTBR + 0x64	reserved	–	
26 (0x1a)	TTBR + 0x68	SPI CH.1 interrupt	• Transmit buffer empty • Receive buffer full	
27 (0x1b)	TTBR + 0x6c	USB function controller interrupt	USB interrupt	
28 (0x1c)	TTBR + 0x70	I^2S interrupt	• I^2S FIFO empty	
29 (0x1d)	TTBR + 0x74		• I^2S FIFO full	
30 (0x1e)	TTBR + 0x78	REMC interrupt	• Envelope counter underflow • REMC_IN rising edge detection • REMC_IN falling edge detection	↓
31 (0x1f)	TTBR + 0x7c	reserved	–	Low *1

*1 When the same interrupt level is set

*2 Either reset or NMI can be selected as the watchdog timer interrupt with software.

Interrupts that share an interrupt vector address

The interrupt vector numbers 12, 14–16, and 18–19 are shared with two causes of interrupts. The interrupt that will occur depends on the setting of the interrupt enable bit (see Section IV.1.3.3). If both the interrupts assigned to one interrupt vector number have been enabled, the interrupt listed in the upper line in the vector table will occur, and the interrupt listed in the lower line in the vector table will not occur.
Set the interrupt enable bits to configure these interrupt systems according to the interrupt to be used as below.

Table IV.1.2.2 Interrupt Vectors 12 and 16 (UART CH.0, CLG_T16U0, and Port 4 Interrupts)

Interrupt enable bit			Interrupt vector 12	Interrupt vector 16
IIEN4 (UART CH.0)	IIEN0 (CLG_T16U0)	EIEN4 (Port 4)		
1	1	1	CLG_T16U0 interrupt	UART CH.0 interrupt
1	1	0	CLG_T16U0 interrupt	UART CH.0 interrupt
1	0	1	Port interrupt 4	UART CH.0 interrupt
1	0	0	–	UART CH.0 interrupt
0	1	1	CLG_T16U0 interrupt	Port interrupt 4
0	1	0	CLG_T16U0 interrupt	–
0	0	1	Port interrupt 4	Port interrupt 4
0	0	0	–	–

(Interrupt enable bit: 1 = enable, 0 = disable)

Table IV.1.2.3 Interrupt Vectors 14 and 18 (SPI CH.0, CLG_T8S, and Port 6 Interrupts)

Interrupt enable bit			Interrupt vector 14	Interrupt vector 18
IIEN6 (SPI CH.0)	IIEN2 (CLG_T8S)	EIEN6 (Port 6)		
1	1	1	CLG_T8S interrupt	SPI CH.0 interrupt
1	1	0	CLG_T8S interrupt	SPI CH.0 interrupt
1	0	1	Port interrupt 6	SPI CH.0 interrupt
1	0	0	–	SPI CH.0 interrupt
0	1	1	CLG_T8S interrupt	Port interrupt 6
0	1	0	CLG_T8S interrupt	–
0	0	1	Port interrupt 6	Port interrupt 6
0	0	0	–	–

Table IV.1.2.4 Interrupt Vectors 15 and 19 (I^2C, CLG_T8I, and Port 7 Interrupts)

Interrupt enable bit			Interrupt vector 15	Interrupt vector 19
IIEN7 (I^2C)	IIEN3 (CLG_T8I)	EIEN7 (Port 7)		
1	1	1	CLG_T8I interrupt	I^2C interrupt
1	1	0	CLG_T8I interrupt	I^2C interrupt
1	0	1	Port interrupt 7	I^2C interrupt
1	0	0	–	I^2C interrupt
0	1	1	CLG_T8I interrupt	Port interrupt 7
0	1	0	CLG_T8I interrupt	–
0	0	1	Port interrupt 7	Port interrupt 7
0	0	0	–	–

IV.1.3 Control of Maskable Interrupts

IV.1.3.1 Enabling ITC

Before the ITC can be used, set the ITEN bit (D0/ITC_CTL register) to 1.

∗ **ITEN**: ITC Enable Bit in the ITC Control (ITC_CTL) Register (D0/0x4304)

IV.1.3.2 Interrupt Request from Peripheral Module and Interrupt Flag

When an enabled interrupt cause occurs in a peripheral module, the module sends an interrupt request signal to the ITC. The interrupt request signal sets the interrupt flag in the ITC corresponding to the cause of interrupt to 1. The interrupt flag holds 1 until it is reset to 0 to indicate that an interrupt request has sent from the peripheral module. The flag status can be read from the ITC_IFLG (0x4300) and ITC_AIFLG (0x42e0) registers.
Table IV.1.3.2.1 lists the relationship between the causes of interrupt and the interrupt flags.

Note: When ITEN (D0/ITC_CTL register) is set to 0, the interrupt flag will not be set even if an interrupt request is generated from the peripheral module.

Table IV.1.3.2.1 Causes of Hardware Interrupt and Interrupt Flags

Cause of hardware interrupt	Interrupt flag
I^2C interrupt: transmit buffer empty/receive buffer full	IIFT7 (D15/ITC_IFLG register)
SPI CH.0 interrupt: transmit buffer empty/receive buffer full	IIFT6 (D14/ITC_IFLG register)
UART interrupt: transmit buffer empty/receive buffer full/receive error	IIFT4 (D12/ITC_IFLG register)
CLG_T8I timer interrupt: timer underflow	IIFT3 (D11/ITC_IFLG register)
CLG_T8S timer interrupt: timer underflow	IIFT2 (D10/ITC_IFLG register)
CLG_T16U0 timer interrupt: timer underflow	IIFT0 (D8/ITC_IFLG register)
Port input interrupt 7: P*x*7 rising/falling edge or high/low level input	EIFT7 (D7/ITC_IFLG register)
Port input interrupt 6: P*x*6 rising/falling edge or high/low level input	EIFT6 (D6/ITC_IFLG register)
Port input interrupt 5: P*x*5 rising/falling edge or high/low level input	EIFT5 (D5/ITC_IFLG register)
Port input interrupt 4: P*x*4 rising/falling edge or high/low level input	EIFT4 (D4/ITC_IFLG register)
Port input interrupt 3: P*x*3 rising/falling edge or high/low level input	EIFT3 (D3/ITC_IFLG register)
Port input interrupt 2: P*x*2 rising/falling edge or high/low level input	EIFT2 (D2/ITC_IFLG register)
Port input interrupt 1: P*x*1 rising/falling edge or high/low level input	EIFT1 (D1/ITC_IFLG register)
Port input interrupt 0: P*x*0 rising/falling edge or high/low level input	EIFT0 (D0/ITC_IFLG register)
Remote controller interrupt: envelope counter underflow/input rising edge/input falling edge	AIFT14 (D14/ITC_AIFLG register)
I^2S interrupt: I^2S FIFO full	AIFT13 (D13/ITC_AIFLG register)
I^2S interrupt: I^2S FIFO empty	AIFT12 (D12/ITC_AIFLG register)
USB interrupt: USB interrupt	AIFT11 (D11/ITC_AIFLG register)
SPI CH.1 interrupt: transmit buffer empty/receive buffer full	AIFT10 (D10/ITC_AIFLG register)
PT8 CH.3 interrupt: timer underflow	AIFT8 (D8/ITC_AIFLG register)
PT8 CH.2 interrupt: timer underflow	AIFT7 (D7/ITC_AIFLG register)
PT8 CH.1 interrupt: timer underflow	AIFT6 (D6/ITC_AIFLG register)
PT8 CH.0 interrupt: timer underflow	AIFT5 (D5/ITC_AIFLG register)
RTC interrupt: 1/64 second, 1 second, 1 minute, or 1 hour count up	AIFT4 (D4/ITC_AIFLG register)
ADC interrupt: end of conversion	AIFT3 (D3/ITC_AIFLG register)
ADC interrupt: out of range	AIFT2 (D2/ITC_AIFLG register)
Multi-function timer interrupt: compare-match/period-match/protection input	AIFT0 (D0/ITC_AIFLG register)

The ITC uses the interrupt flags to generate an interrupt to the S1C17 Core.
When an interrupt flag is set to 1, the ITC sends the interrupt request, interrupt level and vector number signals to the S1C17 Core if the interrupt has been enabled (see the next section).
The interrupt flag that has been set to 1 can be reset by writing 1. Reset the interrupt flag to 0 in the interrupt handler. If the interrupt handler does not reset the interrupt flag, the same interrupt will be generated again when the interrupt handling has finished (interrupts are disabled during interrupt handling and enabled by executing the `reti` instruction placed at the end of the interrupt handler).

Note, however, that the interrupt flags (EIFT0–EIFT7) for the level triggered interrupts (see Section IV.1.3.5) cannot be reset by writing 1. Those interrupt flags are reset when the interrupt signal is negated by the interrupt source.
For the occurrence conditions of the causes of interrupt and the module specific settings, refer to the section that describes the interrupt source module.

IV.1.3.3 Enabling/Disabling Interrupts

To send an interrupt request to the S1C17 Core, the interrupt must be enabled by the interrupt enable bit in the ITC_EN (0x4302) or ITC_AEN (0x42e2) register corresponding to the interrupt flag. To enable an interrupt, set the interrupt enable bit to 1; to disable an interrupt, set the interrupt enable bit to 0 (default). The interrupt enable bit does not affect the interrupt flag status, so the interrupt flag will be set by an interrupt request from the peripheral module regardless of how the interrupt enable bit is set if ITEN (D0/ITC_CTL register) is set to 1.
Table IV.1.3.3.1 lists the correspondence between the interrupt enable bit and the interrupt flag.

Table IV.1.3.3.1 List of Interrupt Enable Bits

Hardware interrupt	Interrupt flag	Interrupt enable bit
I^2C interrupt	IIFT7 (D15/ITC_IFLG register)	IIEN7 (D15/ITC_EN register)
SPI CH.0 interrupt	IIFT6 (D14/ITC_IFLG register)	IIEN6 (D14/ITC_EN register)
UART interrupt	IIFT4 (D12/ITC_IFLG register)	IIEN4 (D12/ITC_EN register)
CLG_T8I timer interrupt	IIFT3 (D11/ITC_IFLG register)	IIEN3 (D11/ITC_EN register)
CLG_T8S timer interrupt	IIFT2 (D10/ITC_IFLG register)	IIEN2 (D10/ITC_EN register)
CLG_T16U0 timer interrupt	IIFT0 (D8/ITC_IFLG register)	IIEN0 (D8/ITC_EN register)
Port input interrupt 7	EIFT7 (D7/ITC_IFLG register)	EIEN7 (D7/ITC_EN register)
Port input interrupt 6	EIFT6 (D6/ITC_IFLG register)	EIEN6 (D6/ITC_EN register)
Port input interrupt 5	EIFT5 (D5/ITC_IFLG register)	EIEN5 (D5/ITC_EN register)
Port input interrupt 4	EIFT4 (D4/ITC_IFLG register)	EIEN4 (D4/ITC_EN register)
Port input interrupt 3	EIFT3 (D3/ITC_IFLG register)	EIEN3 (D3/ITC_EN register)
Port input interrupt 2	EIFT2 (D2/ITC_IFLG register)	EIEN2 (D2/ITC_EN register)
Port input interrupt 1	EIFT1 (D1/ITC_IFLG register)	EIEN1 (D1/ITC_EN register)
Port input interrupt 0	EIFT0 (D0/ITC_IFLG register)	EIEN0 (D0/ITC_EN register)
Remote controller interrupt	AIFT14 (D14/ITC_AIFLG register)	AIEN14 (D14/ITC_AEN register)
I^2S receive interrupt	AIFT13 (D13/ITC_AIFLG register)	AIEN13 (D13/ITC_AEN register)
I^2S transmit interrupt	AIFT12 (D12/ITC_AIFLG register)	AIEN12 (D12/ITC_AEN register)
USB interrupt	AIFT11 (D11/ITC_AIFLG register)	AIEN11 (D11/ITC_AEN register)
SPI CH.1 interrupt	AIFT10 (D10/ITC_AIFLG register)	AIEN10 (D10/ITC_AEN register)
PT8 CH.3 interrupt	AIFT8 (D8/ITC_AIFLG register)	AIEN8 (D8/ITC_AEN register)
PT8 CH.2 interrupt	AIFT7 (D7/ITC_AIFLG register)	AIEN7 (D7/ITC_AEN register)
PT8 CH.1 interrupt	AIFT6 (D6/ITC_AIFLG register)	AIEN6 (D6/ITC_AEN register)
PT8 CH.0 interrupt	AIFT5 (D5/ITC_AIFLG register)	AIEN5 (D5/ITC_AEN register)
RTC interrupt	AIFT4 (D4/ITC_AIFLG register)	AIEN4 (D4/ITC_AEN register)
ADC interrupt (end of conversion)	AIFT3 (D3/ITC_AIFLG register)	AIEN3 (D3/ITC_AEN register)
ADC interrupt (out of range)	AIFT2 (D2/ITC_AIFLG register)	AIEN2 (D2/ITC_AEN register)
Multi-function timer interrupt	AIFT0 (D0/ITC_AIFLG register)	AIEN0 (D0/ITC_AEN register)

Notes: • To avoid unexpected interrupts being generated, always be sure to reset the interrupt flag before enabling the interrupt by writing 1 to the interrupt enable bit.

• In addition to the interrupt enable bit, the IE bit of the Processor Status Register (PSR) in the S1C17 Core must be set to 1 to actually generate an interrupt. If the IE bit has been set to 0, the S1C17 Core cannot accept a maskable interrupt request. In this case, the interrupt request sent from the ITC is held and it will be accepted after the IE bit is set to 1.

IV.1.3.4 Processing when Multiple Interrupts Occur

The ITC provides the ITC_ELV*x* (0x4306 to 0x430c), ITC_ILV*x* (0x430e to 0x4314), ITC_AILV*x* (0x42e6 to 0x42f4) registers to set an interrupt level (zero to seven) for each cause of interrupt.

Table IV.1.3.4.1 Interrupt Level Setup Bits

Hardware interrupt	Interrupt level setup bits	Register address
I^2C interrupt	IILV7[2:0] (D[10:8]/ITC_ILV3 register)	0x4314
SPI CH.0 interrupt	IILV6[2:0] (D[2:0]/ITC_ILV3 register)	0x4314
UART interrupt	IILV4[2:0] (D[2:0]/ITC_ILV2 register)	0x4312
CLG_T8I timer interrupt	IILV3[2:0] (D[10:8]/ITC_ILV1 register)	0x4310
CLG_T8S timer interrupt	IILV2[2:0] (D[2:0]/ITC_ILV1 register)	0x4310
CLG_T16U0 timer interrupt	IILV0[2:0] (D[2:0]/ITC_ILV0 register)	0x430e
Port input interrupt 7	EILV7[2:0] (D[10:8]/ITC_ELV3 register)	0x430c
Port input interrupt 6	EILV6[2:0] (D[2:0]/ITC_ELV3 register)	0x430c
Port input interrupt 5	EILV5[2:0] (D[10:8]/ITC_ELV2 register)	0x430a
Port input interrupt 4	EILV4[2:0] (D[2:0]/ITC_ELV2 register)	0x430a
Port input interrupt 3	EILV3[2:0] (D[10:8]/ITC_ELV1 register)	0x4308
Port input interrupt 2	EILV2[2:0] (D[2:0]/ITC_ELV1 register)	0x4308
Port input interrupt 1	EILV1[2:0] (D[10:8]/ITC_ELV0 register)	0x4306
Port input interrupt 0	EILV0[2:0] (D[2:0]/ITC_ELV0 register)	0x4306
Remote controller interrupt	AILV14[2:0] (D[2:0]/ITC_AILV7 register)	0x42f4
I^2S receive interrupt	AILV13[2:0] (D[10:8]/ITC_AILV6 register)	0x42f2
I^2S transmit interrupt	AILV12[2:0] (D[2:0]/ITC_AILV6 register)	0x42f2
USB interrupt	AILV11[2:0] (D[10:8]/ITC_AILV5 register)	0x42f0
SPI CH.1 interrupt	AILV10[2:0] (D[2:0]/ITC_AILV5 register)	0x42f0
PT8 CH.3 interrupt	AILV8[2:0] (D[2:0]/ITC_AILV4 register)	0x42ee
PT8 CH.2 interrupt	AILV7[2:0] (D[10:8]/ITC_AILV3 register)	0x42ec
PT8 CH.1 interrupt	AILV6[2:0] (D[2:0]/ITC_AILV3 register)	0x42ec
PT8 CH.0 interrupt	AILV5[2:0] (D[10:8]/ITC_AILV2 register)	0x42ea
RTC interrupt	AILV4[2:0] (D[2:0]/ITC_AILV2 register)	0x42ea
ADC interrupt (end of conversion)	AILV3[2:0] (D[10:8]/ITC_AILV1 register)	0x42e8
ADC interrupt (out of range)	AILV2[2:0] (D[2:0]/ITC_AILV1 register)	0x42e8
Multi-function timer interrupt	AILV0[2:0] (D[2:0]/ITC_AILV0 register)	0x42e6

The highest interrupt level is seven and the lowest is zero.
The set interrupt level is sent to the S1C17 Core at the same time the ITC sends an interrupt request and is used by the S1C17 Core to disable subsequent interrupts that have the same or a lower interrupt level. (See Section IV.1.3.6 for more information.)
At initial reset, the interrupt levels are all set to 0. The S1C17 Core does not accept an interrupt request whose interrupt level is set to 0.

In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable bits occur simultaneously, the cause of interrupt whose ITC_ELV*x*, ITC_ILV*x*, or ITC_AILV*x* register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core.
If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first.
Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core.
If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to that of the new cause of interrupt. The first interrupt request is left pending.

IV.1.3.5 Interrupt Trigger Mode

The ITC provides two trigger modes for the port interrupts, the pulse trigger mode and the level trigger mode, to accept either a pulse signal or a level signal as interrupt requests.
The trigger mode can be selected using the EITG*x* bit in the ITC_ELV*x* registers (0x4306 to 0x430c). When EITG*x* is set to 1, level trigger mode is selected; when EITG*x* is set to 0 (default), pulse trigger mode is selected.

The ITC allows these interrupt sources to select the polarity of the interrupt request signal to be sent to the ITC. The signal polarity can be selected using the EITP*x* bit in the ITC_ELV*x* registers (0x4306 to 0x430c). When EITP*x* is set to 1, positive pulse/rising edge (in pulse trigger mode) or active high (in level mode) is selected; when EIPT*x* is set to 0 (default), negative pulse/falling edge (in pulse trigger mode) or active low (in level mode) is selected.

Table IV.1.3.5.1 Trigger Mode/Polarity Select Bits

Hardware interrupt	Trigger mode select bit	Trigger polarity select bit	Register address
Port interrupt 0	EITG0 (D4/ITC_ELV0 register)	EITP0 (D5/ITC_ELV0 register)	0x4306
Port interrupt 1	EITG1 (D12/ITC_ELV0 register)	EITP1 (D13/ITC_ELV0 register)	0x4306
Port interrupt 2	EITG2 (D4/ITC_ELV1 register)	EITP2 (D5/ITC_ELV1 register)	0x4308
Port interrupt 3	EITG3 (D12/ITC_ELV1 register)	EITP3 (D13/ITC_ELV1 register)	0x4308
Port interrupt 4	EITG4 (D4/ITC_ELV2 register)	EITP4 (D5/ITC_ELV2 register)	0x430a
Port interrupt 5	EITG5 (D12/ITC_ELV2 register)	EITP5 (D13/ITC_ELV2 register)	0x430a
Port interrupt 6	EITG6 (D4/ITC_ELV3 register)	EITP6 (D5/ITC_ELV3 register)	0x430c
Port interrupt 7	EITG7 (D12/ITC_ELV3 register)	EITP7 (D13/ITC_ELV3 register)	0x430c

The interrupt source modules other than ports and RTC output only a pulse signal (positive pulse) to the ITC to request an interrupt, therefore, no trigger mode selection bit and trigger polarity selection bit are provided. The RTC interrupt can be configured to either pulse or level trigger mode using a control bit in the RTC module.

Pulse trigger mode

In pulse trigger mode, the ITC samples interrupt signals at the rising edge of the system clock. When the active edge (rising edge or falling edge which can be configured by the trigger polarity bit EITP*x*) of the interrupt signal is sampled, the interrupt detector generates an interrupt pulse. This interrupt pulse sets the interrupt flag EIFT*x* to 1 and trigger the ITC. After the interrupt flag is reset to 0 with software, the interrupt flag is enabled to be set again by the following interrupt pulse.

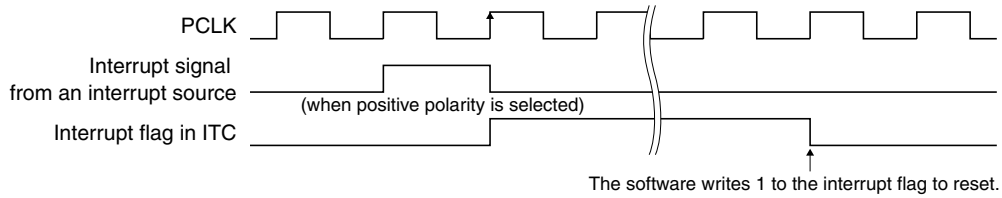


Figure IV.1.3.5.1 Pulse Trigger Mode

Level trigger mode

In level trigger mode, the ITC samples interrupt at any time. When the active level (high level or low level which can be configured by the trigger polarity bit EITP*x*) of the interrupt signal is sampled, the interrupt detector asserts the interrupt signal. This interrupt signal level sets the interrupt flag EIFT*x* to 1 and trigger the ITC. After the interrupt flag is reset to 0 with software, the interrupt flag will be set again immediately and the ITC will be triggered, if the interrupt source holds the interrupt signal at active level.

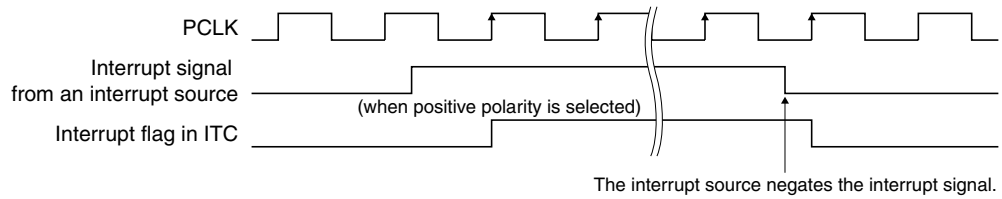


Figure IV.1.3.5.2 Level Trigger Mode

IV.1.3.6 Interrupt Processing by the S1C17 Core

A maskable interrupt to the S1C17 Core occurs when all of the conditions described below are met.

- The ITEN bit (D0/ITC_CTL register) is set to 1.
  - ∗ **ITEN**: ITC Enable Bit in the ITC Control (ITC_CTL) Register (D0/0x4304)
- The interrupt enable bit for the cause of interrupt that has occurred is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The cause of interrupt that has occurred has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

When a cause of interrupt occurs, the corresponding interrupt flag is set to 1 and the flag remains set until it is reset in the software program or by the hardware for a level triggered interrupt. Therefore, in no cases can the generated cause of interrupt be inadvertently cleared even if the above conditions are not met when the cause of interrupt has occurred. The interrupt will occur when the above conditions are met.
If two or more maskable causes of interrupt occur simultaneously, the cause of interrupt that has the highest priority is allowed to signal an interrupt request to the S1C17 Core. The other interrupts with lower priorities are kept pending until the above conditions are met.

The S1C17 Core keeps sampling interrupt requests every cycle. When the S1C17 Core accepts an interrupt request, it enters interrupt processing after completing execution of the instruction that was being executed.
The following lists the contents executed in interrupt processing.

(1) The PSR and the current program counter (PC) value are saved to the stack.
(2) The IE bit of the PSR is reset to 0 (following maskable interrupts are disabled).
(3) The IL of the PSR is set to the interrupt level of the accepted interrupt (NMI does not change the interrupt level).
(4) The vector of the interrupt occurred is loaded into the PC, thus executing the interrupt handler routine.

Thus, once an interrupt is accepted, all maskable interrupts that may follow are disabled in (2). Multiple interrupts can also be handled by setting the IE bit to 1 in the interrupt handler routine. In this case, since the IL has been changed in (3), only an interrupt that has a higher level than that of the currently processed interrupt is accepted.
When the interrupt handler routine is terminated by the `reti` instruction, the PSR is restored to its previous status before the interrupt has occurred. The program restarts processing after branching to the instruction next to the one that was being executed when the interrupt occurred.

IV.1.4 NMI

In the S1C17501, a low level input to the #NMI pin or the watchdog timer generates a non-maskable interrupt (NMI). The vector number of NMI is 2, with the vector address set to the vector table's starting address + 8 bytes. This interrupt is prioritized over other interrupts and is unconditionally accepted by the S1C17 Core.

For how to generate NMI, see Section V.3, “Watchdog Timer (WDT).”

IV.1.5 Software Interrupts

The S1C17 Core provides the `int` *imm5* and `intl` *imm5*,*imm3* instructions allowing the software to generate any interrupts. The operand *imm5* specifies a vector number (0–31) in the vector table. In addition to this, the `intl` instruction has the operand *imm3* to specify the interrupt level (0–7) to be set to the IL field in the PSR.
The processor performs the same interrupt handling as that of the hardware interrupt.

IV.1.6 Clearing Standby Mode by Interrupts

The standby mode (HALT and SLEEP) can be cleared by NMI and normal interrupts.

HALT mode can be cleared by an NMI and a normal interrupt from the modules that are operating in HALT mode with the clock supplied. Note, however, that normal interrupts cannot clear HALT mode when the clock supply to the interrupt source module is stopped in HALT mode or when ITEN (D0/ITC_CTL register) has been set to 0 (interrupt controller is disabled).

∗ **ITEN**: ITC Enable Bit in the ITC Control (ITC_CTL) Register (D0/0x4304)

In SLEEP mode, the system clock is not supply to the interrupt controller (ITC). Therefore, the S1C17 Core can only be released from SLEEP mode by initial reset, #NMI signal and a normal interrupt from the modules which support level trigger mode such as the port and RTC interrupts.

After the S1C17 Core is released from the standby mode, the instruction next to halt or slp will be executed.

Notes:
- Normal interrupts are effective to wake up the S1C17 Core even if the IE bit in PSR is set to 0 (interrupt disabled).
- The interrupts that are set to pulse trigger mode cannot be used to clear SLEEP mode.
- When a cause of interrupt is used to clear HALT or SLEEP mode, the interrupt enable bit corresponding to the cause of interrupt must be set to 1 (interrupt enabled).

IV.1.7 Details of Control Registers

Table IV.1.7.1 List of ITC Registers

Address	Register name		Function
0x42e0	ITC_AIFLG	Additional Interrupt Flag Register	Indicates/resets interrupt occurrence status.
0x42e2	ITC_AEN	Additional Interrupt Enable Register	Enables/disables each maskable interrupt.
0x42e6	ITC_AILV0	Additional Interrupt Level Setup Register 0	Sets the MFT interrupt level.
0x42e8	ITC_AILV1	Additional Interrupt Level Setup Register 1	Sets the ADC interrupt level.
0x42ea	ITC_AILV2	Additional Interrupt Level Setup Register 2	Sets the RTC and PT8 CH.0 interrupt levels.
0x42ec	ITC_AILV3	Additional Interrupt Level Setup Register 3	Sets the PT8 CH.1 and CH.2 interrupt levels.
0x42ee	ITC_AILV4	Additional Interrupt Level Setup Register 4	Sets the PT8 CH.3
0x42f0	ITC_AILV5	Additional Interrupt Level Setup Register 5	Sets the SPI CH.1 and USB interrupt levels.
0x42f2	ITC_AILV6	Additional Interrupt Level Setup Register 6	Sets the I^2S interrupt level.
0x42f4	ITC_AILV7	Additional Interrupt Level Setup Register 7	Sets the REMC interrupt level.
0x4300	ITC_IFLG	Interrupt Flag Register	Indicates/resets interrupt occurrence status.
0x4302	ITC_EN	Interrupt Enable Register	Enables/disables each maskable interrupt.
0x4304	ITC_CTL	ITC Control Register	Enables/disables the ITC.
0x4306	ITC_ELV0	External Interrupt Level Setup Register 0	Sets the port 0 and port 1 interrupt levels and trigger modes.
0x4308	ITC_ELV1	External Interrupt Level Setup Register 1	Sets the port 2 and port 3 interrupt levels and trigger modes.
0x430a	ITC_ELV2	External Interrupt Level Setup Register 2	Sets the port 4 and port 5 interrupt levels and trigger modes.
0x430c	ITC_ELV3	External Interrupt Level Setup Register 3	Sets the port 6 and port 7 interrupt levels and trigger modes.
0x430e	ITC_ILV0	Internal Interrupt Level Setup Register 0	Sets the CLG_T16U0 timer interrupt level.
0x4310	ITC_ILV1	Internal Interrupt Level Setup Register 1	Sets the CLG_T8S and CLG_T8I timer interrupt levels.
0x4312	ITC_ILV2	Internal Interrupt Level Setup Register 2	Sets the UART interrupt level.
0x4314	ITC_ILV3	Internal Interrupt Level Setup Register 3	Sets the SPI CH.0 and I^2C interrupt levels.

The following describes each ITC register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x42e0: Additional Interrupt Flag Register (ITC_AIFLG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Flag Register (ITC_AIFLG)	0x42e0 (16 bits)	D15	–	reserved	–	–	–	0 when being read.
		D14	**AIFT14**	REMC interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D13	**AIFT13**	I^2S receive interrupt flag		0	R/W	
		D12	**AIFT12**	I^2S transmit interrupt flag		0	R/W	
		D11	**AIFT11**	USB interrupt flag		0	R/W	
		D10	**AIFT10**	SPI CH.1 interrupt flag		0	R/W	
		D9	–	reserved	–	–	–	0 when being read.
		D8	**AIFT8**	PT8 CH.3 interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D7	**AIFT7**	PT8 CH.2 interrupt flag		0	R/W	
		D6	**AIFT6**	PT8 CH.1 interrupt flag		0	R/W	
		D5	**AIFT5**	PT8 CH.0 interrupt flag		0	R/W	
		D4	**AIFT4**	RTC interrupt flag		0	R/W	
		D3	**AIFT3**	ADC end-of-conversion interrupt flag		0	R/W	
		D2	**AIFT2**	ADC out-of-range interrupt flag		0	R/W	
		D1	–	reserved	–	–	–	0 when being read.
		D0	**AIFT0**	MFT interrupt flag	1 Occurred / 0 Not occurred	0	R/W	Reset by writing 1.

D15, D9, D1 Reserved

D[14:10], D[8:2], D0 AIFT[14:10], AIFT[8:2], AIFT0: Interrupt Flag Bits

These bits are interrupt flags to indicate the interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Has no effect

The interrupt flag is set to 1 if a cause of interrupt occurs in each peripheral module when ITEN (D0/ ITC_CTL register) in the ITC is set to 1.
If the following conditions are met at this time, an interrupt is generated to the S1C17 Core:

1. The corresponding bit of the Interrupt Enable Register is set to 1.
2. No other interrupt request of higher priority has occurred.
3. The IE bit of the PSR is set to 1 (interrupt enabled).
4. The corresponding interrupt level setup bits are set to a level higher than the S1C17 Core's interrupt level (IL).

The interrupt flag is always set to 1 when a cause of interrupt occurs regardless of how the interrupt enable and interrupt level setup bits are set.
In order for the next interrupt to be accepted after interrupt generation, the interrupt flag must be reset and the PSR must be set up again (by setting the IE bit to 1 or executing the `reti` instruction).
The flag that has been set to 1 can be reset by writing 1.

Table IV.1.7.2 Causes of Hardware Interrupt and Interrupt Flags

Interrupt flag	Cause of hardware interrupt
AIFT14 (D14)	REMC interrupt: envelope counter underflow/input rising edge/input falling edge
AIFT13 (D13)	I^2S interrupt: I^2S FIFO full
AIFT12 (D12)	I^2S interrupt: I^2S FIFO empty
AIFT11 (D11)	USB interrupt: USB interrupt
AIFT10 (D10)	SPI CH.1 interrupt: transmit buffer empty/receive buffer full
–	reserved
AIFT8 (D8)	PT8 CH.3 interrupt: timer underflow
AIFT7 (D7)	PT8 CH.2 interrupt: timer underflow
AIFT6 (D6)	PT8 CH.1 interrupt: timer underflow
AIFT5 (D5)	PT8 CH.0 interrupt: timer underflow
AIFT4 (D4)	RTC interrupt: 1/64 second, 1 second, 1 minute, or 1 hour count up
AIFT3 (D3)	ADC interrupt: end of conversion
AIFT2 (D2)	ADC interrupt: out of range
AIFT0 (D0)	Multi-function timer interrupt: compare-match/period-match/protection input

0x42e2: Additional Interrupt Enable Register (ITC_AEN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Enable Register (ITC_AEN)	0x42e2 (16 bits)	D15	–	reserved	–	–	–	0 when being read.
		D14	AIEN14	REMC interrupt enable	1 Enable 0 Disable	0	R/W	
		D13	AIEN13	I^2S receive interrupt enable		0	R/W	
		D12	AIEN12	I^2S transmit interrupt enable		0	R/W	
		D11	AIEN11	USB interrupt enable		0	R/W	
		D10	AIEN10	SPI CH.1 interrupt enable		0	R/W	
		D9	–	reserved	–	–	–	0 when being read.
		D8	AIEN8	PT8 CH.3 interrupt enable	1 Enable 0 Disable	0	R/W	
		D7	AIEN7	PT8 CH.2 interrupt enable		0	R/W	
		D6	AIEN6	PT8 CH.1 interrupt enable		0	R/W	
		D5	AIEN5	PT8 CH.0 interrupt enable		0	R/W	
		D4	AIEN4	RTC interrupt enable		0	R/W	
		D3	AIEN3	ADC end-of-conversion interrupt enable		0	R/W	
		D2	AIEN2	ADC out-of-range interrupt enable		0	R/W	
		D1	–	reserved	–	–	–	0 when being read.
		D0	AIEN0	MFT interrupt enable	1 Enable 0 Disable	0	R/W	

D15, D9, D1 Reserved

D[14:10], D[8:2], D0 AIEN[14:10], AIEN[8:2], AIEN0: Interrupt Enable Bits

These bits enable or disable interrupt generation.

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

Interrupts are enabled when the corresponding interrupt enable bit is set to 1 and are disabled when the bit is set to 0.

When using a cause of interrupt to clear standby mode, the corresponding interrupt enable bit must be set to 1.

Table IV.1.7.3 Causes of Hardware Interrupt and Interrupt Enable Bits

Interrupt enable bits	Cause of hardware interrupt
AIEN14 (D14)	REMC interrupt: envelope counter underflow/input rising edge/input falling edge
AIEN13 (D13)	I^2S interrupt: I^2S FIFO full
AIEN12 (D12)	I^2S interrupt: I^2S FIFO empty
AIEN11 (D11)	USB interrupt: USB interrupt
AIEN10 (D10)	SPI CH.1 interrupt: transmit buffer empty/receive buffer full
–	reserved
AIEN8 (D8)	PT8 CH.3 interrupt: timer underflow
AIEN7 (D7)	PT8 CH.2 interrupt: timer underflow
AIEN6 (D6)	PT8 CH.1 interrupt: timer underflow
AIEN5 (D5)	PT8 CH.0 interrupt: timer underflow
AIEN4 (D4)	RTC interrupt: 1/64 second, 1 second, 1 minute, or 1 hour count up
AIEN3 (D3)	ADC interrupt: end of conversion
AIEN2 (D2)	ADC interrupt: out of range
AIEN0 (D0)	Multi-function timer interrupt: compare-match/period-match/protection input

0x42e6: Additional Interrupt Level Setup Register 0 (ITC_AILV0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 0 (ITC_AILV0)	**0x42e6** (16 bits)	D15–3	–	reserved	–	–	–	0 when being read.
		D2–0	**AILV0[2:0]**	MFT interrupt level	0 to 7	0x0	R/W	

D[15:3] Reserved

D[2:0] AILV0[2:0]: MFT Interrupt Level Bits

Sets the interrupt level (0 to 7) of the multi-function timer interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

0x42e8: Additional Interrupt Level Setup Register 1 (ITC_AILV1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 1 (ITC_AILV1)	**0x42e8** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV3[2:0]**	ADC end-of-conversion interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV2[2:0]**	ADC out-of-range interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] AILV3[2:0]: ADC End-of-Conversion Interrupt Level Bits

Sets the interrupt level (0 to 7) of the end-of-conversion interrupt of the A/D converter. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

D[7:3] Reserved

D[2:0] AILV2[2:0]: ADC Out-of-Range Interrupt Level Bits

Sets the interrupt level (0 to 7) of the out-of-range interrupt of the A/D converter. (Default: 0x0)
See the description of AILV3[2:0] (D[10:8]).

0x42ea: Additional Interrupt Level Setup Register 2 (ITC_AILV2)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 2 (ITC_AILV2)	0x42ea (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10–8	AILV5[2:0]	PT8 CH.0 interrupt level	0 to 7	0x0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	AILV4[2:0]	RTC interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] AILV5[2:0]: PT8 CH.0 Interrupt Level Bits

Sets the interrupt level (0 to 7) of the 8-bit programmable timer CH.0 interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

D[7:3] Reserved

D[2:0] AILV4[2:0]: RTC Interrupt Level Bits

Sets the interrupt level (0 to 7) of the RTC interrupt. (Default: 0x0)
See the description of AILV5[2:0] (D[10:8]).

0x42ec: Additional Interrupt Level Setup Register 3 (ITC_AILV3)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 3 (ITC_AILV3)	0x42ec (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10–8	AILV7[2:0]	PT8 CH.2 interrupt level	0 to 7	0x0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	AILV6[2:0]	PT8 CH.1 interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] AILV7[2:0]: PT8 CH.2 Interrupt Level Bits

Sets the interrupt level (0 to 7) of the 8-bit programmable timer CH.2 interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

D[7:3] Reserved

D[2:0] AILV6[2:0]: PT8 CH.1 Interrupt Level Bits

Sets the interrupt level (0 to 7) of the 8-bit programmable timer CH.1 interrupt. (Default: 0x0)
See the description of AILV7[2:0] (D[10:8]).

0x42ee: Additional Interrupt Level Setup Register 4 (ITC_AILV4)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 4 (ITC_AILV4)	**0x42ee** (16 bits)	D15–3	–	reserved	–	–	–	0 when being read.
		D2–0	**AILV8[2:0]**	PT8 CH.3 interrupt level	0 to 7	0x0	R/W	

D[15:3] Reserved

D[2:0] AILV8[2:0]: PT8 CH.3 Interrupt Level Bits

Sets the interrupt level (0 to 7) of the 8-bit programmable timer CH.3 interrupt. (Default: 0x0)
See the description of AILV9[2:0] (D[10:8]).

0x42f0: Additional Interrupt Level Setup Register 5 (ITC_AILV5)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 5 (ITC_AILV5)	**0x42f0** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV11[2:0]**	USB interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV10[2:0]**	SPI CH.1 interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] AILV11[2:0]: USB Interrupt Level Bits

Sets the interrupt level (0 to 7) of the USB interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

D[7:3] Reserved

D[2:0] AILV10[2:0]: SPI CH.1 Interrupt Level Bits

Sets the interrupt level (0 to 7) of the SPI CH.1 interrupt. (Default: 0x0)
See the description of AILV11[2:0] (D[10:8]).

0x42f2: Additional Interrupt Level Setup Register 6 (ITC_AILV6)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 6 (ITC_AILV6)	0x42f2 (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10–8	AILV13[2:0]	I^2S receive interrupt level	0 to 7	0x0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	AILV12[2:0]	I^2S transmit interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] AILV13[2:0]: I^2S Receive Interrupt Level Bits

Sets the interrupt level (0 to 7) of the I^2S receive interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

D[7:3] Reserved

D[2:0] AILV12[2:0]: I^2S Transmit Interrupt Level Bits

Sets the interrupt level (0 to 7) of the I^2S transmit interrupt. (Default: 0x0)
See the description of AILV13[2:0] (D[10:8]).

0x42f4: Additional Interrupt Level Setup Register 7 (ITC_AILV7)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Level Setup Register 7 (ITC_AILV7)	**0x42f4** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV14[2:0]**	REMC interrupt level	0 to 7	0x0	R/W	

D[15:3] Reserved

D[2:0] AILV14[2:0]: REMC Interrupt Level Bits

Sets the interrupt level (0 to 7) of the remote controller interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

0x4300: Interrupt Flag Register (ITC_IFLG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Interrupt Flag Register (ITC_IFLG)	0x4300 (16 bits)	D15	**IIFT7**	I^2C interrupt flag	1	Cause of interrupt occurred	0	Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D14	**IIFT6**	SPI CH.0 interrupt flag					0	R/W	
		D13	–	reserved	–				–	–	0 when being read.
		D12	**IIFT4**	UART interrupt flag	1	Cause of interrupt occurred	0	Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D11	**IIFT3**	CLG_T8I timer interrupt flag					0	R/W	
		D10	**IIFT2**	CLG_T8S timer interrupt flag					0	R/W	
		D9	–	reserved	–				–	–	0 when being read.
		D8	**IIFT0**	CLG_T16U0 timer interrupt flag	1	Cause of interrupt occurred	0	Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D7	**EIFT7**	Port interrupt 7 flag					0	R/W	Reset by writing 1 in pulse trigger mode. Cannot be reset by software in level trigger mode.
		D6	**EIFT6**	Port interrupt 6 flag					0	R/W	
		D5	**EIFT5**	Port interrupt 5 flag					0	R/W	
		D4	**EIFT4**	Port interrupt 4 flag					0	R/W	
		D3	**EIFT3**	Port interrupt 3 flag					0	R/W	
		D2	**EIFT2**	Port interrupt 2 flag					0	R/W	
		D1	**EIFT1**	Port interrupt 1 flag					0	R/W	
		D0	**EIFT0**	Port interrupt 0 flag					0	R/W	

D13, D9 Reserved

D[15:14], D[12:10], D8 IIFT[7:6], IIFT[4:2], IIFT0: Interrupt Flag Bits

These bits are interrupt flags to indicate the interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Has no effect

The interrupt flag is set to 1 if a cause of interrupt occurs in each peripheral module when ITEN (D0/ ITC_CTL register) in the ITC is set to 1.
If the following conditions are met at this time, an interrupt is generated to the S1C17 Core:

1. The corresponding bit of the Interrupt Enable Register is set to 1.
2. No other interrupt request of higher priority has occurred.
3. The IE bit of the PSR is set to 1 (interrupt enabled).
4. The corresponding interrupt level setup bits are set to a level higher than the S1C17 Core's interrupt level (IL).

The interrupt flag is always set to 1 when a cause of interrupt occurs regardless of how the interrupt enable and interrupt level setup bits are set.
In order for the next interrupt to be accepted after interrupt generation, the interrupt flag must be reset and the PSR must be set up again (by setting the IE bit to 1 or executing the `reti` instruction).
The flag that has been set to 1 can be reset by writing 1.

Table IV.1.7.4 Causes of Hardware Interrupt and Interrupt Flags

Interrupt flag	Cause of hardware interrupt
IIFT7 (D15)	I^2C interrupt: transmit buffer empty/receive buffer full
IIFT6 (D14)	SPI CH.0 interrupt: transmit buffer empty/receive buffer full
IIFT4 (D12)	UART interrupt: transmit buffer empty/receive buffer full/receive error
IIFT3 (D11)	CLG_T8I timer interrupt: timer underflow
IIFT2 (D10)	CLG_T8S timer interrupt: timer underflow
IIFT0 (D8)	CLG_T16U0 timer interrupt: timer underflow

D[7:0] EIFT[7:0]: Interrupt Flag Bits

These bits are interrupt flags to indicate the interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Has no effect
0 (W): Has no effect

See the description for IIFT[7:0].
However, these interrupts allows selection of interrupt trigger conditions using the ITC_ELV*x* register (0x4306 to 0x430c).

Table IV.1.7.5 Causes of Hardware Interrupt and Interrupt Flags

Interrupt flag	Cause of hardware interrupt
EIFT7 (D7)	Port input interrupt 7: P*x*7 rising/falling edge or high/low level input
EIFT6 (D6)	Port input interrupt 6: P*x*6 rising/falling edge or high/low level input
EIFT5 (D5)	Port input interrupt 5: P*x*5 rising/falling edge or high/low level input
EIFT4 (D4)	Port input interrupt 4: P*x*4 rising/falling edge or high/low level input
EIFT3 (D3)	Port input interrupt 3: P*x*3 rising/falling edge or high/low level input
EIFT2 (D2)	Port input interrupt 2: P*x*2 rising/falling edge or high/low level input
EIFT1 (D1)	Port input interrupt 1: P*x*1 rising/falling edge or high/low level input
EIFT0 (D0)	Port input interrupt 0: P*x*0 rising/falling edge or high/low level input

0x4302: Interrupt Enable Register (ITC_EN)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Interrupt Enable Register (ITC_EN)	0x4302 (16 bits)	D15	IIEN7	I^2C interrupt enable	1	Enable	0	Disable	0	R/W	
		D14	IIEN6	SPI CH.0 interrupt enable					0	R/W	
		D13	–	reserved	–				–	–	0 when being read.
		D12	IIEN4	UART interrupt enable	1	Enable	0	Disable	0	R/W	
		D11	IIEN3	CLG_T8I timer interrupt enable					0	R/W	
		D10	IIEN2	CLG_T8S timer interrupt enable					0	R/W	
		D9	–	reserved	–				–	–	0 when being read.
		D8	IIEN0	CLG_T16U0 timer interrupt enable	1	Enable	0	Disable	0	R/W	
		D7	EIEN7	Port interrupt 7 enable					0	R/W	
		D6	EIEN6	Port interrupt 6 enable					0	R/W	
		D5	EIEN5	Port interrupt 5 enable					0	R/W	
		D4	EIEN4	Port interrupt 4 enable					0	R/W	
		D3	EIEN3	Port interrupt 3 enable					0	R/W	
		D2	EIEN2	Port interrupt 2 enable					0	R/W	
		D1	EIEN1	Port interrupt 1 enable					0	R/W	
		D0	EIEN0	Port interrupt 0 enable					0	R/W	

D13, D9 Reserved

D[15:14], D[12:10], D8, D[7:0] IIEN[7:6], IIEN[4:2], IIEN0, EIEN[7:0]: Interrupt Enable Bits

These bits enable or disable interrupt generation.

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

Interrupts are enabled when the corresponding interrupt enable bit is set to 1 and are disabled when the bit is set to 0.

When using a cause of interrupt to clear standby mode, the corresponding interrupt enable bit must be set to 1.

Table IV.1.7.6 Causes of Hardware Interrupt and Interrupt Enable Bits

Interrupt enable bits	Cause of hardware interrupt
IIEN7 (D15)	I^2C interrupt: transmit buffer empty/receive buffer full
IIEN6 (D14)	SPI CH.0 interrupt: transmit buffer empty/receive buffer full
IIEN4 (D12)	UART interrupt: transmit buffer empty/receive buffer full/receive error
IIEN3 (D11)	CLG_T8I timer interrupt: timer underflow
IIEN2 (D10)	CLG_T8S timer interrupt: timer underflow
IIEN0 (D8)	CLG_T16U0 timer interrupt: timer underflow
EIEN7 (D7)	Port input interrupt 7: P*x*7 rising/falling edge or high/low level input
EIEN6 (D6)	Port input interrupt 6: P*x*6 rising/falling edge or high/low level input
EIEN5 (D5)	Port input interrupt 5: P*x*5 rising/falling edge or high/low level input
EIEN4 (D4)	Port input interrupt 4: P*x*4 rising/falling edge or high/low level input
EIEN3 (D3)	Port input interrupt 3: P*x*3 rising/falling edge or high/low level input
EIEN2 (D2)	Port input interrupt 2: P*x*2 rising/falling edge or high/low level input
EIEN1 (D1)	Port input interrupt 1: P*x*1 rising/falling edge or high/low level input
EIEN0 (D0)	Port input interrupt 0: P*x*0 rising/falling edge or high/low level input

0x4304: ITC Control Register (ITC_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
ITC Control Register (ITC_CTL)	0x4304 (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	ITEN	ITC enable	1 Enable 0 Disable	0	R/W	

D[15:1] Reserved

D0 ITEN: ITC Enable Bit

Enables the ITC to control interrupt generation.
1 (R/W): Enable
0 (R/W): Disable (default)

Before the ITC can be used, this bit must be set to 1.

0x4306: External Interrupt Level Setup Register 0 (ITC_ELV0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Interrupt Level Setup Register 0 (ITC_ELV0)	0x4306 (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13	EITP1	Port interrupt 1 trigger polarity	1 Positive 0 Negative	0	R/W	
		D12	EITG1	Port interrupt 1 trigger mode	1 Level 0 Pulse	0	R/W	
		D11	–	reserved	–	–	–	0 when being read.
		D10–8	EILV1[2:0]	Port interrupt 1 level	0 to 7	0x0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5	EITP0	Port interrupt 0 trigger polarity	1 Positive 0 Negative	0	R/W	
		D4	EITG0	Port interrupt 0 trigger mode	1 Level 0 Pulse	0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	EILV0[2:0]	Port interrupt 0 level	0 to 7	0x0	R/W	

D[15:14] Reserved

D13 EITP1: Port Interrupt 1 Trigger Polarity Bit

Selects the polarity of the port interrupt 1 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

In pulse trigger mode, the port outputs a positive pulse for an interrupt request to the ITC when this bit is set to 1 or a negative pulse when this bit is set to 0.
In level trigger mode, the port outputs an active high signal for an interrupt request to the ITC when this bit is set to 1 or a active low signal when this bit is set to 0.

D12 EITG1: Port Interrupt 1 Trigger Mode Bit

Selects the trigger mode of the port interrupt 1.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

In pulse trigger mode, the ITC samples interrupt signals at the rising edge of the system clock. When a pulse with the specified polarity is sampled, the ITC sets the interrupt flag (EIFT*x*) to 1 and stops sampling of that interrupt signal. The ITC resumes the sampling operation for the interrupt signal after the interrupt flag (EIFT*x*) is reset to 0 in the application program (interrupt handler).
In level trigger mode, the ITC continuously samples interrupt signals at every rising edge of the system clock. The interrupt flag (EIFT*x*) is set to 1 when the specified active level is sampled and is reset to 0 when the inactive level is sampled. In this mode, writing 1 cannot reset the interrupt flag (EIFT*x*). Therefore, the interrupt source module must hold the interrupt signal to high until the S1C17 Core accepts the interrupt request and must reset the interrupt signal after that.

D11 Reserved

D[10:8] EILV1[2:0]: Port Interrupt 1 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 1. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

D[7:6] Reserved

D5 EITP0: Port Interrupt 0 Trigger Polarity Bit

Selects the polarity of the port interrupt 0 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13).

D4 EITG0: Port Interrupt 0 Trigger Mode Bit

Selects the trigger mode of the port interrupt 0.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12).

D3 Reserved

D[2:0] EILV0[2:0]: Port Interrupt 0 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 0. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]).

0x4308: External Interrupt Level Setup Register 1 (ITC_ELV1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Interrupt Level Setup Register 1 (ITC_ELV1)	0x4308 (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13	EITP3	Port interrupt 3 trigger polarity	1 Positive 0 Negative	0	R/W	
		D12	EITG3	Port interrupt 3 trigger mode	1 Level 0 Pulse	0	R/W	
		D11	–	reserved	–	–	–	0 when being read.
		D10–8	EILV3[2:0]	Port interrupt 3 level	0 to 7	0x0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5	EITP2	Port interrupt 2 trigger polarity	1 Positive 0 Negative	0	R/W	
		D4	EITG2	Port interrupt 2 trigger mode	1 Level 0 Pulse	0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	EILV2[2:0]	Port interrupt 2 level	0 to 7	0x0	R/W	

D[15:14] Reserved

D13 EITP3: Port Interrupt 3 Trigger Polarity Bit

Selects the polarity of the port interrupt 3 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13) in the ITC_ELV0 register (0x4306).

D12 EITG3: Port Interrupt 3 Trigger Mode Bit

Selects the trigger mode of the port interrupt 3.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12) in the ITC_ELV0 register (0x4306).

D11 Reserved

D[10:8] EILV3[2:0]: Port Interrupt 3 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 3. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]) in the ITC_ELV0 register (0x4306).

D[7:6] Reserved

D5 EITP2: Port Interrupt 2 Trigger Polarity Bit

Selects the polarity of the port interrupt 2 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13) in the ITC_ELV0 register (0x4306).

D4 EITG2: Port Interrupt 2 Trigger Mode Bit

Selects the trigger mode of the port interrupt 2.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12) in the ITC_ELV0 register (0x4306).

D3 Reserved

D[2:0] EILV2[2:0]: Port Interrupt 2 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 2. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]) in the ITC_ELV0 register (0x4306).

0x430a: External Interrupt Level Setup Register 2 (ITC_ELV2)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Interrupt Level Setup Register 2 (ITC_ELV2)	0x430a (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13	EITP5	Port interrupt 5 trigger polarity	1 Positive 0 Negative	0	R/W	
		D12	EITG5	Port interrupt 5 trigger mode	1 Level 0 Pulse	0	R/W	
		D11	–	reserved	–	–	–	0 when being read.
		D10–8	EILV5[2:0]	Port interrupt 5 level	0 to 7	0x0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5	EITP4	Port interrupt 4 trigger polarity	1 Positive 0 Negative	0	R/W	
		D4	EITG4	Port interrupt 4 trigger mode	1 Level 0 Pulse	0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	EILV4[2:0]	Port interrupt 4 level	0 to 7	0x0	R/W	

D[15:14] Reserved

D13 EITP5: Port Interrupt 5 Trigger Polarity Bit

Selects the polarity of the port interrupt 5 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13) in the ITC_ELV0 register (0x4306).

D12 EITG5: Port Interrupt 5 Trigger Mode Bit

Selects the trigger mode of the port interrupt 5.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12) in the ITC_ELV0 register (0x4306).

D11 Reserved

D[10:8] EILV5[2:0]: Port Interrupt 5 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 5. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]) in the ITC_ELV0 register (0x4306).

D[7:6] Reserved

D5 EITP4: Port Interrupt 4 Trigger Polarity Bit

Selects the polarity of the port interrupt 4 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13) in the ITC_ELV0 register (0x4306).

D4 EITG4: Port Interrupt 4 Trigger Mode Bit

Selects the trigger mode of the port interrupt 4.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12) in the ITC_ELV0 register (0x4306).

D3 Reserved

D[2:0] EILV4[2:0]: Port Interrupt 4 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 4. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]) in the ITC_ELV0 register (0x4306).

0x430c: External Interrupt Level Setup Register 3 (ITC_ELV3)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Interrupt Level Setup Register 3 (ITC_ELV3)	0x430c (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13	EITP7	Port interrupt 7 trigger polarity	1 Positive 0 Negative	0	R/W	
		D12	EITG7	Port interrupt 7 trigger mode	1 Level 0 Pulse	0	R/W	
		D11	–	reserved	–	–	–	0 when being read.
		D10–8	EILV7[2:0]	Port interrupt 7 level	0 to 7	0x0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5	EITP6	Port interrupt 6 trigger polarity	1 Positive 0 Negative	0	R/W	
		D4	EITG6	Port interrupt 6 trigger mode	1 Level 0 Pulse	0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	EILV6[2:0]	Port interrupt 6 level	0 to 7	0x0	R/W	

D[15:14] Reserved

D13 EITP7: Port Interrupt 7 Trigger Polarity Bit

Selects the polarity of the port interrupt 7 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13) in the ITC_ELV0 register (0x4306).

D12 EITG7: Port Interrupt 7 Trigger Mode Bit

Selects the trigger mode of the port interrupt 7.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12) in the ITC_ELV0 register (0x4306).

D11 Reserved

D[10:8] EILV7[2:0]: Port Interrupt 7 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 7. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]) in the ITC_ELV0 register (0x4306).

D[7:6] Reserved

D5 EITP6: Port Interrupt 6 Trigger Polarity Bit

Selects the polarity of the port interrupt 6 signal.
1 (R/W): Positive/active high
0 (R/W): Negative/active low (default)

See the description of EITP1 (D13) in the ITC_ELV0 register (0x4306).

D4 EITG6: Port Interrupt 6 Trigger Mode Bit

Selects the trigger mode of the port interrupt 6.
1 (R/W): Level trigger mode
0 (R/W): Pulse trigger mode (default)

See the description of EITG1 (D12) in the ITC_ELV0 register (0x4306).

D3 Reserved

D[2:0] EILV6[2:0]: Port Interrupt 6 Level Bits

Sets the interrupt level (0 to 7) of the port interrupt 6. (Default: 0x0)
See the description of EILV1[2:0] (D[10:8]) in the ITC_ELV0 register (0x4306).

0x430e: Internal Interrupt Level Setup Register 0 (ITC_ILV0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Internal Interrupt Level Setup Register 0 (ITC_ILV0)	**0x430e** (16 bits)	D15–3	–	reserved	–	–	–	0 when being read.
		D2–0	**IILV0[2:0]**	CLG_T16U0 timer interrupt level	0 to 7	0x0	R/W	

D[15:3] Reserved

D[2:0] IILV0[2:0]: CLG_T16U0 Timer Interrupt Level Bits

Sets the interrupt level (0 to 7) of the CLG_T16U0 timer interrupt. (Default: 0x0)
If the level is set below the IL value of the PSR, the S1C17 Core does not accept the interrupt request.
In the ITC, the interrupt level is used when two or more causes of interrupt occur simultaneously.
If two or more causes of interrupt that have been enabled by the interrupt enable register occur simultaneously, the cause of interrupt whose Interrupt Level Setup Register contains the highest value is allowed by the ITC to send an interrupt request to the S1C17 Core. If two or more causes of interrupt that have the same interrupt level occur, the interrupt with the smallest vector number is processed first. Other causes of interrupt are kept pending until all interrupts of higher priority are accepted by the S1C17 Core. If another cause of interrupt of higher priority occurs during outputting an interrupt request signal, the ITC changes the vector number and interrupt level to those of the new cause of interrupt. The first interrupt request is left pending.

0x4310: Internal Interrupt Level Setup Register 1 (ITC_ILV1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Internal Interrupt Level Setup Register 1 (ITC_ILV1)	**0x4310** (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10–8	**IILV3[2:0]**	CLG_T8I timer interrupt level	0 to 7	0x0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**IILV2[2:0]**	CLG_T8S timer interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] IILV3[2:0]: CLG_T8I Timer Interrupt Level Bits

Sets the interrupt level (0 to 7) of the CLG_T8I timer interrupt. (Default: 0x0)
See the description of IILV0[2:0] (D[2:0]) in the ITC_ILV0 register (0x430e).

D[7:3] Reserved

D[2:0] IILV2[2:0]: CLG_T8S Timer Interrupt Level Bits

Sets the interrupt level (0 to 7) of the CLG_T8S timer interrupt. (Default: 0x0)
See the description of IILV0[2:0] (D[2:0]) in the ITC_ILV0 register (0x430e).

0x4312: Internal Interrupt Level Setup Register 2 (ITC_ILV2)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Internal Interrupt Level Setup Register 2 (ITC_ILV2)	**0x4312** (16 bits)	D15–3	–	reserved	–	–	–	0 when being read.
		D2–0	**IILV4[2:0]**	UART interrupt level	0 to 7	0x0	R/W	

D[15:3] Reserved

D[2:0] IILV4[2:0]: UART Interrupt Level Bits

Sets the interrupt level (0 to 7) of the UART interrupt. (Default: 0x0)
See the description of IILV0[2:0] (D[2:0]) in the ITC_ILV0 register (0x430e).

0x4314: Internal Interrupt Level Setup Register 3 (ITC_ILV3)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Internal Interrupt Level Setup Register 3 (ITC_ILV3)	**0x4314** (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10–8	**IILV7[2:0]**	I^2C interrupt level	0 to 7	0x0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**IILV6[2:0]**	SPI CH.0 interrupt level	0 to 7	0x0	R/W	

D[15:11] Reserved

D[10:8] IILV7[2:0]: I^2C Interrupt Level Bits

Sets the interrupt level (0 to 7) of the I^2C interrupt. (Default: 0x0)
See the description of IILV0[2:0] (D[2:0]) in the ITC_ILV0 register (0x430e).

D[7:3] Reserved

D[2:0] IILV6[2:0]: SPI CH.0 Interrupt Level Bits

Sets the interrupt level (0 to 7) of the SPI CH.0 interrupt. (Default: 0x0)
See the description of IILV0[2:0] (D[2:0]) in the ITC_ILV0 register (0x430e).

IV.1.8 Precautions

- To prevent another interrupt from being generated for the same cause again after generation of an interrupt, be sure to reset the interrupt flag before enabling interrupts and setting the PSR again or executing the reti instruction.
- When an interrupt is used to cancel standby mode, the S1C17 Core always executes the instruction that follows the slp or halt instruction before the interrupt is accepted. Therefore, place one nop instruction following the slp or halt instruction to ensure that the interrupt can be generated. If a memory access instruction that makes the S1C17 Core into a wait status is placed, level interrupt signals may not be accepted and the interrupt handling may not be executed.
- To ensure that the interrupt handler routine will be executed when a port input interrupt (level interrupt) is used to cancel standby mode, the port input signal must be asserted for more than the time shown below.

  (1) When the clock is stopped during SLEEP mode
  OSC3 oscillation start time + OSC3 oscillation stabilization wait time (set by the user) + 10 system clock cycle time

  (2) When the clock is not stopped during SLEEP mode, or in HALT mode
  10 system clock cycle time
- To generate an interrupt or to clear standby mode, the CMU must be configured to supply the clock required for the interrupt along with the interrupt control register settings.
  Table IV.1.8.1 lists the clock settings required for generating interrupts.

Table IV.1.8.1 Clock Settings Required for Generating Interrupts

Clock system	Mode in which the CMU register settings are effective	Clock supply status			Clock settings required for generating interrupts																						
					Normal/HALT mode																				SLEEP mode			
		Normal mode	HALT mode	SLEEP mode	AMI	DBG	NMI	MFT	ADC	T16	T8	UART	SPI	I^2C	PORT	RTC	PT8	SPI	USB	I^2S	REMC	DBG	NMI	PORT	RTC
PCLK	Normal/HALT	REG	REG	Stop				○	○	○	○	○	○	○	○	○	○	○	○	○	○			○	○
USB_CLK	Normal/HALT	REG	REG	Stop															∗6							
USB_SAPB_CLK	Normal/HALT	REG	REG	Stop															∗6							
FLASHC	HALT	Cannot be stopped	REG	Stop																						
MFT_CLK	Normal/HALT	REG	REG	Stop				○																		
PT8_CLK	Normal/HALT	REG	REG	Stop													○									
SRAMC_CLK	HALT	Cannot be stopped	REG	Stop																						
RTC_SAPB_CLK	Normal/HALT	REG	REG	Stop												○										○
PORT_CLK	Normal/HALT	REG	REG	Stop											∗4										∗5	
WDT_CLK	Normal/HALT	REG	REG	Stop			∗3																			
ADC_CLK	Normal/HALT	REG	REG	Stop					○																	
REMC_CLK	Normal/HALT	REG	REG	Stop																	○					
SPI_CLK	Normal/HALT	REG	REG	Stop															○							
CPU	–	Cannot be stopped	Stop	Stop	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/	/
RTC	–	Cannot be stopped ∗1	Cannot be stopped ∗1	Cannot be stopped ∗1												○										○
OSC1 oscillator	–	Cannot be stopped ∗2	Cannot be stopped ∗2	Cannot be stopped ∗2	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	○	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	○
OSC3 oscillator	Normal/HALT	REG	REG	REG	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7	∗7

REG: Configurable with the register
○: Clock absolutely required for generating interrupts
AMI: Address misaligned interrupt
∗1: Stops when the function is disabled in the peripheral module.
∗2: Cannot be oscillated if the external input signal level is fixed.
∗3: The clock must be supplied to generate NMI from WDT.
∗4: Required only when pulse trigger interrupt is configured.
∗5: Not required after the registers have been configured. (Level trigger interrupt only)
∗6: Not required for generating a VBUS_Changed or NonJ interrupt in Snooze status. Required for other interrupts.
∗7: The OSC1 or OSC3 oscillator circuit, which is used as the system clock source, must always be active except in SLEEP mode.

V S1C17501 TIMER MODULES

V.1 8-bit Programmable Timers (PT8)

V.1.1 Outline of the 8-bit Programmable Timers

The S1C17501 incorporates four channels of 8-bit programmable timers (PT8 CH.0–CH.3).
The 8-bit programmable timer includes an 8-bit presettable down counter and an 8-bit reload data register for setting the preset value. The timer counts down from the initial value set in the reload data register and outputs an underflow signal when the counter underflows. The underflow signal is used to generate an interrupt. The CH.0 underflow signal can also be used as the A/D converter trigger signal to start A/D conversion. The underflow period can be programmed by selecting a prescaler clock and setting reload data. This allows the application program to get any desired time intervals. The prescaler is built into the PT8 module and it generates 13 count clocks from PT8_CLK•1/1 to PT8_CLK•1/4096.
Figure V.1.1.1 shows the structure of the 8-bit programmable timers.

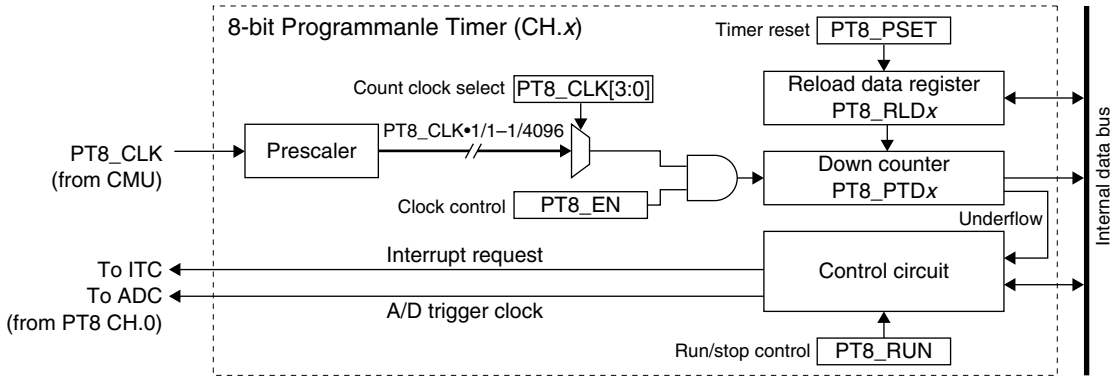


Figure V.1.1.1 Structure of 8-bit Programmable Timer (one channel)

Notes: • The descriptions in this section apply to all 8-bit programmable timer channels because they have the same functions except for the control register addresses. The '*x*' in the register names denotes a channel number (0 to 3) and the register addresses are described as (*CH.0*/*CH.1*/*CH.2*/*CH.3*).
Example: PT8_CTL*x* register (0x4803/0x4807/0x480b/0x480f)
CH.0: PT8_CTL0 register (0x4803)
CH.1: PT8_CTL1 register (0x4807)
CH.2: PT8_CTL2 register (0x480b)
CH.3: PT8_CTL3 register (0x480f)

• The prescaler in the PT8 module is also be used by the SPI CH.1.

V.1.2 Count Clock

The 8-bit programmable timer uses a prescaler output clock as the count clock. The prescaler divides PT8_CLK (with the same frequency as the system clock) by 1 to 4096 to generate 13 clocks. Select one of the prescaler output clocks using PT8_CLK[3:0] (D[3:0]/PT8_CLK*x* register).

∗ **PT8_CLK[3:0]**: PT8 Clock Division Ratio Selection Bits in the PT8 CH. *x* Input Clock Select (PT8_CLK*x*) Registers (D[3:0]/0x4800/0x4804/0x4808/0x480c)

Table V.1.2.1 Selecting the Count Clock

PT8_CLK[3:0]	Prescaler output clock	PT8_CLK[3:0]	Prescaler output clock
0xf	Reserved	0x7	PT8_CLK•1/128
0xe	Reserved	0x6	PT8_CLK•1/64
0xd	Reserved	0x5	PT8_CLK•1/32
0xc	PT8_CLK•1/4096	0x4	PT8_CLK•1/16
0xb	PT8_CLK•1/2048	0x3	PT8_CLK•1/8
0xa	PT8_CLK•1/1024	0x2	PT8_CLK•1/4
0x9	PT8_CLK•1/512	0x1	PT8_CLK•1/2
0x8	PT8_CLK•1/256	0x0	PT8_CLK•1/1

(Default: 0x0)

The selected clock is input to the counter by setting PT8_EN (D4/PT8_CLK*x* register) to 1. When PT8_EN is set to 0, the timer channel does not operate.

∗ **PT8_EN**: PT8 Clock Enable Bit in the PT8 CH.*x* Input Clock Select (PT8_CLK*x*) Registers (D4/0x4800/0x4804/0x4808/0x480c)

Note: When setting the count clock, make sure the 8-bit programmable timer counter is stopped.

V.1.3 Reload Register and Underflow Period

The Reload Data (PT8_RLD*x*) Register (0x4801/0x4805/0x4809/0x480d) is used to set the initial value to the down counter.

The counter initial value set in the reload data register is preset to the down counter when the 8-bit programmable timer is reset or when the counter underflows. When starting the 8-bit programmable timer after resetting, the timer counts down from the reload value. So the reload value and the input clock frequency determine the period of time from starting the timer until an underflow occurs (and between underflows). This makes it possible to obtain a desired wait time or a periodical interrupt interval.

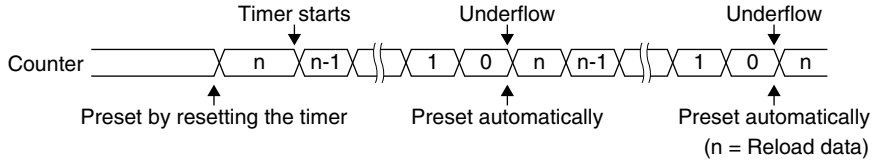


Figure V.1.3.1 Preset Timing

The underflow period is calculated by the expression below.

$$\text{Underflow period} = \frac{\text{RLD} + 1}{\text{clk\_in}}\ [\text{s}] \qquad \text{Underflow cycle} = \frac{\text{clk\_in}}{\text{RLD} + 1}\ [\text{Hz}]$$

clk_in: Count clock (prescaler output clock) frequency [Hz]
RLD: Reload data (0–255)

V.1.4 Resetting the Timer

To reset the 8-bit programmable timer, write 1 to PT8_PSET (D1/PT8_CTL*x* register). This initializes the counter by presetting the Reload Data Register value.

∗ **PT8_PSET**: Timer Reset Bit in the PT8 CH. *x* Control (PT8_CTL*x*) Registers (D1/0x4803/0x4807/0x480b/0x480f)

V.1.5 Timer Run/Stop Control

Before starting the 8-bit programmable timer, set up the conditions as shown below.

(1) Select the count clock (prescaler output clock). See Section V.1.2.
(2) Calculate the counter initial value and set it to the reload data register. See Section V.1.3.
(3) Reset the timer to preset the initial value to the counter. See Section V.1.4.
(4) Set up the interrupt level and enable the interrupt of the timer channel if the timer interrupt is used. See Section V.1.7.

To start the 8-bit programmable timer, write 1 to PT8_RUN (D0/PT8_CTL*x* register).

∗ **PT8_RUN**: Timer Run/Stop Control Bit in the PT8 CH.*x* Control (PT8_CTL*x*) Registers (D0/0x4803/0x4807/0x480b/0x480f)

The timer starts counting down from the initial value or the current counter value if the initial value has not been preset. When the counter underflows, the timer outputs an underflow pulse and presets the initial value again. At the same time, an interrupt request is sent to the interrupt controller (ITC).
The timer continues counting from the reloaded initial value.

To stop the 8-bit programmable timer from the application program, write 0 to PT8_RUN. The counter stops counting and holds the current counter value until the timer is reset or restarted. To restart counting from the initial value, reset the timer before writing 1 to PT8_RUN.

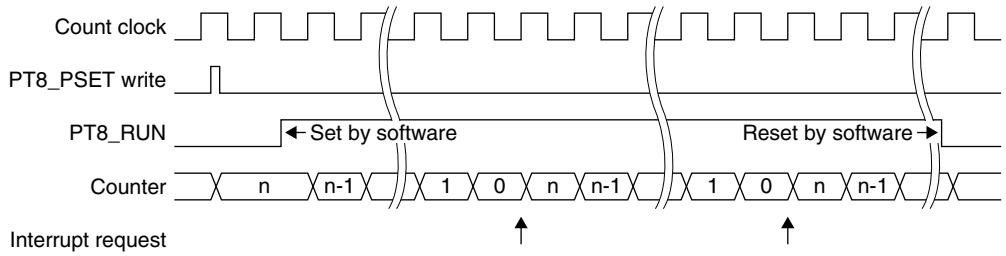


Figure V.1.5.1 Count Operation

V.1.6 Timer Output Signal

The 8-bit programmable timer outputs an underflow pulse when the counter underflows.
This pulse is used to request a timer interrupt. The CH.0 underflow pulse can also be used as the A/D converter trigger signal to start A/D conversion.

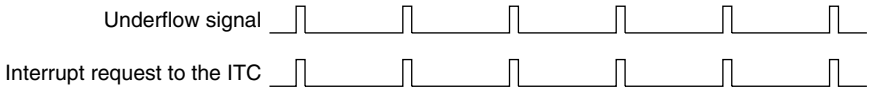


Figure V.1.6.1 Timer Output Clock

V.1.7 8-bit Programmable Timer Interrupt

The 8-bit programmable timer outputs an interrupt request signal to the interrupt controller (ITC) when the counter underflows.
To generate a timer underflow interrupt, set up the interrupt level and enable the interrupt using the ITC registers.

ITC registers for timer interrupts

Table V.1.7.1 shows the control registers of the ITC provided for each timer channel.

Table V.1.7.1 ITC Registers

Channel	Interrupt flag	Interrupt enable bit	Interrupt level setup bits
CH.0	AIFT5 (D5/ITC_AIFLG)	AIEN5 (D5/ITC_AEN)	AILV5[2:0] (D[10:8]/ITC_AILV2)
CH.1	AIFT6 (D6/ITC_AIFLG)	AIEN6 (D6/ITC_AEN)	AILV6[2:0] (D[2:0]/ITC_AILV3)
CH.2	AIFT7 (D7/ITC_AIFLG)	AIEN7 (D7/ITC_AEN)	AILV7[2:0] (D[10:8]/ITC_AILV3)
CH.3	AIFT8 (D8/ITC_AIFLG)	AIEN8 (D8/ITC_AEN)	AILV8[2:0] (D[2:0]/ITC_AILV4)

ITC_AIFLG register (0x42e0)
ITC_AEN register (0x42e2)
ITC_AILV2, ITC_AILV3, ITC_AILV4 registers (0x42ea, 0x42ec, 0x42ee)

When an underflow occurs in the timer, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the timer interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the timer underflow pulse, regardless of how the interrupt enable bit is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the timer interrupt. If the same interrupt level is set, timer CH.0 has highest priority and timer CH.3 has lowest priority.
An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The timer interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vectors

The following shows the vector numbers and vector addresses for the timer interrupt:

Table V.1.7.2 Timer Interrupt Vectors

Channel	Vector number	Vector address
CH.0	21 (0x15)	TTBR + 0x54
CH.1	22 (0x16)	TTBR + 0x58
CH.2	23 (0x17)	TTBR + 0x5c
CH.3	24 (0x18)	TTBR + 0x60

V.1.8 Details of Control Registers

Table V.1.8.1 List of 8-bit Programmable Timer Registers

Address	Register name		Function
0x4800	PT8_CLK0	PT8 CH.0 Input Clock Select Register	Selects the count clock.
0x4801	PT8_RLD0	PT8 CH.0 Reload Data Register	Sets reload data.
0x4802	PT8_PTD0	PT8 CH.0 Counter Data Register	Counter data
0x4803	PT8_CTL0	PT8 CH.0 Control Register	Sets the timer mode and starts/stops the timer.
0x4804	PT8_CLK1	PT8 CH.1 Input Clock Select Register	Selects the count clock.
0x4805	PT8_RLD1	PT8 CH.1 Reload Data Register	Sets reload data.
0x4806	PT8_PTD1	PT8 CH.1 Counter Data Register	Counter data
0x4807	PT8_CTL1	PT8 CH.1 Control Register	Sets the timer mode and starts/stops the timer.
0x4808	PT8_CLK2	PT8 CH.2 Input Clock Select Register	Selects the count clock.
0x4809	PT8_RLD2	PT8 CH.2 Reload Data Register	Sets reload data.
0x480a	PT8_PTD2	PT8 CH.2 Counter Data Register	Counter data
0x480b	PT8_CTL2	PT8 CH.2 Control Register	Sets the timer mode and starts/stops the timer.
0x480c	PT8_CLK3	PT8 CH.3 Input Clock Select Register	Selects the count clock.
0x480d	PT8_RLD3	PT8 CH.3 Reload Data Register	Sets reload data.
0x480e	PT8_PTD3	PT8 CH.3 Counter Data Register	Counter data
0x480f	PT8_CTL3	PT8 CH.3 Control Register	Sets the timer mode and starts/stops the timer.

The following describes each 8-bit programmable timer register. These are all 8-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x4800/0x4804/0x4808/0x480c: PT8 CH.*x* Input Clock Select Registers (PT8_CLK*x*)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PT8 CH.*x* Input Clock Select Register (PT8_CLK*x*)	0x4800 0x4804 0x4808 0x480c (8 bits)	D7–5	–	reserved	–		–	–	0 when being read.
		D4	PT8_EN	PT8 Clock Enable	1 Enable	0 Disable	0	R/W	
		D3–0	PT8_CLK [3:0]	PT8 clock division ratio selection (Prescaler output clock)	PT8_CLK[3:0]	Clock	0x0	R/W	
					0xf–0xd	reserved			
					0xc	PT8_CLK•1/4096			
					0xb	PT8_CLK•1/2048			
					0xa	PT8_CLK•1/1024			
					0x9	PT8_CLK•1/512			
					0x8	PT8_CLK•1/256			
					0x7	PT8_CLK•1/128			
					0x6	PT8_CLK•1/64			
					0x5	PT8_CLK•1/32			
					0x4	PT8_CLK•1/16			
					0x3	PT8_CLK•1/8			
					0x2	PT8_CLK•1/4			
					0x1	PT8_CLK•1/2			
					0x0	PT8_CLK•1/1			

Note: The letter '*x*' in register names, etc., denotes a channel number from 0 to 3.

0x4800: PT8 CH.0 Input Clock Select Register (PT8_CLK0)
0x4804: PT8 CH.1 Input Clock Select Register (PT8_CLK1)
0x4808: PT8 CH.2 Input Clock Select Register (PT8_CLK2)
0x480c: PT8 CH.3 Input Clock Select Register (PT8_CLK3)

D[7:5] Reserved

D4 PT8_EN: PT8 Clock Enable Bit

Enables the count clock input to the counter.
1 (R/W): Enable
0 (R/W): Disable (default)

Write 1 to this bit before the timer channel can start counting.

D[3:0] PT8_CLK[3:0]: PT8 Clock Division Ratio Selection Bits

These bits select the count clock of the 8-bit programmable timer from 13 prescaler output clocks.

Table V.1.8.2 Selecting the Count Clock

PT8_CLK[3:0]	Prescaler output clock	PT8_CLK[3:0]	Prescaler output clock
0xf	Reserved	0x7	PT8_CLK•1/128
0xe	Reserved	0x6	PT8_CLK•1/64
0xd	Reserved	0x5	PT8_CLK•1/32
0xc	PT8_CLK•1/4096	0x4	PT8_CLK•1/16
0xb	PT8_CLK•1/2048	0x3	PT8_CLK•1/8
0xa	PT8_CLK•1/1024	0x2	PT8_CLK•1/4
0x9	PT8_CLK•1/512	0x1	PT8_CLK•1/2
0x8	PT8_CLK•1/256	0x0	PT8_CLK•1/1

(Default: 0x0)

Note: When setting the count clock, make sure the 8-bit programmable timer counter is stopped.

0x4801/0x4805/0x4809/0x480d: PT8 CH.*x* Reload Data Registers (PT8_RLD*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.*x* Reload Data Register (PT8_RLD*x*)	**0x4801** **0x4805** **0x4809** **0x480d** (8 bits)	D7–0	**PT8_RLD [7:0]**	8-bit programmable timer reload data PT8_RLD7 = MSB PT8_RLD0 = LSB	0 to 255	X	R/W	

Note: The letter '*x*' in register names, etc., denotes a channel number from 0 to 3.

0x4801: PT8 CH.0 Reload Data Register (PT8_RLD0)
0x4805: PT8 CH.1 Reload Data Register (PT8_RLD1)
0x4809: PT8 CH.2 Reload Data Register (PT8_RLD2)
0x480d: PT8 CH.3 Reload Data Register (PT8_RLD3)

D[7:0] PT8_RLD[7:0]: 8-bit Programmable Timer Reload Data Bits

Set the initial value for the counter. (Default: undefined)

The reload data written in this register is preset to the respective counter when the timer is reset or when the counter underflows.

When starting the 8-bit programmable timer after resetting, the timer counts down from the reload value. So the reload value and the input clock frequency determine the period of time from starting the timer until an underflow occurs (and between underflows). This makes it possible to obtain a desired wait time or a periodical interrupt interval.

0x4802/0x4806/0x480a/0x480e: PT8 CH.*x* Counter Data Registers (PT8_PTD*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.*x* Counter Data Register (PT8_PTD*x*)	**0x4802** **0x4806** **0x480a** **0x480e** (8 bits)	D7–0	**PT8_PTD [7:0]**	8-bit programmable timer counter data PT8_PTD7 = MSB PT8_PTD0 = LSB	0 to 255	X	R	

Note: The letter '*x*' in register names, etc., denotes a channel number from 0 to 3.

0x4802: PT8 CH.0 Counter Data Register (PT8_PTD0)
0x4806: PT8 CH.1 Counter Data Register (PT8_PTD1)
0x480a: PT8 CH.2 Counter Data Register (PT8_PTD2)
0x480e: PT8 CH.3 Counter Data Register (PT8_PTD3)

D[7:0] PT8_PTD[7:0]: 8-bit Programmable Timer Counter Data Bits

The counter data can be read from this register. (Default: undefined)
This is a read-only register, so the writing operation is invalid.

0x4803/0x4807/0x480b/0x480f: PT8 CH.*x* Control Registers (PT8_CTL*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.*x* Control Register (PT8_CTL*x*)	0x4803 0x4807 0x480b	D7–2	–	reserved	–	–	–	0 when being read.
	0x480f	D1	PT8_PSET	Timer reset	1 Reset 0 Ignored	0	W	
	(8 bits)	D0	PT8_RUN	Timer run/stop control	1 Run 0 Stop	0	R/W	

Note: The letter '*x*' in register names, etc., denotes a channel number from 0 to 3.

0x4803: PT8 CH.0 Control Register (PT8_CTL0)
0x4807: PT8 CH.1 Control Register (PT8_CTL1)
0x480b: PT8 CH.2 Control Register (PT8_CTL2)
0x480f: PT8 CH.3 Control Register (PT8_CTL3)

D[7:2] Reserved

D1 PT8_PSET: Timer Reset Bit

Resets the 8-bit programmable timer.
1 (W): Reset
0 (W): Has no effect
0 (R): Always 0 when read (default)

Writing 1 to this bit presets the reload data in the counter. If the counter is reset when in a run state, the counter starts counting immediately after the reload data is preset.

D0 PT8_RUN: Timer Run/Stop Control Bit

Controls the timer's Run/Stop state.
1 (R/W): Run
0 (R/W): Stop (default)

The timer starts counting by writing 1 to PT8_RUN and stops counting by writing 0.
In the stop state, the counter data is retained until the timer is reset or placed in a run state.

V.1.9 Precaution

When setting the count clock, make sure the 8-bit programmable timer is turned off.

V.2 16-bit Multi-Function Timer (MFT)

V.2.1 Configuration of 16-bit Multi-Function Timer

The S1C17501 contains a 16-bit multi-function timer (MFT). The following lists the main functions of the MFT.

- 16-bit presettable up-counter
- Programmable count clocks using the prescaler embedded in the MFT module
- Event counter function using an external clock
- Interrupt generation function with programmable interrupt cycles using the period and compare data registers
- PWM output with an IGBT control function

Figure V.2.1.1 shows the structure of one channel of the MFT.

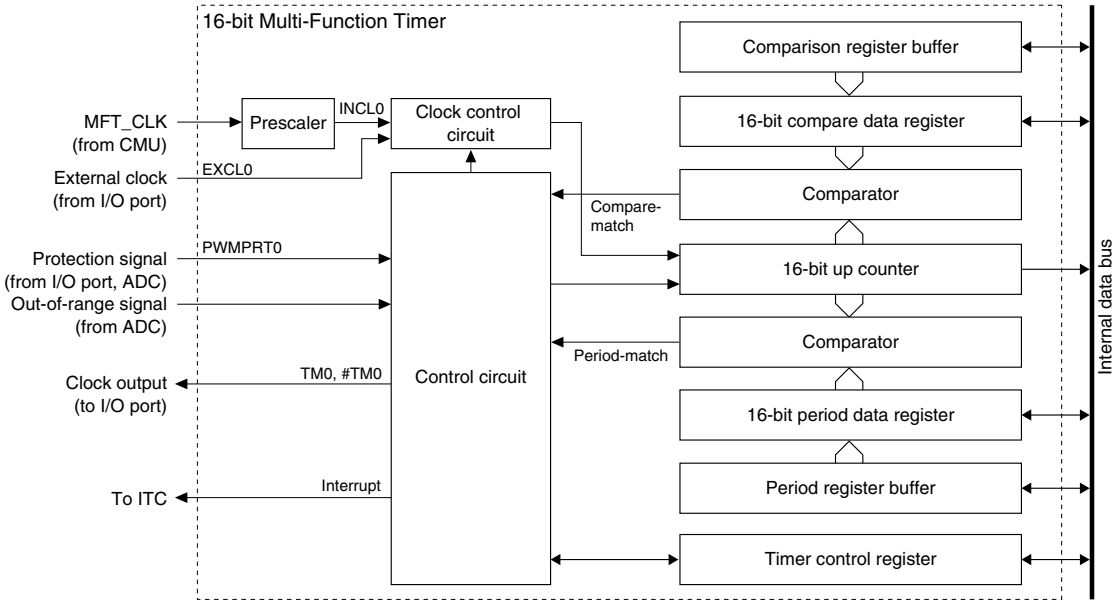


Figure V.2.1.1 Structure of 16-bit Multi-Function Timer

The MFT consists of a 16-bit up-counter (MFT_TC register), 16-bit period data and comparison data registers (MFT_PRD register, MFT_CMP register) and their buffers.

* **TC[15:0]**: Counter Data Bits in the MFT Counter Data (MFT_TC) Register (D[15:0]/0x5200)
* **PRD[15:0]**: Period Data Bits in the MFT Period Data (MFT_PRD) Register (D[15:0]/0x5202)
* **CR[15:0]**: Compare Data Bits in the MFT Compare Data (MFT_CMP) Register (D[15:0]/0x5204)

The 16-bit counter can be reset to 0 by software and counts up using the prescaler output clock or an external signal input from the I/O port. The counter value can be read by software.
The period and comparison data registers are used to store the data to be compared with the content of the up-counter. These registers can be directly read and written. Furthermore, period and comparison data can be set via the buffers. In this case, the set value is loaded to the period and comparison data register when the counter is reset by the period-match signal or software (by writing 1 to PRST (D1/MFT_CTL register)). The software can select whether period/comparison data is written to the register or the buffer.

* **PRST**: Timer Reset Bit in the MFT Control (MFT_CTL) Register (D1/0x5206)

When the counter value matches to the content of the period/comparison data register, the comparator outputs a signal that controls the interrupt and the output signal. Thus the registers allow interrupt generating intervals and the timer's output clock frequency and duty ratio to be programmed.

V.2.2 I/O Pins of 16-bit Multi-Function Timer

Table V.2.2.1 shows the input/output pins used for the MFT.

Table V.2.2.1 MFT I/O Pins

Pin name	I/O	Function
EXCL0	I	External clock input for the event counter function
TM0	O	MFT output (positive)
#TM0	O	MFT output (negative)
PWMPRT0	I	PWM channel protection input

TM0, #TM0 (output pins of the MFT)

These pins output the clock or PWM signal generated by the MFT. The TM0 pin outputs the positive signal and the #TM0 pin outputs the negative signal.

EXCL0 (event counter input pin)

When using the MFT as an event counter, input count pulses from an external source to this pin.

PWMPRT0 (PWM channel protection input pin)

This pin is used to input an output protect signal for disabling the MFT PWM output forcibly. This function can be used to control the external IGBT for protecting the chip from over-voltage, over-current, and excessive temperature. The out-of-range signal from the A/D converter is also used as the protect signal.

Note: The MFT input/output pins are shared with general-purpose I/O ports or other peripheral circuit inputs/outputs, so that functionality in the initial state is set to other than the MFT input/output. Before the MFT input/output signals assigned to these pins can be used, the function of these pins must be switched for the MFT input/output by setting the corresponding Port Function Select Registers. For details of pin functions and how to switch over, see Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

V.2.3 Uses of 16-bit Multi-Function Timer

The comparators of the MFT cyclically output a compare-match signal and a period-match signal in accordance with the comparison data and period data that are set in the software. These signals are used to generate an interrupt request to the CPU or control the internal peripheral circuits. A clock generated from the signals can also be output to external devices.

CPU interrupt request

A matching of the counter data and comparison or period data can be used as a cause of interrupt to generate an interrupt request to the CPU. Therefore, an interrupt can be generated at an interval that is set in the software.

Clock output to external devices

The positive (TM0) and negative (#TM0) clocks (PWM signals) generated from the compare-match and period-match signals can be output from the chip to the outside. The clock cycle is determined by period data, and the duty ratio is determined by comparison data. These outputs can be used to control external devices. The output pins are described in the preceding section.

The outputs are also be controlled by the protection signal input from the I/O port or A/D converter. If the protection signal is asserted, the outputs will be disabled (forcibly set to the initial levels). This function can be used to control the external IGBT for power protection.

A/D converter start trigger

The A/D converter allows a trigger to start the A/D conversion to be selected from among four available types. One is the period-match of the MFT. This makes it possible to perform the A/D conversion at programmable intervals.

To use this function, write 0b01 to the A/D converter control bit TS[1:0] (D[4:3]/AD_TRIG_CH register) to select the MFT as the trigger.

* **TS[1:0]**: A/D Conversion Trigger Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[4:3]/0x5542)

V.2.4 MFT Operating Clock

The MFT use the MFT_CLK clock (= system clock) generated by the CMU as the operating clock. The count clock is generated from the MFT_CLK by the prescaler embedded in the MFT module.

Controlling the supply of the operating clock

The MFT_CLK clock is supplied to the MFT with default settings. It can be turned off using MFT_CLK_EN (D0/CMU_GATEDCLK1 register) to reduce the amount of power consumed on the chip if MFT is not used.

* **MFT_CLK_EN**: Multi-Function Timer Clock Control Bit in the Gated Clock Control 1 (CMU_GATEDCLK1) Register (D0/0x4907)

Setting MFT_CLK_EN to 0 (1 by default) turns off the MFT_CLK clock supply to the MFT. When the clock supply is turned off, the MFT control registers cannot be accessed and the count operation stops.
For details on how to set and control the operating clock, refer to Section II.2, "Clock Management Unit (CMU)."

Clock state in standby mode

The clock supply to the MFT stops depending on type of standby mode.
HALT mode: The operating clock is supplied the same way as in normal mode.
SLEEP mode: The operating clock supply stops.

Therefore, the MFT also stops operating in SLEEP mode.

V.2.5 Control and Operation of the MFT

The following settings must first be made before the 16-bit multi-function timer starts counting:
1. Setting pins for input/output (only when necessary) ... See Sections V.2.2 and I.3.3.
2. Setting count clock
3. Selecting comparison/period data register/buffer
4. Setting clock output conditions (signal active level, initial signal level, protect input) ... See Section V.2.6.
5. Setting comparison/period data
6. Setting interrupt ... See Section V.2.7.

Setting the count clock

The count clock can be selected from between an internal clock and an external clock. Select the input clock using TCKS (D5/MFT_CTL register).

∗ **TCKS**: Input Clock Selection Bit in the MFT Control (MFT_CTL) Register (D5/0x5206)

An external clock is selected by writing 1 to TCKS, and the internal clock is selected by writing 0. At initial reset, TCKS is set for the internal clock.
An external clock can be used for the timer for which the pin is set for input.

- **Internal clock**

When the internal clock is selected, the count clock is generated from the MFT_CLK by the prescaler embedded in the MFT module.
The prescaler's division ratio can be selected from among eight ratios using TPS[2:0] (D[10:8]/MFT_CTL register). The divided clock is output from the prescaler by writing 1 to TPSON (D11/MFT_CTL register).

∗ **TPS[2:0]**: Clock Division Ratio Selection Bits in the MFT Control (MFT_CTL) Register (D[10:8]/0x5206)
∗ **TPSON**: Clock Control Bit in the MFT Control (MFT_CTL) Register (D11/0x5206)

Table V.2.5.1 Division Ratio

TPS[2:0]	Count clock
0x7	MFT_CLK•1/128
0x6	MFT_CLK•1/64
0x5	MFT_CLK•1/32
0x4	MFT_CLK•1/16
0x3	MFT_CLK•1/8
0x2	MFT_CLK•1/4
0x1	MFT_CLK•1/2
0x0	MFT_CLK•1/1

(Default: 0x0)

Notes: • When setting a count clock, make sure the MFT is turned off.

• TPSON (D11/MFT_CTL register) should be set to 0 to reduce current consumption when the MFT is not used.

- **External clock**

When using the MFT as an event counter by supplying clock pulses from an external source, make sure the event cycle is at least two MFT_CLK cycles.

Selecting comparison/period data register/buffer

The comparison data and period data registers are used to store the data to be compared with the content of the up-counter. These registers can be directly read and written. Furthermore, comparison/period data can be set via the comparison/period register buffer. In this case, the set value is loaded to the comparison/period data register when the counter is reset by the period-match signal or software (by writing 1 to PRST (D1/MFT_CTL register)).

Select whether comparison/period data is written to the comparison/period data register or the buffer using BUFEN (D7/MFT_CTL register).

* **BUFEN**: Comparison/Period Buffer Enable Bit in the MFT Control (MFT_CTL) Register (D7/5206)

When 1 is written to BUFEN, the comparison/period register buffer is selected and when 0 is written, the comparison/period data register is selected.
At initial reset, the comparison/period data register is selected.

Setting comparison and period data

The timer contains two data comparators that allow the count data to be compared with given values. CR[15:0] (D[15:0]/MFT_CMP register) and PRD[15:0] (D[15:0]/MFT_PRD register) are used to set these values.

* **CR[15:0]**: Compare Data Bits in the MFT Compare Data (MFT_CMP) Register (D[15:0]/0x5204)
* **PRD[15:0]**: Period Data Bits in the MFT Period Data (MFT_PRD) Register (D[15:0]/0x5202)

When BUFEN is set to 0, these registers allow direct reading/writing from/to the comparison and period data registers.
When BUFEN is set to 1, these registers are used to read/write from/to the comparison and period register buffers. The content of the buffer is loaded to the comparison and period data registers when the counter is reset.
At initial reset, the comparison and period data registers/buffers are initialized to 0.

The MFT compares the comparison data register and count data and, when the two values are equal, generates a compare-match signal. Also the period data register is compared with count data and, when the two values are equal, the MFT generates a period-match signal. The compare-match and period-match signals control the clock outputs (TM0 and #TM0 signals) to external devices, in addition to generating an interrupt.
The period-match signal is also used to reset the counter.

Resetting the counter

The MFT provides PRST (D1/MFT_CTL register) to reset the counter.

* **PRST**: Timer Reset Bit in the MFT Control (MFT_CTL) Register (D1/0x5206)

Normally, reset the counter before starting count-up by writing 1 to this control bit.
After the counter starts counting, it will be reset when a period-match occurs.

Timer RUN/STOP control

The MFT provides TMEN (D0/MFT_CTL register) to control RUN/STOP.

* **TMEN**: Timer Run/Stop Control Bit in the MFT Control (MFT_CTL) Register (D0/0x5206)

The timer starts counting when 1 is written to TMEN. The clock input is disabled and the timer stops counting when 0 is written to TMEN.
This RUN/STOP control does not affect the counter data. Even when the timer has stopped counting, the counter retains its count so that the timer can start counting again from that point.
If the count of the counter matches the set value of the comparison data register during count-up, the timer generates a compare-match interrupt.
When the counter matches period data, an interrupt is generated and the counter is reset. At the same time, the values set in the compare and period register buffers are loaded to the compare and period data registers if BUFEN is set to 1.
The counter continues counting up regardless of which interrupt has occurred. In the case of a period-match interrupt, the counter starts counting beginning with 0.
When both PRST and TMEN are set to 1 at the same time, the timer starts counting after resetting the counter.

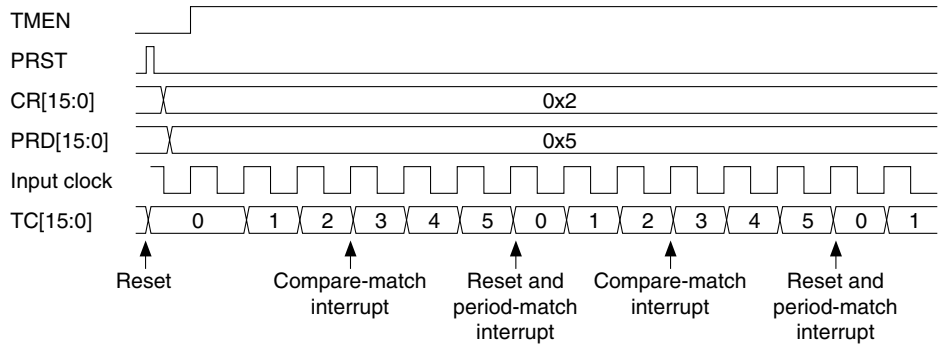


Figure V.2.5.1 Basic Operation Timing of Counter

Reading counter data

The counter data can be read out from TC[15:0] (D[15:0]/MFT_TC register) at any time.

* **TC[15:0]**: Counter Data Bits in the MFT Counter Data (MFT_TC) Register (D[15:0]/0x5200)

Writing counter data

Counter data can be written to TC[15:0] at any time. This makes it possible to change the interrupt and/or clock output cycles temporarily.

V.2.6 Controlling Clock Output

The timers can generate the TM0 and #TM0 signals using the compare-match and period-match signals from the counter.
Figure V.2.6.1 shows the MFT clock output circuit.

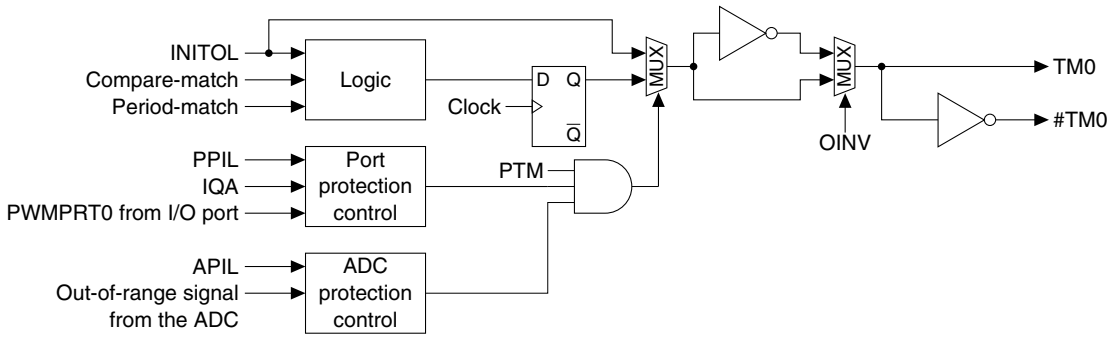


Figure V.2.6.1 MFT Clock Output Circuit

Setting the initial output level

The default output level while the clock output is turned off is 0 (low level). This level can be changed to 1 (high level) using INITOL (D1/MFT_IOCTL register).

∗ **INITOL**: Initial Output Level Bit in the MFT Input/Output Control (MFT_IOCTL) Register (D1/0x521e)

When INITOL is 0 (default), the initial output level is low. When INITOL is set to 1, the initial output level is set to high.
The timer output goes to the initial output level when the timer output is turned off.

Setting the signal active level

By default, an active high signal (normal low) is generated. This logic can be inverted using OINV (D0/MFT_IOCTL register). When 1 is written to OINV, the timer generates an active low (normal high) signal.

∗ **OINV**: Inverse Output Bit in the MFT Input/Output Control (MFT_IOCTL) Register (D0/0x521e)

Note that the initial output level set by INITOL is inverted when OINV is set to 1.
See Figure V.2.6.2 for the waveforms.

Setting the output port

The TM0 (#TM0) signal generated here can be output from the clock output pins (see Table V.2.2.1), enabling a programmable clock to be supplied to external devices.
After an initial reset, the output pins are set for the I/O ports and set in input mode. The pins go into high-impedance status.
When the pin function is switched to the timer output, the pin outputs the level according to the set values of INITOL and OINIV. The output pin holds this level until the output level changes due to the counter value after the timer output is enabled.

Table V.2.6.1 Initial Output Level

INITOL	OINV	Initial output level
1	1	Low
1	0	High
0	1	High
0	0	Low

Starting clock output

To output the TM0 (#TM0) clock, write 1 to the clock output control bit PTM (D2/MFT_IOCTL register). Clock output is stopped by writing 0 to PTM and goes to the initial output level according to the set values of INITOL and OINIV.

∗ **PTM**: Clock Output Enable Bit in the MFT Input/Output Control (MFT_IOCTL) Register (D2/0x521e)

Figure V.2.6.2 shows the waveform of the output signal.

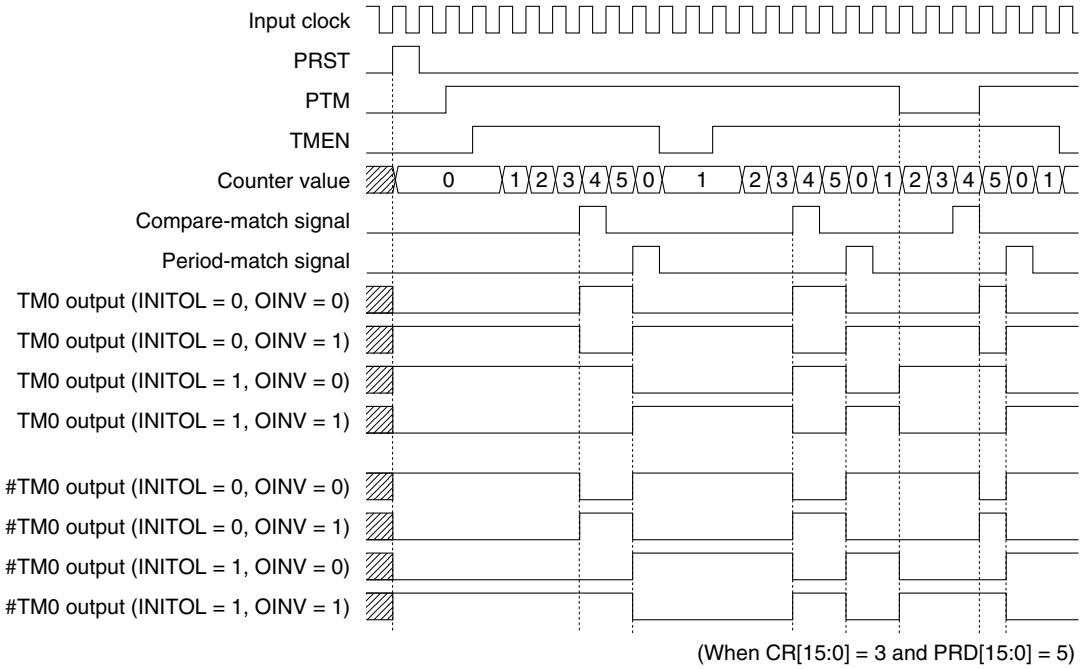


Figure V.2.6.2 Waveform of MFT Output

TM0 output when OINV = 0 (active high):

The timer outputs a low level (initial output level when output is started) until the counter becomes equal to the comparison data set in CR[15:0] (D[15:0]/MFT_CMP register). When the counter is incremented to the next value from the comparison data, the TM0 output pin goes high and a compare-match interrupt occurs. When the counter becomes equal to the period data set in PRD[15:0] (D[15:0]/MFT_PRD register), the counter is reset and the TM0 output pin goes low. At the same time a period-match interrupt occurs.

∗ **CR[15:0]**: Compare Data Bits in the MFT Compare Data (MFT_CMP) Register (D[15:0]/0x5204)
∗ **PRD[15:0]**: Period Data Bits in the MFT Period Data (MFT_PRD) Register (D[15:0]/0x5202)

TM0 output when OINV = 1 (active low):

The timer outputs a high level (inverted initial output level when output is started) until the counter becomes equal to the comparison data set in CR[15:0] (D[15:0]/MFT_CMP register). When the counter is incremented to the next value from the comparison data, the output pin goes low and a compare-match interrupt occurs. When the counter becomes equal to the period data set in PRD[15:0] (D[15:0]/MFT_PRD register), the counter is reset and the output pin goes high. At the same time a period-match interrupt occurs.

Output protection for IGBT control

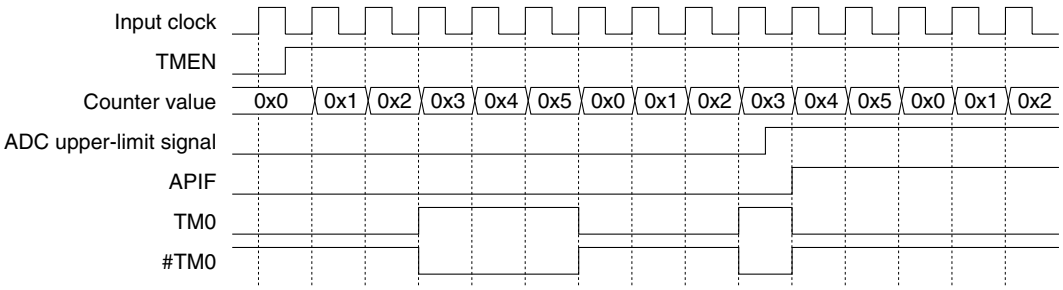
The MFT provides a power protection feature with the interrupt for safe operation of the external IGBT output. If the power protection is enabled and activated, the PWM output pins will be put in its initial status immediately. At the same time, an interrupt will also be generated. The interrupt can be used to inform the monitoring program of IGBT drive abnormalities such as over-voltage, over-current, and excessive temperature. The ADC out-of-range signal or a port input can be used to control the power protection.

ADC protection

Setting APIE (D1/MFT_IE register) to 1 enables the ADC protection that uses ADC CH.0 as the protection detection input. The ADC allows the application to check whether the conversion results of the specified channel are within the specified range or not. If the conversion result is higher than the upper-limit value set with software or is lower than the lower-limit value set with software, the ADC outputs the out-of-range signal and generates an interrupt. The MFT uses the out-of-range signal generated by the ADC CH.0 to control the power protection. Use APIL (D6/MFT_IOCTL register) to select upper-limit protection or lower-limit protection mode. Set APIL to 0 (default) to disable the PWM output (setting the output to the initial level) when the A/D conversion result is higher than the upper-limit value, or set APIL to 1 to disable the PWM output when the A/D conversion result is lower than the lower-limit value. The upper- and lower-limit values can be set to the AD_UPPER (0x5558) and AD_LOWER (0x555a) registers in the ADC module. When the designated out-of-range signal is asserted by the ADC, APIF (D1/MFT_IF register) is set to 1. At the same time, the MFT forcibly sets the PWM output pins to the initial status (PWM outputs are disabled) if APIE has been set to 1.

- **APIE**: ADC Protection Interrupt Enable Bit in the MFT Interrupt Enable (MFT_IE) Register (D1/0x5230)
- **APIF**: ADC Protection Interrupt Flag Bit in the MFT Interrupt Flag (MFT_IF) Register (D1/0x5238)
- **APIL**: ADC Protection Input Selection Bit in the MFT Input/Output Control (MFT_IOCTL) Register (D6/0x521e)

Figure V.2.6.3 shows the ADC protection timing (upper limit protection).



(When CR[15:0] = 2, PRD[15:0] = 5, and INITOL = OINV = 0)

Figure V.2.6.3 ADC Protection for IGBT Output

Port protection

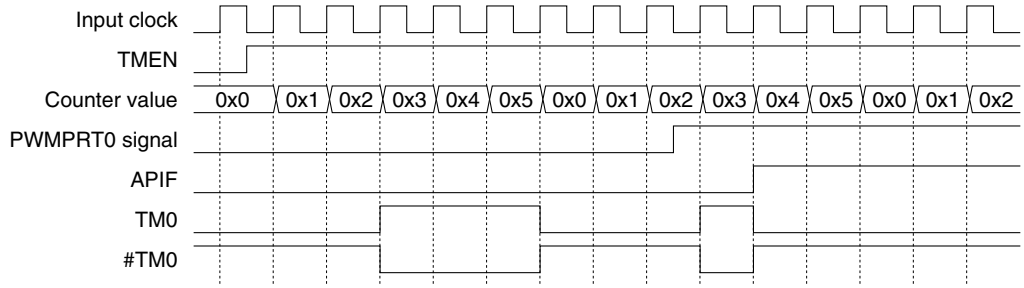
The MFT also supports a port protection that uses the dedicated input port PWMPRT0. Setting PPIE (D0/MFT_IE register) to 1 enables the port protection. Either high or low level can be selected as the protection input signal level using PPIL (D5/MFT_IOCTL register). When PPIL is set to 1, the PWM output is disabled and PPIF (D0/MFT_IF register) is set to 1 to generate a port protection interrupt when the PWMPRT0 input signal goes low. When PPIL is set to 0 (default), the PWM output is disabled and PPIF is set to 1 when the PWMPRT0 input signal goes high.

- **PPIE**: Port Protection Interrupt Enable Bit in the MFT Interrupt Enable (MFT_IE) Register (D0/0x5230)
- **PPIF**: Port Protection Interrupt Flag Bit in the MFT Interrupt Flag (MFT_IF) Register (D0/0x5238)
- **PPIL**: Port Protection Input Level Selection Bit in the MFT Input/Output Control (MFT_IOCTL) Register (D5/0x521e)

To avoid the port protection to be activated due to noise, the MFT provides a noise filter at the input of the PWMPRT0 pin. The pulse width to be removed as noise can be selected using IQA (D4/MFT_IOCTL register). When IQA is set to 1, a pulse shorter than 12 system clocks are regarded as noise and it is not accepted as the valid input. When IQA is set to 0 (default), a pulse shorter than 6 system clocks are regarded as noise.

- **IQA**: Port Protection Input Noise Filter Bit in the MFT Input/Output Control (MFT_IOCTL) Register (D4/0x521e)

Figure V.2.6.4 shows the port protection timing (high level protection).



(When CR[15:0] = 2, PRD[15:0] = 5, and INITOL = OINV = 0)

Figure V.2.6.4 Port Protection for IGBT Output

Precautions

(1) If a same value is set to the comparison data and period data registers, a hazard may be generated in the output signal. Therefore, do not set the data registers as MFT_CMP = MFT_PRD.
There is no problem when the interrupt function only is used.

(2) When using the output clock, set the comparison and period data registers as MFT_CMP ≥ 0 and MFT_PRD ≥ 1. The minimum settings are MFT_CMP = 0 and MFT_PRD = 1. In this case, the timer output clock cycle is the input clock × 1/2.

(3) When the comparison and period data registers are set as MFT_CMP > MFT_PRD, no compare-match signal is generated. In this case, the output signal is fixed at the off level.

V.2.7 MFT Interrupts

The MFT module can generate the following four types of interrupts:
• Compare-match interrupt
• Period-match interrupt
• ADC protection interrupt
• Port protection interrupt

The MFT module has one interrupt signal to be output to the interrupt controller (ITC) and it is shared with the four causes of interrupt. To determine the cause of interrupt that has occurred, read the interrupt flags in the MFT module.

Compare-match interrupt

This interrupt request occurs when the count of the counter matches the set value of the compare data register during count-up, and it sets the interrupt flag CMPIF (D3/MFT_IF register) in the MFT module to 1.

∗ **CMPIF**: Compare-Match Interrupt Flag Bit in the MFT Interrupt Flag (MFT_IF) Register (D3/0x5238)

Set CMPIE (D3/MFT_IE register) to 1 when using this interrupt. If CMPIE is set to 0 (default), CMPIF will not be set to 1 and an interrupt request by this cause will not be sent to the ITC.

∗ **CMPIE**: Compare-Match Interrupt Enable Bit in the MFT Interrupt Enable (MFT_IE) Register (D3/0x5230)

If CMPIF is set to 1, the MFT module outputs the interrupt request signal to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The MFT interrupt handler routine should read the CMPIF flag to check if the interrupt has occurred due to a compare-match or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) CMPIF in the MFT module, not the MFT interrupt flag in the ITC, to clear the cause of interrupt.

Period-match interrupt

This interrupt request occurs when the count of the counter matches the set value of the period data register during count-up, and it sets the interrupt flag PRDIF (D2/MFT_IF register) in the MFT module to 1.

∗ **PRDIF**: Period-Match Interrupt Flag Bit in the MFT Interrupt Flag (MFT_IF) Register (D2/0x5238)

Set PRDIE (D2/MFT_IE register) to 1 when using this interrupt. If PRDIE is set to 0 (default), PRDIF will not be set to 1 and an interrupt request by this cause will not be sent to the ITC.

∗ **PRDIE**: Period-Match Interrupt Enable Bit in the MFT Interrupt Enable (MFT_IE) Register (D2/0x5230)

If PRDIF is set to 1, the MFT module outputs the interrupt request signal to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The MFT interrupt handler routine should read the PRDIF flag to check if the interrupt has occurred due to a period-match or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) PRDIF in the MFT module, not the MFT interrupt flag in the ITC, to clear the cause of interrupt.

ADC protect interrupt

This interrupt request occurs when the A/D converter CH.0 asserts the out-of-range signal (upper-limit or lower-limit mode selectable) input to the MFT after an A/D conversion. This signal indicates that the A/D conversion result falls outside the range specified with software. It sets the interrupt flag APIF (D1/MFT_IF register) in the MFT module to 1.

* **APIF**: ADC Protection Interrupt Flag Bit in the MFT Interrupt Flag (MFT_IF) Register (D1/0x5238)

Set APIE (D1/MFT_IE register) to 1 when using this interrupt and disabling the PWM output. Although APIF will be set to 1 by the input signal even if APIE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC and the PWM output will not be changed to the initial value.

* **APIE**: ADC Protection Interrupt Enable Bit in the MFT Interrupt Enable (MFT_IE) Register (D1/0x5230)

If APIF is set to 1 when APIE has been set to 1, the MFT module outputs the interrupt request signal to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The MFT interrupt handler routine should read the APIF flag to check if the interrupt has occurred by the ADC protect input or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) APIF in the MFT module, not the MFT interrupt flag in the ITC, to clear the cause of interrupt.

Port protect interrupt

This interrupt request occurs when the specified level (high or low) of a signal is input to the PWMPRT0 pin. This signal sets the interrupt flag PPIF (D0/MFT_IF register) in the MFT module to 1.

* **PPIF**: Port Protection Interrupt Flag Bit in the MFT Interrupt Flag (MFT_IF) Register (D0/0x5238)

Set PPIE (D0/MFT_IE register) to 1 when using this interrupt and disabling the PWM output. Although PPIF will be set to 1 by the input signal even if PPIE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC and the PWM output will not be changed to the initial value.

* **PPIE**: Port Protection Interrupt Enable Bit in the MFT Interrupt Enable (MFT_IE) Register (D0/0x5230)

If PPIF is set to 1 when PPIE has been set to 1, the MFT module outputs the interrupt request signal to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The MFT interrupt handler routine should read the PPIF flag to check if the interrupt has occurred by the port protect input or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) PPIF in the MFT module, not the MFT interrupt flag in the ITC, to clear the cause of interrupt.

Note: To avoid occurrence of unnecessary interrupts, be sure to reset the CMPIF, PRDIF, APIF, or PPIF flag before the MFT interrupt is enabled using CMPIE, PRDIE, APIE, or PPIE.

ITC registers for MFT interrupt

When an enabled cause of MFT interrupt occurs according to the interrupt condition settings shown above, the MFT module asserts the interrupt signal sent to the ITC.
To generate an MFT interrupt, set the interrupt level and enable the interrupt using the ITC registers.
The following shows the control bits for the MFT interrupt in the ITC.

Interrupt flag in the ITC

- **AIFT0**: MFT Interrupt Flag Bit in the Additional Interrupt Flag (ITC_AIFLG) Register (D0/0x42e0)

Interrupt enable bit in the ITC

- **AIEN0**: MFT Interrupt Enable Bit in the Additional Interrupt Enable (ITC_AEN) Register (D0/0x42e2)

Interrupt level setup bits in the ITC

- **AILV0[2:0]**: MFT Interrupt Level Bits in the Additional Interrupt Level Setup (ITC_AILV0) Register 0 (D[2:0]/0x42e6)

When an enabled cause of MFT interrupt occurs, AIFT0 is set to 1. If AIEN0 has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the MFT interrupt, set AIEN0 to 0. AIFT0 is always set to 1 by the interrupt signal sent from the MFT module, regardless of how AIEN0 is set (even when set to 0).
AILV0[2:0] sets the interrupt level (0 to 7) of the MFT interrupt.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The MFT interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, refer to Section IV.1 “Interrupt Controller (ITC).”

Note: After an MFT interrupt occurs, reset the CMPIF, PRDIF, APIF, or PPIF interrupt flag of the MFT module in the interrupt handler routine (this also resets the interrupt flag in the ITC).

Interrupt vector

The following shows the vector number and vector address for the MFT interrupt:

Vector number: 8 (0x8)
Vector address: TTBR + 0x20

V.2.8 Details of Control Registers

Table V.2.8.1 List of MFT Registers

Address	Register name		Function
0x5200	MFT_TC	MFT Counter Data Register	Counter data
0x5202	MFT_PRD	MFT Period Data Register	Sets period data.
0x5204	MFT_CMP	MFT Compare Data Register	Sets compare data.
0x5206	MFT_CTL	MFT Control Register	Sets the timer mode and starts/stops the timer.
0x521e	MFT_IOCTL	MFT Input/Output Control Register	Controls the clock input/output.
0x5230	MFT_IE	MFT Interrupt Enable Register	Enables the MFT interrupt.
0x5238	MFT_IF	MFT Interrupt Flag Register	Indicates the MFT interrupt status.
0x527e	MFT_TST	MFT Test Register	Controls the MFT test.

The following describes each MFT register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x5200: MFT Counter Data Register (MFT_TC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MFT Counter Data Register (MFT_TC)	**0x5200** (16 bits)	D15–0	**TC[15:0]**	Counter data TC15 = MSB TC0 = LSB	0x0 to 0xffff	0x0	R/W	

D[15:0] TC[15:0]: Counter Data Bits

The counter data can be read from this register. (Default: 0x0)
Furthermore, data can be set to the counter by writing it to this register.

0x5202: MFT Period Data Register (MFT_PRD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MFT Period Data Register (MFT_PRD)	**0x5202** (16 bits)	D15–0	**PRD[15:0]**	Period data PRD15 = MSB PRD0 = LSB	0x0 to 0xffff	0x0	R/W	

D[15:0] PRD[15:0]: Period Data Bits

Sets the period data for the MFT. (Default: 0x0)

When BUFEN (D7/MFT_CTL register) is set to 0, period data is directly read or written from/to the period data register.

When BUFEN is set to 1, period data is read or written from/to the period data buffer through this address. The content of the buffer is loaded to the period data register when the counter is reset.

The data set in this register is compared with the counter data. When the contents match, a cause of period-match interrupt is generated and the output signal rises (OINV (D0/MFT_IOCTL register) = 0) or falls (OINV = 1). Furthermore, the counter is reset to 0.

0x5204: MFT Compare Data Register (MFT_CMP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MFT Compare Data Register (MFT_CMP)	**0x5204** (16 bits)	D15–0	**CR[15:0]**	Compare data CR15 = MSB CR0 = LSB	0x0 to 0xffff	0x0	R/W	

D[15:0] CR[15:0]: Compare Data Bits

Sets the compare data for the MFT. (Default: 0x0)

When BUFEN (D7/MFT_CTL register) is set to 0, compare data is directly read or written from/to the compare data register.

When BUFEN is set to 1, compare data is read or written from/to the compare data buffer through this address. The content of the buffer is loaded to the compare data register when the counter is reset.

The data set in this register is compared with the counter data. When the contents match, a cause of compare-match interrupt is generated and the output signal rises (OINV (D0/MFT_IOCTL register) = 0) or falls (OINV = 1). This does not affect the counter value and count-up operation.

0x5206: MFT Control Register (MFT_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MFT Control Register (MFT_CTL)	0x5206 (16 bits)	D15–12	–	reserved	–	–	–	0 when being read.
		D11	TPSON	Clock control	1 On 0 Off	0	R/W	
		D10–8	TPS[2:0]	Clock division ratio selection (Prescaler output clock)	TPS[2:0] / Count clock: 0x7 MFT_CLK•1/128; 0x6 MFT_CLK•1/64; 0x5 MFT_CLK•1/32; 0x4 MFT_CLK•1/16; 0x3 MFT_CLK•1/8; 0x2 MFT_CLK•1/4; 0x1 MFT_CLK•1/2; 0x0 MFT_CLK•1/1	0x0	R/W	
		D7	BUFEN	Comparison/period buffer enable	1 Enable 0 Disable	0	R/W	
		D6	–	reserved	–	–	–	0 when being read.
		D5	TCKS	Input clock selection	1 External 0 Internal	0	R/W	
		D4–2	–	reserved	–	–	–	0 when being read.
		D1	PRST	Timer reset	1 Reset 0 Ignored	0	W	
		D0	TMEN	Timer run/stop control	1 Run 0 Stop	0	R/W	

D[15:12] Reserved

D11 TPSON: MFT Clock Control Bit

Enables the MFT prescaler to output the count clock.
1 (R/W): On
0 (R/W): Off (default)

Write 1 to this bit before the MFT can start counting.

D[10:8] TPS[2:0]: MFT Clock Division Ratio Selection Bits

These bits select the count clock of the MFT from 8 prescaler output clocks.

Table V.2.8.2 Selecting the Count Clock

TPS[2:0]	Count clock
0x7	MFT_CLK•1/128
0x6	MFT_CLK•1/64
0x5	MFT_CLK•1/32
0x4	MFT_CLK•1/16
0x3	MFT_CLK•1/8
0x2	MFT_CLK•1/4
0x1	MFT_CLK•1/2
0x0	MFT_CLK•1/1

(Default: 0x0)

Note: When setting the count clock, make sure the MFT counter is stopped.

D7 BUFEN: Comparison/Period Data Buffer Enable Bit

Enables or disables writing to the compare/period data buffer.
1 (R/W): Enabled
0 (R/W): Disabled (default)

When BUFEN is set to 1, compare and period data are read and written from/to the compare and period data buffers. The contents of the buffers are loaded to the compare and period data registers when the counter is reset by the software or the period-match signal.
When BUFEN is set to 0, compare and period data are read and written from/to the compare and period data registers.

D6 Reserved

D5 TCKS: Input Clock Selection Bit

Selects the input clock for the MFT.
1 (R/W): External clock
0 (R/W): Internal clock (default)

The internal clock (prescaler output) is selected for the input clock of the timer by writing 0 to TCKS. An external clock (one that is fed from the EXCL0 pin) is selected by writing 1, and the timer functions as an event counter. In this case, the clock input pin must be configured using the corresponding port function select register before an external clock is selected here.

D[4:2] Reserved

D1 PRST: Timer Reset Bit

Resets the counter.
1 (W): Reset
0 (W): Has no effect
0 (R): Always 0 when read (default)

Writing 1 to PRST resets the counter in the MFT.

D0 TMEN: Timer Run/Stop Control Bit

Controls the timer's Run/Stop state.
1 (R/W): Run
0 (R/W): Stop (default)

The MFT starts counting up by writing 1 to TMEN and stops by writing 0.
In stop state, the counter data is retained until the timer is reset or placed in a run state. By changing states from stop to run, the timer can restart counting beginning at the retained count.

0x521e: MFT Input/Output Control Register (MFT_IOCTL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
MFT Input/ Output Control Register (MFT_IOCTL)	0x521e (16 bits)	D15–7	–	reserved	–				–	–	0 when being read.
		D6	APIL	ADC protection input selection	1	Lower limit	0	Upper limit	0	R/W	
		D5	PPIL	Port protection input level selection	1	Low level	0	High level	0	R/W	
		D4	IQA	Port protection input noise filter	1	12 clocks	0	6 clocks	0	R/W	
		D3	–	reserved	–				–	–	0 when being read.
		D2	PTM	Clock output enable	1	Enable	0	Disable	0	R/W	
		D1	INITOL	Initial output level	1	High	0	Low	0	R/W	
		D0	OINV	Inverse output	1	Invert	0	Normal	0	R/W	

D[15:7] Reserved

D6 APIL: ADC Protection Input Selection Bit

Selects the ADC out-of-range signal for power protection.
1 (R/W): Lower limit signal
0 (R/W): Upper limit signal (default)

When using the ADC protection function that disables the PWM output when an A/D conversion result is out-of-range, select the ADC output signal to activate the protection from the lower-limit and upper-limit signals using APIL.
The ADC protection function is enabled by setting APIE (D1/MFT_IE register) to 1.

D5 PPIL: Port Protection Input Level Selection Bit

Selects the port input signal level for power protection.
1 (R/W): Low level
0 (R/W): High level (default)

When using the port protection function that disables the PWM output by the PWMPRT0 input signal, select the input signal level to activate the protection.
The port protection function is enabled by setting PPIE (D0/MFT_IE register) to 1.

D4 IQA: Port Protection Input Noise Filter Bit

Configures the noise filter for the port protection input signal.
1 (R/W): 12 clocks
0 (R/W): 6 clocks (default)

To avoid the port protection to be activated due to noise, the MFT provides a noise filter at the input of the PWMPRT0 pin. The pulse width to be removed as noise can be selected using IQA. When IQA is set to 1, a pulse shorter than 12 system clocks are regarded as noise and it is not accepted as the valid input. When IQA is set to 0, a pulse shorter than 6 system clocks are regarded as noise.

D3 Reserved

D2 PTM: Clock Output Enable Bit

Controls the output of the TM0 and #TM0 signals (timer output clocks).
1 (R/W): Enable
0 (R/W): Disable (default)

The TM0 and #TM0 signal outputs are enabled by writing 1 to PTM. The clock outputs are stopped by writing 0 to PTM and go to the off level according to the set values of OINV (D0) and INITOL (D1). In this case, the clock output pins must be configured using the corresponding port function select register before outputting the TM0 and #TM0 signals here.

D1 INITOL: Initial Output Level Bit

Selects an initial output level for timer output.
1 (R/W): High
0 (R/W): Low (default)

The timer output pin goes to the initial output level set using this bit when the timer output is turned off by writing 0 to PTM (D2) or when the timer is reset by writing 1 to PRST (D1/MFT_CTL register). However, this level is inverted if OINV (D0) is set to 1.

D0 OINV: Inverse Output Bit

Selects a logic of the output signal.
1 (R/W): Inverted (active low)
0 (R/W): Normal (active high) (default)

By writing 1 to OINV, an active-low signal (off level = high) is generated for the TM0 output (active-high for #TM0). When OINV is set to 0, an active-high signal (off level = low) is generated for the TM0 output (active-low for #TM0).
Writing 1 to this bit inverts the initial output level set using INITOL (D1) as well.

0x5230: MFT Interrupt Enable Register (MFT_IE)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
MFT Interrupt Enable Register (MFT_IE)	0x5230 (16 bits)	D15–4	–	reserved	–				–	–	0 when being read.
		D3	CMPIE	Compare-match interrupt enable	1	Enable	0	Disable	0	R/W	
		D2	PRDIE	Period-match interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	APIE	ADC protection interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	PPIE	Port protection interrupt enable	1	Enable	0	Disable	0	R/W	

D[15:4] Reserved

D3 CMPIE: Compare-Match Interrupt Enable Bit

Enables/disables the compare-match interrupt.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting CMPIE to 1 enables the compare-match interrupt; setting to 0 disables the interrupt.
In addition, it is necessary to set the MFT interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

D2 PRDIE: Period-Match Interrupt Enable Bit

Enables/disables the period-match interrupt.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting PRDIE to 1 enables the period-match interrupt; setting to 0 disables the interrupt.
In addition, it is necessary to set the MFT interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

D1 APIE: ADC Protection Interrupt Enable Bit

Enables/disables the ADC protection interrupt.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting APIE to 1 enables the ADC protection function and its interrupt; setting to 0 disables the ADC protection function and the interrupt.
In addition, it is necessary to set the MFT interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

D0 PPIE: Port Protection Interrupt Enable Bit

Enables/disables the port protection interrupt.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting PPIE to 1 enables the port protection function and its interrupt; setting to 0 disables the port protection function and the interrupt.
In addition, it is necessary to set the MFT interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

0x5238: MFT Interrupt Flag Register (MFT_IF)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
MFT Interrupt Flag Register (MFT_IF)	0x5238 (16 bits)	D15–4	–	reserved	–				–	–	0 when being read.
		D3	CMPIF	Compare-match interrupt flag	1	Cause of interrupt occurred	0	Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D2	PRDIF	Period-match interrupt flag					0	R/W	
		D1	APIF	ADC protection interrupt flag					0	R/W	
		D0	PPIF	Port protection interrupt flag					0	R/W	

D[15:4] Reserved

D3 CMPIF: Compare-Match Interrupt Flag Bit

This is the interrupt flag to indicate the compare-match interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Has no effect

CMPIF is the interrupt flag for the compare-match interrupt. The interrupt flag is set to 1 when the count of the counter matches the set value of the compare data register during count-up if CMPIE (D3/MFT_IE register) has been set to 1. At the same time, the MFT interrupt request signal is output to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
CMPIF is reset by writing 1.

D2 PRDIF: Period-Match Interrupt Flag Bit

This is the interrupt flag to indicate the period-match interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Has no effect

PRDIF is the interrupt flag for the period-match interrupt. The interrupt flag is set to 1 when the count of the counter matches the set value of the period data register during count-up if PRDIE (D2/MFT_IE register) has been set to 1. At the same time, the MFT interrupt request signal is output to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
PRDIF is reset by writing 1.

D1 APIF: ADC Protection Interrupt Flag Bit

This is the interrupt flag to indicate the ADC protection interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Has no effect

APIF is the interrupt flag for the ADC protection interrupt. The interrupt flag is set to 1 when the A/D converter CH.0 asserts the out-of-range signal (upper-limit or lower-limit mode selectable) input to the MFT after an A/D conversion if APIE (D1/MFT_IE register) has been set to 1. At the same time, the MFT interrupt request signal is output to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
APIF is reset by writing 1.

D0 PPIF: Port Protection Interrupt Flag Bit

This is the interrupt flag to indicate the port protection interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Has no effect

PPIF is the interrupt flag for the port protection interrupt. The interrupt flag is set to 1 when the specified level (high or low) of a signal is input to the PWMPRT0 pin if PPIE (D0/MFT_IE register) has been set to 1. At the same time, the MFT interrupt request signal is output to the ITC. The interrupt request signal sets the MFT interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
PPIF is reset by writing 1.

Notes:
- After an MFT interrupt occurs, reset the PRDIF, CMPIF, APIF, or PPIF interrupt flag of the MFT module in the interrupt handler routine (this also resets the interrupt flag in the ITC).
- To avoid occurrence of unnecessary interrupts, be sure to reset the CMPIF, PRDIF, APIF, or PPIF flag before the MFT interrupt is enabled using CMPIE, PRDIE, APIE, or PPIE (D3, D2, D1, D0/MFT_IE register).

0x527e: MFT Test Register (MFT_TST)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MFT Test Register (MFT_TST)	0x527e (16 bits)	D15–7	–	reserved	–	–	–	0 when being read.
		D6	DBGMD	MFT operation in debug mode	1 Stop 0 Run	0	R/W	
		D5–3	–	reserved	–	–	–	0 when being read.
		D2–0	–	reserved	0x7	0x7	–	Fix at 0x7.

D[15:7] Reserved

D6 DBGMD: MFT Operation in Debug Mode Bit

Selects the MFT operation in debug mode.
1 (R/W): Stop
0 (R/W): Run (default)

If DBGMD is set to 0, the MFT operates in debug mode. If DBGMD is set to 1, the MFT stops operating when the S1C17 Core enters debug mode.

D[5:3] Reserved

D[2:0] Reserved (Always set these bits to 0x7.)

V.2.9 Precautions

- When setting the count clock, make sure the MFT is turned off.
- If a same value is set to the comparison and period data registers, a hazard may be generated in the output signal. Therefore, do not set the registers as MFT_CMP = MFT_PRD.
  There is no problem when the interrupt function only is used.
- When using the output clock, set the comparison and period data registers as MFT_CMP ≥ 0 and MFT_PRD ≥ 1. The minimum settings are MFT_CMP = 0 and MFT_PRD = 1. In this case, the timer output clock cycle is the input clock × 1/2.
- When the comparison and period data registers are set as MFT_CMP > MFT_PRD, no compare-match interrupt is generated. In this case, the output signal is fixed at the off level.
- To prevent another interrupt from being generated by the same cause of interrupt after an interrupt has occurred, be sure to reset the interrupt flag before setting the PSR again or executing the reti instruction.

V.3 Watchdog Timer (WDT)

V.3.1 Configuration of the Watchdog Timer

The S1C17501 incorporates a watchdog timer to detect the CPU running uncontrollably. The watchdog timer consists of a 30-bit up counter and comparison data register for generating an NMI or internal reset signal at programmable cycles.
By resetting the watchdog timer within such a cycle in software so as not to generate NMI or internal reset signals, it is possible to detect a program running uncontrollably that does not execute that processing routine.
The WDT clock (= system clock) or external clock input for the MFT (EXCL0) can be selected as the count clock for the watchdog timer.
Moreover, a clock can be generated synchronously with NMI/reset generation cycles (set by the comparison data register) and output from the watchdog timer to external devices.
Figure V.3.1.1 shows a block diagram of the watchdog timer.

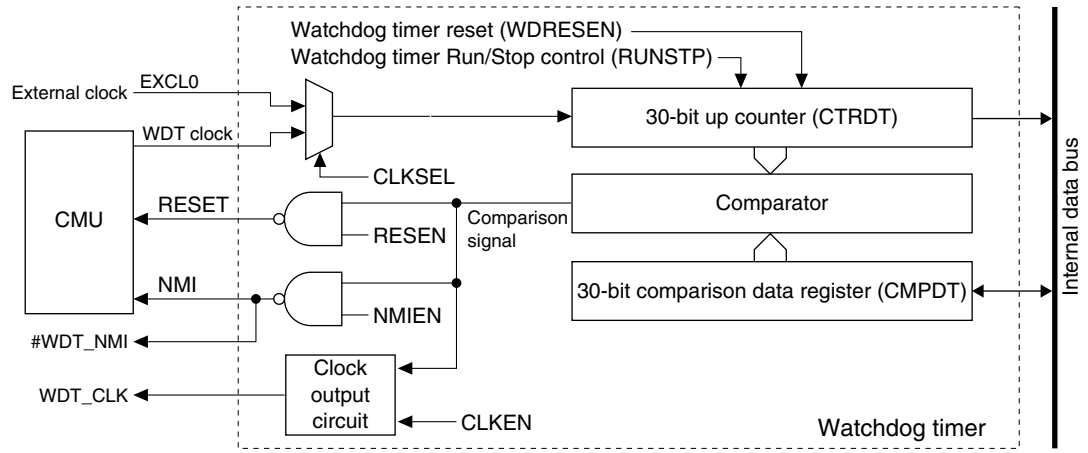


Figure V.3.1.1 Block Diagram of Watchdog Timer

V.3.2 Input/Output Pins of the Watchdog Timer

Table V.3.2.1 Input/Output Pins of Watchdog Timer

Pin name	I/O	Function
EXCL0	I	External clock input pin (external clock input for MFT)
WDT_CLK	O	Watchdog timer clock output pin
#WDT_NMI	O	Watchdog timer NMI output pin

The EXCL0 pin is used to clock the counter of the watchdog timer with an external clock.
The WDT_CLK pin is used to output the clock generated in the watchdog timer to external devices.
The #WDT_NMI pin is used to output the NMI signal generated in the watchdog timer to external devices.

Note: These pins are shared with general-purpose input/output ports or other peripheral circuit input/output pins, and set for other than the watchdog timer function by default. Therefore, before these pins can be used as input/output ports for the watchdog timer clock, the corresponding Port Function Select Register must be set to switch over the pin functions.
For details about pin functions and how to switch over, see Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

V.3.3 Operating Clock of the Watchdog Timer

The watchdog timer module is clocked by the WDT clock (= system clock) supplied from the CMU. At initial reset, this clock is selected as the operating clock for the watchdog timer. While the watchdog timer remains idle or is not being used, the clock supplied from the CMU can be turned off to reduce the amount of current consumed on the chip. Use WDT_CLK_EN (D2/CMU_GATEDCLK2 register) for this control.

* **WDT_CLK_EN**: WDT Module Clock Control Bit in the Gated Clock Control 2 (CMU_GATEDCLK2) Register (D2/0x4908)

Setting WDT_CLK_EN to 0 turns off the clock supplied from the CMU to the watchdog timer.
For details about clock generation and control, see Section II.2, "Clock Management Unit (CMU)."

Notes:
- Even when using an external clock as the count clock for the watchdog timer, the WDT clock is required for watchdog timer operation and access to its control register.
- The Gated Clock Control 2 Register (0x4908) is write-protected. To rewrite this register and other CMU control registers at addresses 0x4900 to 0x4908, write protection must be removed by writing 0x96 to the CMU Write Protect Register (0x4920). Since unnecessary rewrites to addresses 0x4900 to 0x4908 may cause the system to operate erratically, make sure that data set in the CMU Write Protect Register (0x4920) is other than 0x96 unless rewriting said registers.

V.3.4 Control of the Watchdog Timer

V.3.4.1 Setting Up the Watchdog Timer

Selecting the count clock

The internal clock or external clock (EXCL0) can be selected as the count clock for the 30-bit up-counter by using CLKSEL (D6/WD_EN register).

* **CLKSEL**: WDT Input Clock Select Bit in the WDT Enable and Setup (WD_EN) Register (D6/0x5662)

Setting CLKSEL to 0 (default) selects the internal clock; setting it to 1 selects the external clock (EXCL0). Therefore, before an external clock can be used, the function of the pin set as an I/O port by default must be switched to EXCL0 (external clock input for MFT) by using the Port Function Select Register. For details about pin functions and how to switch over, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."
For details about WDT clock supply control, see Section II.2, "Clock Management Unit (CMU)."

Setting the NMI/reset generation cycle

The watchdog timer has a 30-bit comparison data register that can be used to set a cycle in which to generate an NMI or reset signal.

* **CMPDT[15:0]**: WDT Comparison Data Bits in the WDT Comparison Data L (WD_CMP_L) Register (D[15:0]/0x5664)
* **CMPDT[29:16]**: WDT Comparison Data Bits in the WDT Comparison Data H (WD_CMP_H) Register (D[13:0]/0x5666)

The data set in these register bits is compared with the up-counter value. When both match, a specified NMI or reset signal is output. The up-counter is reset to 0 at this time.
The NMI/reset generation cycle can be calculated from the equation below.

$$\text{NMI generating cycle} = \frac{\text{CMPDT} + 1}{f_{WDTIN}} \text{ [sec]}$$

where

CMPDT = value set in CMPDT[29:0] (D[13:0]/WD_CMP_H register, D[15:0]/WD_CMP_L register)
f_{WDTIN} = Input clock frequency [Hz]

For example, the specifiable maximum NMI/reset generation cycle is about 21.47 seconds at 50-MHz clock input.

Note: Do not set a value equal to or less than 0x1f in the comparison data register.

Selecting the NMI/reset generation function

To output an NMI signal when the watchdog timer is not reset within a specified cycle, set NMIEN (D1/WD_EN register) to 1. To output a reset signal instead, set RESEN (D0/WD_EN register) to 1.

* **NMIEN**: WDT NMI Enable Bit in the WDT Enable and Setup (WD_EN) Register (D1/0x5662)
* **RESEN**: WDT RESET Enable Bit in the WDT Enable and Setup (WD_EN) Register (D0/0x5662)

Setting both bits to 0 (default) generates neither an NMI signal nor a reset signal, although the up-counter remains active and can output a clock.
Setting both bits to 1 outputs both an NMI signal and a reset signal. In this case, however, reset handling is executed since it has priority over the NMI handling.
The NMI and reset signals are both output as pulses of 32 system clocks in width.

Note: Depending on the counter and comparison register values, an NMI or reset signal may be generated after the NMI or reset function is enabled here (or even when the watchdog timer has not yet been started). Always be sure to set comparison data and reset the watchdog timer before writing 1 to NMIEN or RESEN.

Write protection of watchdog timer registers

The WDT Enable and Setup Register (0x5662) and WDT Comparison Data Registers (0x5664, 0x5666) are write-protected to prevent NMI or reset signals from being inadvertently generated by unnecessary write operations. To rewrite these registers, write protection must be removed by writing 0x96 to the WDT Write Protect Register (0x5660). Once the registers are rewritten, be sure to write other than 0x96 to the WDT Write Protect Register (0x5660) to reapply write protection.

V.3.4.2 Starting/Stopping the Watchdog Timer

Writing 1 to RUNSTP (D4/WD_EN register) starts counting by the watchdog timer; writing 0 stops the watchdog timer.

* **RUNSTP**: WDT Run/Stop Control Bit in the WDT Enable and Setup (WD_EN) Register (D4/0x5662)

Since RUNSTP exists in the write-protected WDT Enable and Setup Register, write protection must be removed by writing 0x96 to the WDT Write Protect Register (0x5660) before the content of RUNSTP can be altered.

V.3.4.3 Resetting the Watchdog Timer

Before the NMI/reset generation function of the watchdog timer can be used, a routine to reset the watchdog timer before NMI or reset generation must be prepared in a location for periodic processing. Make sure that this routine is processed within the NMI/reset generation cycle described earlier.
Writing 1 to WDRESEN (D0/WD_CTL register) resets the watchdog timer. The up-counter is reset to 0 at this time, then starts counting NMI/reset generation cycles all over again.

* **WDRESEN**: WDT Reset Bit in the WDT Control (WD_CTL) Register (D0/0x566c)

If the watchdog timer is not reset within the set cycle for some reason, the CPU is placed into trap handling by an NMI or reset signal to execute the processing routine.
The reset and NMI vector addresses are set by default to 0x20000 and 0x20008, respectively. The vector table base address can be altered by using TTBR.
The count value of the up-counter can be read out from the WDT Count Data Registers (0x5668, 0x566a) at any time.

* **CTRDT[15:0]**: WDT Counter Data Bits in the WDT Count Data L (WD_CNT_L) Register (D[15:0]/0x5668)
* **CTRDT[29:16]**: WDT Counter Data Bits in the WDT Count Data H (WD_CNT_H) Register (D[13:0]/0x566a)

V.3.4.4 Operation in Standby Mode

In HALT mode

In HALT mode, the watchdog timer remains active as it is supplied with a clock. Therefore, if HALT mode remains active beyond the NMI/reset generation cycle, an NMI or reset signal deactivates HALT mode.
To disable the watchdog timer in HALT mode, set NMIEN (D1/WD_EN register) or RESEN (D0/WD_EN register) to 0. Otherwise, write 0 to RUNSTP (D4/WD_EN register) to stop the watchdog timer before executing the halt instruction.
When NMIEN (D1/WD_EN register) or RESEN (D0/WD_EN register) disables NMI or reset generation, the watchdog timer continues counting even in HALT mode. To reenable NMI or reset generation after exiting HALT mode, be sure to reset the watchdog timer beforehand.
When HALT mode is entered after stopping the watchdog timer, be sure to reset the watchdog timer before restarting it.

In SLEEP mode

The supply of the WDT clock from the CMU stops in SLEEP mode. Therefore, the watchdog timer also stops operating. To prevent an unnecessary NMI or reset signal from being generated after exiting SLEEP mode, be sure to reset the watchdog timer before executing the slp instruction. Moreover, disable NMI/reset generation by setting NMIEN (D1/WD_EN register) or RESEN (D0/WD_EN register) as required.

V.3.4.5 Clock Output of the Watchdog Timer

The watchdog timer can output an NMI/reset generation cycle-synchronous clock from the IC to external devices. For this clock output, set CLKEN (D5/WD_EN register) to 1 after setting up the WDT_CLK pin.

* **CLKEN**: WDT Clock Output Control Bit in the WDT Enable and Setup (WD_EN) Register (D5/0x5662)

Since CLKEN also exists in the write-protected WDT Enable and Setup Register, write protection must be removed by writing 0x96 to the WDT Write Protect Register before the content of CLKEN can be altered.

If the watchdog timer is not reset in software, the level of clock output from the IC is reversed synchronously with the NMI generation cycles. (This applies when reset generation is disabled.)
When the watchdog timer is reset in software, clock output from the IC goes low at that time and remains low.

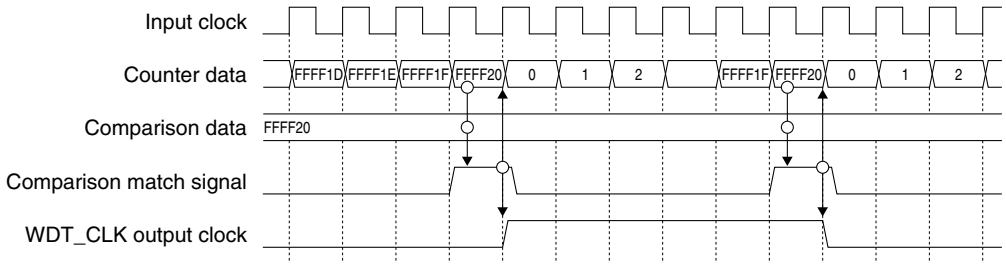


Figure V.3.4.5.1 Clock Output of Watchdog Timer

V.3.4.6 External NMI Output

The watchdog timer can output the NMI signal generated to external devices. The watchdog timer uses the #WDT_NMI pin for this output. This pin is configured as a general-purpose I/O pin at initial reset, therefore, the pin function must be set as #WDT_NMI (see Section I.3.3).

Setting NMIEN (D1/WD_EN register) to 1 enables the external NMI signal output as well as the internal NMI signal output.
When the watchdog timer counter reaches the comparison data, the #WDT_NMI pin outputs a low pulse with 32 system clock cycles.

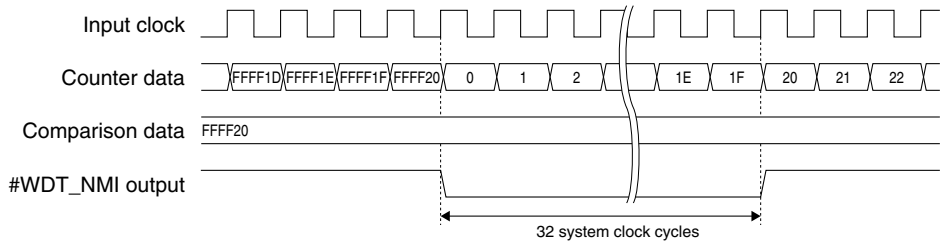


Figure V.3.4.6.1 External NMI Output

V.3.5 Details of Control Registers

Table V.3.5.1 List of WDT Control Registers

Address	Register name		Function
0x5660	WD_WP	WDT Write Protect Register	Enables WDT control registers for writing.
0x5662	WD_EN	WDT Enable and Setup Register	Configures and starts watchdog timer.
0x5664	WD_CMP_L	WDT Comparison Data L Register	Comparison data
0x5666	WD_CMP_H	WDT Comparison Data H Register	
0x5668	WD_CNT_L	WDT Count Data L Register	Watchdog timer counter data
0x566a	WD_CNT_H	WDT Count Data H Register	
0x566c	WD_CTL	WDT Control Register	Resets watchdog timer.

The following describes each WDT register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x5660: WDT Write Protect Register (WD_WP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Write Protect Register (WD_WP)	**0x5660** (16 bits)	D15–0	**WDPTC [15:0]**	WDT register write protect flag	Writing 0x96 removes the write protection of the WD_EN, WD_CMP_L, and WD_CMP_H registers (0x5662–0x5666). Writing another value set the write protection.	X	W	0 when being read.

D[15:0] WDPTC[15:0]: WDT Register Write Protect Flag Bits

These bits set or clear write protection at addresses 0x5662 to 0x5666.

0x96 (W): Clears write protection
Other than 0x96 (W): Applies write protection (default, indeterminate value)
0x0 (R): Always 0x0 when read

Before altering the WDT Enable and Setup Register (0x5662) or WDT Comparison Data Registers (0x5664, 0x5666), write 0x96 to this register to remove write protection. Setting this register to other than 0x96 will result in the contents of the registers above not being altered even when executing the write instruction without any problem. Once write protection is removed by writing 0x96 to this register, said registers can be rewritten any number of times until this register is set to other than 0x96. When the WDT Enable and Setup Register (0x5662) or WDT Comparison Data Registers (0x5664, 0x5666) have been rewritten, be sure to write other than 0x96 to this register to prevent erroneous writing to the registers.

0x5662: WDT Enable and Setup Register (WD_EN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Enable and Setup Register (WD_EN)	0x5662 (16 bits)	D15–7	–	reserved	–	–	–	0 when being read.
		D6	CLKSEL	WDT input clock select	1 External clk 0 Internal clk	0	R/W	
		D5	CLKEN	WDT clock output control	1 On 0 Off	0	R/W	
		D4	RUNSTP	WDT Run/Stop control	1 Run 0 Stop	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	NMIEN	WDT NMI enable	1 Enable 0 Disable	0	R/W	
		D0	RESEN	WDT RESET enable	1 Enable 0 Disable	0	R/W	

Note: This register is write-protected to prevent NMI or reset signals from being inadvertently generated by unnecessary write operations. To rewrite this register, write protection must be removed by writing 0x96 to the WDT Write Protect Register (0x5660). Once the register has been rewritten, be sure to write other than 0x96 to the WDT Write Protect Register (0x5660) to reapply write protection.

D[15:7] Reserved

D6 CLKSEL: WDT Input Clock Select Bit

This bit selects the count clock for the watchdog timer.
1 (R/W): External clock (EXCL0)
0 (R/W): Internal clock (default)

Setting this bit to 0 (default) selects the internal clock; setting it to 1 selects the external clock (EXCL0). Before an external clock can be used, the function of the pin set by default as an I/O port must be switched to EXCL0 (external clock input for MFT) by using the Port Function Select Register. For details about pin functions and how to switch over, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

D5 CLKEN: WDT Clock Output Control Bit

This bit controls the clock output of the watchdog timer.
1 (R/W): On
0 (R/W): Off (default)

Setting this bit to 1 outputs an NMI/reset generation cycle-synchronous clock from the IC. Before this clock output can be used, however, the function of the pin set by default as an I/O port must be switched to WDT_CLK (watchdog timer clock output) by using the Port Function Select Register. For details about pin functions and how to switch over, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

D4 RUNSTP: WDT Run/Stop Control Bit

This bit starts or stops the watchdog timer.
1 (R/W): Start
0 (R/W): Stop (default)

When the NMI or reset generation function is enabled, be sure to set comparison data and reset the watchdog timer before starting the watchdog timer, thus preventing the generation of unnecessary NMI or reset signals.

D[3:2] Reserved

D1 NMIEN: WDT NMI Enable Bit

This bit enables NMI signal output by the watchdog timer.
1 (R/W): Enable
0 (R/W): Disable (default)

Setting this bit to 1 outputs an NMI signal (a pulse 32 system clocks in width) to the CMU and the #WDT_NMI pin when the count of the up-counter matches the value set in the comparison data register. Setting this bit to 0 outputs no NMI signals.
Regardless of how this bit is set, the up-counter is reset to 0 when the up-counter and set value of the comparison data register match, then starts counting all over again.

D0 RESEN: WDT RESET Enable Bit

This bit enables internal reset signal output by the watchdog timer.
1 (R/W): Enable
0 (R/W): Disable (default)

Setting this bit to 1 outputs a reset signal (a pulse 32 system clocks in width) to the CMU when the count of the up-counter matches the value set in the comparison data register. Setting this bit to 0 outputs no reset signals.

0x5664: WDT Comparison Data L Register (WD_CMP_L)
0x5666: WDT Comparison Data H Register (WD_CMP_H)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Comparison Data L Register (WD_CMP_L)	**0x5664** (16 bits)	D15–0	**CMPDT [15:0]**	WDT comparison data CMPDT0 = LSB	0x0 to 0x3fffffff (low-order 16 bits)	0x0	R/W	
WDT Comparison Data H Register (WD_CMP_H)	**0x5666** (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13-0	**CMPDT [29:16]**	WDT comparison data CMPDT29 = MSB	0x0 to 0x3fffffff (high-order 14 bits)	0x0	R/W	

Note: These registers are write-protected to prevent NMI or reset signals from being inadvertently generated by unnecessary write operations. To rewrite these registers, write protection must be removed by writing 0x96 to the WDT Write Protect Register (0x5660). Once the registers have been rewritten, be sure to write other than 0x96 to the WDT Write Protect Register (0x5660) to reapply write protection.

Use these registers to set the NMI/reset generation cycle.
With NMI or reset generation enabled, an NMI or reset signal is output when the up-counter matches the comparison data set in these registers.
When a clock is output from the watchdog timer, these registers also set the output clock cycle.

D[15:0]/0x5664 CMPDT[15:0]: WDT Comparison Data Bits (16 low-order bits)

The 16 low-order bits of comparison data are set in these bits. (Default: 0x0)

D[13:0]/0x5666 CMPDT[29:16]: WDT Comparison Data Bits (14 high-order bits)

The 14 high-order bits of comparison data are set in these bits. (Default: 0x0)

Note: Do not set a value equal to or less than 0x1f as comparison data.

0x5668: WDT Count Data L Register (WD_CNT_L)
0x566a: WDT Count Data H Register (WD_CNT_H)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Count Data L Register (WD_CNT_L)	**0x5668** (16 bits)	D15–0	**CTRDT [15:0]**	WDT counter data CTRDT0 = LSB	0x0 to 0x3fffffff (low-order 16 bits)	X	R	
WDT Count Data H Register (WD_CNT_H)	**0x566a** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13-0	**CTRDT [29:16]**	WDT counter data CTRDT29 = MSB	0x0 to 0x3fffffff (high-order 14 bits)	X	R	

The current count value of the up-counter can be read out from these registers.

D[15:0]/0x5668 CTRDT[15:0]: WDT Counter Data Bits (16 low-order bits)

The 16 low-order bits of the 30-bit up-counter are read out from these bits. (Default: indeterminate)

D[13:0]/0x566a CTRDT[29:16]: WDT Counter Data Bits (14 high-order bits)

The 14 high-order bits of the 30-bit up-counter are read out from these bits. (Default: indeterminate)

0x566c: WDT Control Register (WD_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Control Register (WD_CTL)	0x566c (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	WDRESEN	WDT reset	1 Reset 0 ignored	0	W	

D[15:1] Reserved

D0 WDRESEN: WDT Reset Bit

This bit resets the watchdog timer.

1 (W): Reset

0 (W): Has no effect

0 (R): Always 0 when read (default)

With NMI or reset signal output enabled, the watchdog timer must be reset by writing 1 to this bit within the set NMI/reset generation cycle. The up-counter is thereby reset to 0, then starts counting NMI/reset generation cycles all over again.

V.3.6 Precautions

- When NMI or reset signal output by the watchdog timer is enabled, the watchdog timer must be reset within the set NMI/reset generation cycle.
- Do not set a value equal to or less than 0x1f in the comparison data register.
- Depending on the counter and comparison register values, an NMI or reset signal may be generated after the NMI or reset function is enabled, or immediately after the watchdog timer starts. Always be sure to set comparison data and reset the watchdog timer before writing 1 to NMIEN (D1/WD_EN register), RESEN (D0/WD_EN register), or RUNSTP (D4/WD_EN register).
    - * **NMIEN**: WDT NMI Enable Bit in the WDT Enable and Setup (WD_EN) Register (D1/0x5662)
    - * **RESEN**: WDT RESET Enable Bit in the WDT Enable and Setup (WD_EN) Register (D0/0x5662)
    - * **RUNSTP**: WDT Run/Stop Control Bit in the WDT Enable and Setup (WD_EN) Register (D4/0x5662)

VI S1C17501 INTERFACE MODULES

VI.1 UART

VI.1.1 Outline of the UART

The S1C17501 equipped with one channel of UART. The UART performs asynchronous data transfer from/to an external serial device in a 150 to 460800 bps (max. 115200 bps in IrDA mode) transfer rate. The UART contains two-byte receive data buffer and one-byte transmit data buffer allowing full-duplex communication. The transfer clock is internally generated using a timer module or an external clock is input from the #SCLK0 pin. The character length (seven or eight bits), number of stop bits (one or two bits), and parity mode (even, odd, or none) are programmable. The start bit is fixed at one bit. In data receive operation, overrun, framing, and parity errors are detectable. The UART can generate three types of interrupts (transmit buffer empty, receive buffer full, and receive error), this makes it possible to process serial data transfer simply in an interrupt handler.
Furthermore, the UART module contains an RZI modulator/demodulator, allowing an infrared-ray communication circuit to be configured based on IrDA 1.0 simply by adding an external circuit.
Figure VI.1.1.1 shows the structure of the UART.

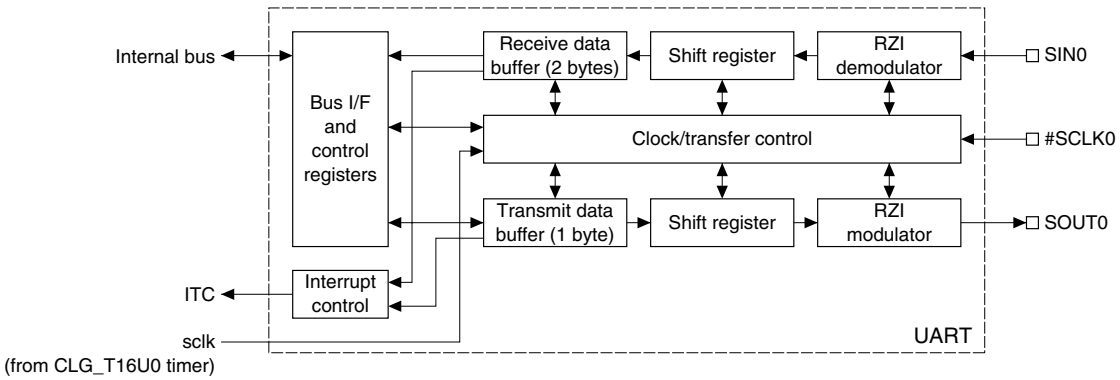


Figure VI.1.1.1 Structure of UART

VI.1.2 UART Pins

Table VI.1.2.1 lists the I/O pins for the UART.

Table VI.1.2.1 List of UART Pins

Pin name	I/O	Size	Function
SIN0	I	1	UART data input pin This pin inputs serial data sent from an external serial device.
SOUT0	O	1	UART data output pin This pin outputs serial data to be sent to an external serial device.
#SCLK0	I	1	UART clock input pin This pin inputs the transfer clock when an external clock is used.

The UART input/output pins (SIN0, SOUT0, #SCLK0) are shared with the I/O ports (P40, P41, P42) and they are initialized as general-purpose I/O port pins by default. Before using these pins for the UART, the pin functions must be switched using the Port Function Select Register.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.1.3 Transfer Clock

The UART allows the application to select either the internal clock or an external clock as the transfer clock. Use the SSCK bit (D0/UART_MOD register) for this selection.

* **SSCK**: Input Clock Select Bit in the UART Mode (UART_MOD) Register (D0/0x4103)

Note: Make sure that the UART is disabled (RXEN/UART_CTL register = 0) when alter the SSCK bit.

* **RXEN**: UART Enable Bit in the UART Control (UART_CTL) Register (D0/0x4104)

Internal clock

When SSCK is set to 0 (default), the internal clock is selected. The UART uses the CLG_T16U0 timer output clock as the transfer clock. Therefore, it is necessary to program the CLG_T16U0 timer so that it will output a clock according to the transfer rate.
See Section II.4, "Clock Generator (CLG)," for controlling the CLG_T16U0 timer.

External clock

When SSCK is set to 1, an external clock is selected. Configure the #SCLK0 pin and input an external clock to the pin.

Notes:
- The UART divides the CLG_T16U0 timer output clock or external clock by 16 to generate the sampling clock. Make sure of the division ratio when setting a transfer rate.
- The frequency of the external clock input from the #SCLK0 pin must be half of PCLK or lower and the clock duty ratio must be 50%.

VI.1.4 Setting Transfer Data Conditions

The following conditions are selectable to configure transfer data format:
- Character length: 7 or 8 bits
- Start bit: 1 bit, fixed
- Stop bit: 1 or 2 bits
- Parity bit: Even, odd, or none

Note: Make sure that the UART is disabled (RXEN/UART_CTL register = 0) when setting the transfer data conditions.

∗ **RXEN**: UART Enable Bit in the UART Control (UART_CTL) Register (D0/0x4104)

Character length

Use the CHLN bit (D4/UART_MOD register) to select the character length. When CHLN is set to 0 (default), the character length is configured to seven bits; when CHLN is set to 1, the character length is configured to eight bits.

∗ **CHLN**: Character Length Bit in the UART Mode (UART_MOD) Register (D4/0x4103)

Stop bit

Use the STPB bit (D1/UART_MOD register) to select the stop bit length. When STPB is set to 0 (default), the stop bit length is set to one bit; when STPB is set to 1, the stop bit length is set to two bits.

∗ **STPB**: Stop Bit Select Bit in the UART Mode (UART_MOD) Register (D1/0x4103)

Parity bit

Use the PREN bit (D3/UART_MOD register) to select whether the parity function is enabled or not. When PREN is set to 0 (default), parity function is disabled. In this case, a parity bit will not be added to transfer data and the parity check will not be performed when data is received. When PREN is set to 1, parity function is enabled. In this case, a parity bit will be added to transfer data and the parity check will be performed when data is received.
When the parity function is enabled, select a parity mode using the PMD bit (D2/UART_MOD register). When PMD is set to 0 (default), the parity bit is added/checked as even parity; when PMD is set to 1, the parity bit is added/checked as odd parity.

∗ **PREN**: Parity Enable Bit in the UART Mode (UART_MOD) Register (D3/0x4103)
∗ **PMD**: Parity Mode Select Bit in the UART Mode (UART_MOD) Register (D2/0x4103)

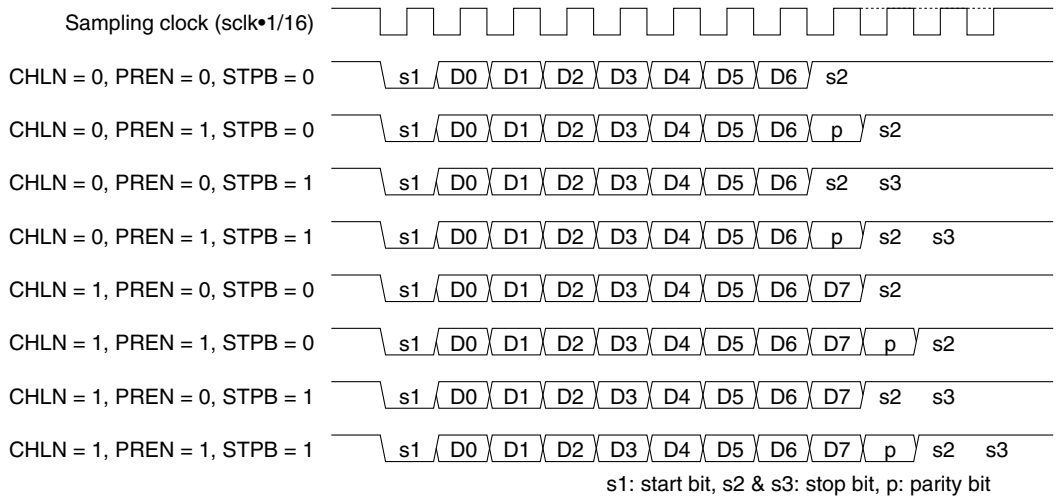


Figure VI.1.4.1 Transfer Data Format

VI.1.5 Data Transmit/Receive Control

Before starting data transfer, set up the conditions as shown below.

(1) Select an input clock. See Section VI.1.3.
    Set up the CLG_T16U0 timer to output the transfer clock if the internal clock is used as the transfer clock. See Section II.4.
(2) Configure the transfer data format. See Section VI.1.4.
(3) Set IrDA mode when using the IrDA interface. See Section VI.1.8.
(4) Set up the interrupt conditions if the UART interrupt is used. See Section VI.1.7.

Note: Make sure that the UART is disabled (RXEN/UART_CTL register = 0) when setting the conditions above.

∗ **RXEN**: UART Enable Bit in the UART Control (UART_CTL) Register (D0/0x4104)

Enabling data transmission/reception

First, set the RXEN bit (D0/UART_CTL register) to 1 to enable data transmission/reception. This puts the transmitter/receiver in ready-to-transmit/receive status.

Note: Do not set the RXEN bit to 0 while the UART is transmitting/receiving data.

Data transmit control

To start transmission, write transmit data to the UART_TXD register (0x4101).

∗ **UART_TXD**: UART Transmit Data Register (0x4101)

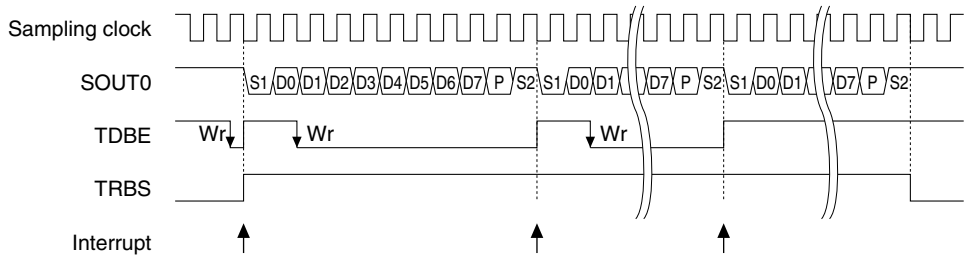
Data is written to the transmit data buffer and the transmitter starts data transmission.
The buffered data is sent to the shift register for transmission and a start bit is output from the SOUT0 pin. Then data in the shift register is output from the LSB. The transmit data bits are shifted in sync with the rising edge of the sampling clock and output from the SOUT0 pin sequentially. After the MSB has been output, a parity bit (if parity is enabled) and a stop bit are output.

The transmitter provides two status flags, TDBE (D0/UART_ST register) and TRBS (D2/UART_ST register).

∗ **TDBE**: Transmit Data Buffer Empty Flag Bit in the UART Status (UART_ST) Register (D0/0x4100)
∗ **TRBS**: Transmit Busy Flag Bit in the UART Status (UART_ST) Register (D2/0x4100)

The TDBE flag indicates the transmit data buffer status; it goes 0 when the application program writes data to the transmit data buffer and returns to 1 when the data in the transmit data buffer is sent to the shift register for transmitting. An interrupt can be generated when this flag goes 1 (see Section VI.1.7). Use this interrupt or read the TDBE flag to check that the transmit data buffer is empty before transmitting the next data. Although the transmit data buffer size is one byte, transmit data can be written while the previous data is being transmitted as the shift register is separately provided. However, make sure that the transmit data buffer is empty before writing transmit data. If data is written when the TDBE flag is 0, the previous transmit data in the transmit data buffer is overwritten with the new data.

The TRBS flag indicates the shift register status; it goes 1 when transmit data is loaded from the transmit data buffer and returns to 0 upon completion of a data transmission. Read this flag to check whether the transmitter is busy or idle.



S1: Start bit, S2: Stop bit, P: Parity bit, Wr: Data write to transmit data buffer

Figure VI.1.5.1 Data Transmit Timing Chart

Data receive control

The receiver activates by setting the RXEN bit to 1 and is ready to receive data sent from an external serial device.
When an external serial device has sent a start bit, the receiver detects its low level and starts following data bit sampling. The data bits are sampled at the rising edge of the sampling clock and received in the receive shift register assuming that the first data bit is LSB. After the MSB is received in the shift register, the received data is loaded to the receive data buffer. At the same time, the receiver performs a parity check with the parity bit received after the MSB if parity check is enabled.
The receive data buffer is a two-byte FIFO and can receive data until it becomes full.

The received data in the buffer can be read from the UART_RXD register (0x4102). The older data is read out first and cleared by reading.

* **UART_RXD**: UART Receive Data Register (0x4102)

The receiver provides two buffer status flags, RDRY (D1/UART_ST register) and RD2B (D3/UART_ST register).

* **RDRY**: Receive Data Ready Flag Bit in the UART Status (UART_ST) Register (D1/0x4100)
* **RD2B**: Second Byte Receive Flag Bit in the UART Status (UART_ST) Register (D3/0x4100)

The RDRY flag indicates that the receive data buffer contains the received data. The RD2B flag indicates that the receive data buffer is full.

(1) RDRY = 0, RD2B = 0
No data has been received. Therefore, it is not necessary to read the receive data buffer.

(2) RDRY = 1, RD2B = 0
One data has been received. Read the receive data buffer once. This reading resets the RDRY flag. The buffer status returns to (1) above.
If the receive data buffer is read twice, the second read value is invalid data.

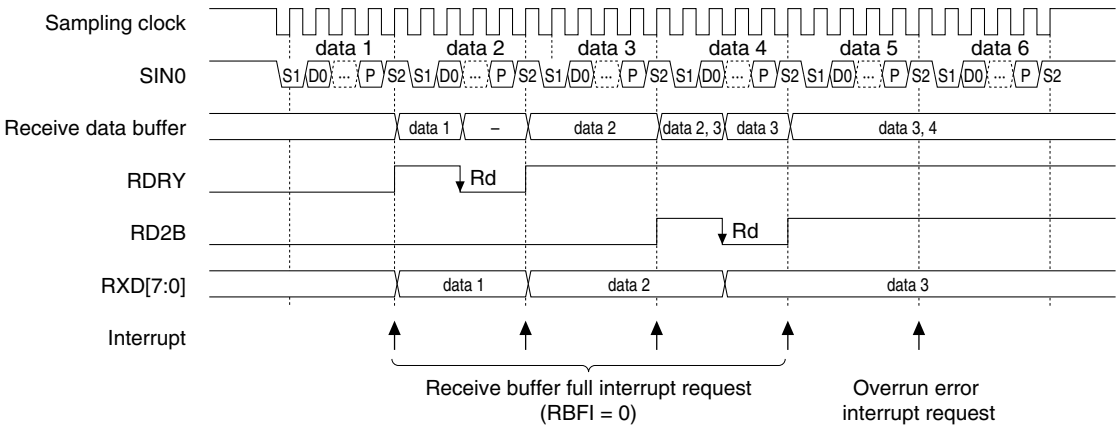
(3) RDRY = 1, RD2B = 1
Two data have been received. Read the receive data buffer twice. The receive data buffer outputs the older received data in the first reading. This reading resets the RD2B flag. The buffer status goes to (2) above. The latest received data is output in the second reading. The buffer status returns to (1) above after reading twice.
Even when the receive data buffer is full, the shift register can start receiving one more 8-bit data. An overrun error will occur if receiving is finished before the receive data buffer has been read. In this case, the last received data cannot be read. Therefore, be sure to read the receive data buffer before an overrun error occurs. Refer to Section VI.1.6 for the overrun error.

By reading these flags, the application program can check how many data have been received.

Furthermore, the UART can generate a receive data buffer full interrupt when data is received in the receive data buffer. This interrupt can be used to read the received data. A receive data buffer full interrupt occurs when one data has been received in the receive data buffer (status (2) above) by default. This may be changed by setting the RBFI bit (D1/UART_CTL register) to 1 so that the interrupt will occur when two data have been received in the received data buffer.

* **RBFI**: Receive Buffer Full Interrupt Condition Bit in the UART Control (UART_CTL) Register (D1/0x4104)

In addition to the flags above, three receive error flags are provided. Refer to Section VI.1.6 for these flags and details of receive errors.



S1: Start bit, S2: Stop bit, P: Parity bit, Rd: Data read from RXD[7:0]

Figure VI.1.5.2 Data Receive Timing Chart

Disabling data transmission/reception

After data transfer (both transmission and reception) has finished, write 0 to the RXEN bit to disable data transmission/reception.

Always make sure that the TDBE flag is 1 and TRBS and RDRY flags are 0 before data transmission/reception is disabled.

When the RXEN bit is set to 0, the transmit data buffer is placed in empty status (data is cleared if any remains). Furthermore, the data being transferred cannot be guaranteed if RXEN is set to 0 during transmitting/receiving.

VI.1.6 Receive Errors

Three types of receive errors can be detected in data reception.
The receive errors are causes of interrupt, so the error can be processed in the interrupt handler routine. Refer to Section VI.1.7 for controlling the UART interrupts.

Parity error

If the PREN bit (D3/UART_MOD register) is set to 1 (parity enabled), the parity bit is checked when data is received.
This parity check is performed when the data received in the shift register is loaded to the receive data buffer in order to check conformity with the PMD bit (D2/UART_MOD register) setting (odd or even parity).
If any nonconformity is found in this check, a parity error is assumed and the parity error flag PER (D5/UART_ST register) is set to 1.
Even when this error occurs, the received data in error is loaded to the receive data buffer and the receive operation is continued. However, the received data in which a parity error has occurred cannot be guaranteed.
The PER flag (D5/UART_ST register) is reset to 0 by writing 1.

- **PREN**: Parity Enable Bit in the UART Mode (UART_MOD) Register (D3/0x4103)
- **PMD**: Parity Mode Select Bit in the UART Mode (UART_MOD) Register (D2/0x4103)
- **PER**: Parity Error Flag Bit in the UART Status (UART_ST) Register (D5/0x4100)

Framing error

If data with a stop bit = 0 is received, the UART assumes that the data is out of sync and generates a framing error.
If two stop bits are used, only the first stop bit is checked.
When this error occurs, the framing-error flag FER (D6/UART_ST register) is set to 1.
Even when this error occurs, the received data in error is loaded to the receive data buffer and the receive operation is continued. However, the received data in which a framing error has occurred cannot be guaranteed, even if no framing error is found in the following data received.
The FER flag (D6/UART_ST register) is reset to 0 by writing 1.

- **FER**: Framing Error Flag Bit in the UART Status (UART_ST) Register (D6/0x4100)

Overrun error

Even when the receive data buffer is full (two data have been received), the next (third) data can be received into the shift register. However, if the receive data buffer is not emptied (by reading out data received) by the time this data has been received, the third data received in the shift register will not be sent to the buffer and generate an overrun error.
When an overrun error occurs, the overrun error flag OER (D4/UART_ST register) is set to 1.
Even when this error occurs, the receive operation is continued.
The OER flag (D4/UART_ST register) is reset to 0 by writing 1.

- **OER**: Overrun Error Flag Bit in the UART Status (UART_ST) Register (D4/0x4100)

VI.1.7 UART Interrupt

The UART can generate the following three types of interrupts:
• Transmit buffer empty interrupt
• Receive buffer full interrupt
• Receive error interrupt

The UART has one interrupt signal to be output to the interrupt controller (ITC) and it is shared with all three causes of interrupt. To determine the cause of interrupt that has occurred, read the status and error flags.

Transmit buffer empty interrupt

Set the TIEN bit (D4/UART_CTL register) to 1 when using this interrupt. If TIEN is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **TIEN**: Transmit Buffer Empty Interrupt Enable Bit in the UART Control (UART_CTL) Register (D4/0x4104)

When the transmit data set in the transmit data buffer is transferred to the shift register, the UART sets the TDBE bit (D0/UART_ST register) to 1 to indicate that the transmit data buffer is empty. At the same time, the UART outputs an interrupt request pulse to the ITC if the transmit buffer empty interrupt has been enabled (TIEN = 1).

∗ **TDBE**: Transmit Data Buffer Empty Flag Bit in the UART Status (UART_ST) Register (D0/0x4100)

If other interrupt conditions are satisfied, an interrupt is generated.
The UART interrupt handler routine should read the TDBE flag to check if the interrupt has occurred due to a transmit buffer empty or another cause. When TDBE = 1, the UART interrupt handler routine can write the next transmit data to the transmit data buffer.

Receive buffer full interrupt

Set the RIEN bit (D5/UART_CTL register) to 1 when using this interrupt. If RIEN is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **RIEN**: Receive Buffer Full Interrupt Enable Bit in the UART Control (UART_CTL) Register (D5/0x4104)

When the specified number of received data is loaded to the receive data buffer, the UART outputs an interrupt request pulse to the ITC if the receive buffer full interrupt has been enabled (RIEN = 1). If the RBFI bit (D1/UART_CTL register) is 0, an interrupt request pulse is output when received data is loaded to the receive data buffer (when the RDRY flag (D1/UART_ST register) goes 1). If the RBFI bit (D1/UART_CTL register) is 1, an interrupt request pulse is output when two received data occupy the receive data buffer (when the RD2B flag (D3/UART_ST register) goes 1).

∗ **RBFI**: Receive Buffer Full Interrupt Condition Bit in the UART Control (UART_CTL) Register (D1/0x4104)
∗ **RDRY**: Receive Data Ready Flag Bit in the UART Status (UART_ST) Register (D1/0x4100)
∗ **RD2B**: Second Byte Receive Flag Bit in the UART Status (UART_ST) Register (D3/0x4100)

If other interrupt conditions are satisfied, an interrupt is generated.
The UART interrupt handler routine should read the RDRY and RD2B flags to check if the interrupt has occurred due to a receive buffer full or another cause. When RDRY or RD2B = 1, the UART interrupt handler routine can read the received data from the receive data buffer.

Receive error interrupt

Set the REIEN bit (D6/UART_CTL register) to 1 when using this interrupt. If REIEN is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

* **REIEN**: Receive Error Interrupt Enable Bit in the UART Control (UART_CTL) Register (D6/0x4104)

When a parity, framing, or overrun error is detected during data reception, the UART sets the error flag listed below to 1 and outputs an interrupt request pulse to the ITC if the receive error interrupt has been enabled (REIEN = 1).

* **PER**: Parity Error Flag Bit in the UART Status (UART_ST) Register (D5/0x4100)
* **FER**: Framing Error Flag Bit in the UART Status (UART_ST) Register (D6/0x4100)
* **OER**: Overrun Error Flag Bit in the UART Status (UART_ST) Register (D4/0x4100)

If other interrupt conditions are satisfied, an interrupt is generated.
The UART interrupt handler routine should read the error flags to check if the interrupt has occurred due to a receive error or another cause. When an error flag has been set to 1, the UART interrupt handler routine should execute an error recovery process.

ITC registers for UART interrupts

The following shows the control bits of the ITC provided for the UART:

Interrupt flag

* **IIFT4**: UART Interrupt Flag Bit in the Interrupt Flag (ITC_IFLG) Register (D12/0x4300)

Interrupt enable bits

* **IIEN4**: UART Interrupt Enable Bit in the Interrupt Enable (ITC_EN) Register (D12/0x4302)

Interrupt level setup bits

* **IILV4[2:0]**: UART Interrupt Level Bits in the Internal Interrupt Level Setup (ITC_ILV2) Register 2 (D[2:0]/0x4312)

When the UART outputs an interrupt request pulse, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the UART interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the UART interrupt request pulse, regardless of how the interrupt enable register is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the UART interrupt.
An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The UART interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vector

The following shows the vector number and vector address for the UART interrupt:

Vector number: 16 (0x10)
Vector address: TTBR + 0x40

VI.1.8 IrDA Interface

The UART module contains an RZI modulator/demodulator, allowing an infrared-ray communication circuit to be configured based on IrDA 1.0 simply by adding an external circuit.

The transmit data output from the shift register of the UART is input to the modulator to convert the low pulse width into 3/16 sclk cycles before it is output from the SOUT0 pin.

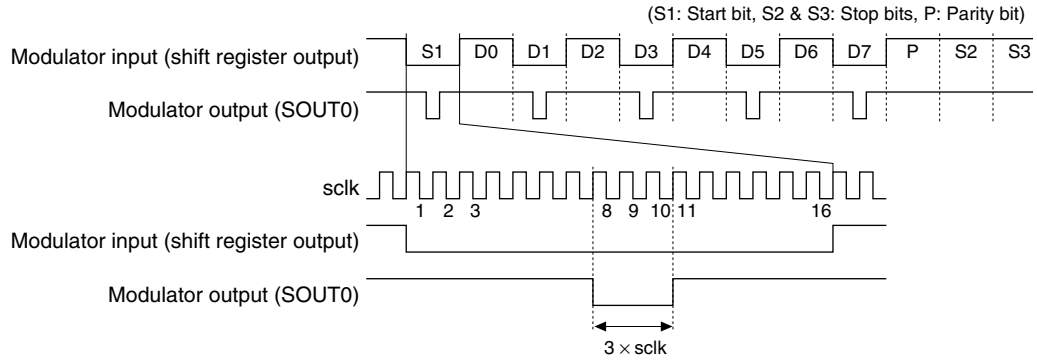


Figure VI.1.8.1 Transmit Signal Waveform

The received IrDA signal is input to the demodulator to convert the low pulse width into 16 sclk cycles before input to the shift register for receiving. To detect low pulses input to the demodulator (minimum pulse width = 1.41 µs at 115200 bps), the demodulator uses a pulse detection clock selected from the prescaler output clocks separately with the transfer clock sclk.

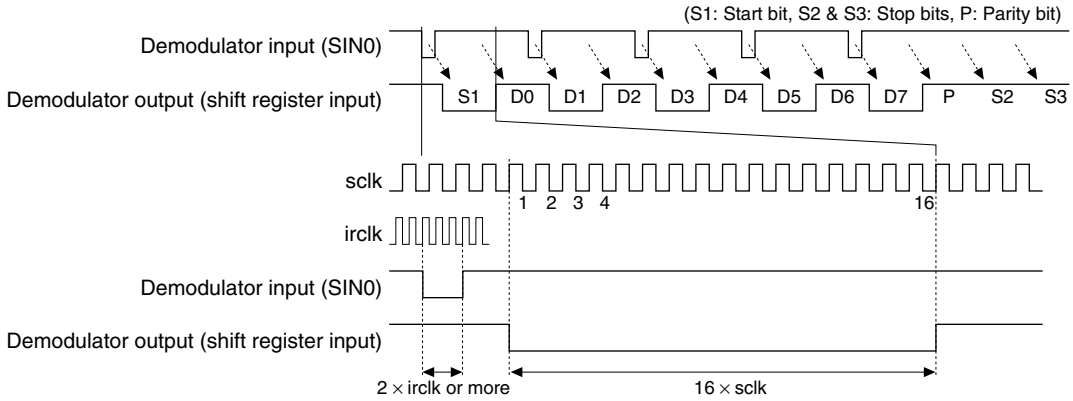


Figure VI.1.8.2 Receive Signal Waveform

Enabling the IrDA mode

To use the IrDA interface function, set the IRMD bit (D0/UART_EXP register) to 1. This enables the RZI modulator/demodulator.

∗ **IRMD**: IrDA Mode Select Bit in the UART Expansion (UART_EXP) Register (D0/0x4105)

Note: This setting must be performed before setting other UART conditions.

Selecting the IrDA receive detection clock

Select a prescaler output clock within the range from PCLK•1/1 to PCLK•1/128 as the input pulse detection clock using the IRCLK[2:0] bits (D[6:4]/UART_EXP register).

* **IRCLK[2:0]**: IrDA Receive Detection Clock Select Bits in the UART Expansion (UART_EXP) Register (D[6:4]/0x4105)

Table VI.1.8.1 Selecting the IrDA Receive Detection Clock

IRCLK[2:0]	Prescaler output clock
0x7	PCLK•1/128
0x6	PCLK•1/64
0x5	PCLK•1/32
0x4	PCLK•1/16
0x3	PCLK•1/8
0x2	PCLK•1/4
0x1	PCLK•1/2
0x0	PCLK•1/1

(Default: 0x0)

This clock must be faster than the transfer clock sclk supplied from the CLG_T16U0 timer or input from the #SCLK0 pin.

The demodulator regards a low pulse of which the width is longer than two cycles of the IrDA receive detection clock as a valid low pulse and converts it to a 16 sclk cycles width of low pulse. Select an appropriate prescaler output clock that can detect a minimum 1.41 μs width of an input pulse.

Controlling serial data transfer

The control method to transmit/receive data in IrDA mode is the same as that of the normal interface. See previous sections for details on how to set and control the data formats, data transfers, and interrupts.

VI.1.9 Details of Control Registers

Table VI.1.9.1 List of UART Registers

Address	Register name		Function
0x4100	UART_ST	UART Status Register	Indicates transfer, buffer and error statuses.
0x4101	UART_TXD	UART Transmit Data Register	Transmit data
0x4102	UART_RXD	UART Receive Data Register	Receive data
0x4103	UART_MOD	UART Mode Register	Sets transfer data format.
0x4104	UART_CTL	UART Control Register	Controls data transfer.
0x4105	UART_EXP	UART Expansion Register	Sets IrDA mode.

The following describes each UART register. These are all 8-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x4100: UART Status Register (UART_ST)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
UART Status Register (UART_ST)	0x4100 (8 bits)	D7	–	reserved	–				–	–	0 when being read.
		D6	FER	Framing error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D5	PER	Parity error flag	1	Error	0	Normal	0	R/W	
		D4	OER	Overrun error flag	1	Error	0	Normal	0	R/W	
		D3	RD2B	Second byte receive flag	1	Ready	0	Empty	0	R	
		D2	TRBS	Transmit busy flag	1	Busy	0	Idle	0	R	Shift register status
		D1	RDRY	Receive data ready flag	1	Ready	0	Empty	0	R	
		D0	TDBE	Transmit data buffer empty flag	1	Empty	0	Not empty	1	R	

D7 Reserved

D6 FER: Framing Error Flag Bit

Indicates whether a framing error has occurred or not.

1 (R): An error has occurred
0 (R): No error has occurred (default)
1 (W): Reset to 0
0 (W): Has no effect

When a framing error has occurred, FER is set to 1. A framing error occurs when data with a stop bit = 0 is received.
FER is reset by writing 1.

D5 PER: Parity Error Flag Bit

Indicates whether a parity error has occurred or not.

1 (R): An error has occurred
0 (R): No error has occurred (default)
1 (W): Reset to 0
0 (W): Has no effect

When a parity error has occurred, PER is set to 1. The parity check function is effective only when PREN (D3/UART_MOD register) is set to 1. This check is performed when the received data is transferred from the shift register to the receive data buffer.
PER is reset by writing 1.

D4 OER: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): An error has occurred
0 (R): No error has occurred (default)
1 (W): Reset to 0
0 (W): Has no effect

When an overrun error has occurred, OER is set to 1. An overrun error will occur if new data is received when the receive data buffer is full and also if the shift register contains received data. When this error occurs, the shift register is overwritten with the new received data. The receive data in the buffer is left unchanged.
OER is reset by writing 1.

D3 RD2B: Second Byte Receive Flag Bit

Indicates that the receive data buffer contains two received data.

1 (R): Second byte is ready to read out
0 (R): Second entry is empty (default)

RD2B is set to 1 when the second data is loaded to the receive data buffer, and is reset to 0 when the first data is read out from the receive data buffer.

D2 TRBS: Transmit Busy Flag Bit

Indicates the transmit shift register status.

1 (R): Busy

0 (R): Idle (default)

TRBS goes 1 when transmit data is loaded to the shift register from the transmit data buffer and returns to 0 upon completion of a data transmission. Read this flag to check whether the transmitter is busy or idle.

D1 RDRY: Receive Data Ready Flag Bit

Indicates that the receive data buffer contains valid received data.

1 (R): Data is ready to read out

0 (R): Buffer is empty (default)

RDRY is set to 1 when received data is loaded to the receive data buffer, and is reset to 0 when all data are read out from the receive data buffer.

D0 TDBE: Transmit Data Buffer Empty Flag Bit

Indicates the status of the transmit data buffer.

1 (R): Empty (default)

0 (R): Not empty

TDBE is reset to 0 when transmit data is written to the transmit data buffer and set to 1 when the transmit data in the buffer is transferred to the shift register.

0x4101: UART Transmit Data Register (UART_TXD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Transmit Data Register (UART_TXD)	**0x4101** (8 bits)	D7–0	**TXD[7:0]**	Transmit data TXD7(6) = MSB TXD0 = LSB	0x0 to 0xff (0x7f)	0x0	R/W	

D[7:0] TXD[7:0]: Transmit Data Bits

Write transmit data to be set to the transmit data buffer. (Default: 0x0)

When data is written to this register, the UART starts transmitting. The data written to TXD[7:0] enters the transmit data buffer and waits for transmission. When the data in the transmit data buffer is transferred, a cause of transmit buffer empty interrupt occurs.

In 7-bit mode, TXD7 (MSB) is ignored.

The serial-converted data is output from the SOUT0 pin beginning with the LSB, in which the bits set to 1 are output as high-level signals and those set to 0 output as low-level signals.

This register can be read as well as written.

0x4102: UART Receive Data Register (UART_RXD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Receive Data Register (UART_RXD)	**0x4102** (8 bits)	D7–0	**RXD[7:0]**	Receive data in the receive data buffer RXD7(6) = MSB RXD0 = LSB	0x0 to 0xff (0x7f)	0x0	R	Older data in the buffer is read out first.

D[7:0] RXD[7:0]: Receive Data in the Receive Data Buffer Bits

The data in the receive data buffer can be read from this register beginning with the older data first. The received data enters the receive data buffer. The receive data buffer is a two-byte FIFO and can receive data until it becomes full. When the buffer is full and also if the shift register contains received data, an overrun error will occur if the received data is not read by the time the next data receiving begins. The receive data buffer status flags RDRY (D1/UART_ST register) and RD2B (D3/UART_ST register) are provided to indicate that the receive data buffer contains valid received data and the second data, respectively.

When the receive data buffer has received the number of data specified with RBFI (D1/UART_CTL register), a cause of receive buffer full interrupt occurs.

In 7-bit mode, 0 is stored in RXD7.

The serial data input from the SIN0 pin is converted into parallel data beginning with the LSB, with the high-level signals changed to 1s and the low-level signals changed to 0s. The resulting data is stored in the receive data buffer.

This register is a read-only register, so no data can be written to it. (Default: 0x0)

0x4103: UART Mode Register (UART_MOD)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
UART Mode Register (UART_MOD)	0x4103 (8 bits)	D7–5	–	reserved	–				–	–	0 when being read.
		D4	CHLN	Character length	1	8 bits	0	7 bits	0	R/W	
		D3	PREN	Parity enable	1	With parity	0	No parity	0	R/W	
		D2	PMD	Parity mode select	1	Odd	0	Even	0	R/W	
		D1	STPB	Stop bit select	1	2 bits	0	1 bit	0	R/W	
		D0	SSCK	Input clock select	1	External	0	Internal	0	R/W	

D[7:5] Reserved

D4 CHLN: Character Length Bit

Selects the character length of serial transfer data.
1 (R/W): 8 bits
0 (R/W): 7 bits (default)

D3 PREN: Parity Enable Bit

Enables the parity function.
1 (R/W): With parity
0 (R/W): No parity (default)

PREN is used to select whether the parity check for receive data will be performed or not, and whether a parity bit will be added to transmit data. When PREN is set to 1, the received data is checked for parity. A parity bit is automatically added to the transmit data. When PREN is set to 0, parity is not checked and no parity bit is added.

D2 PMD: Parity Mode Select Bit

Selects the parity mode.
1 (R/W): Odd parity
0 (R/W): Even parity (default)

Odd parity is selected by writing 1 to PMD, and even parity is selected by writing 0. Parity check and the addition of a parity bit are effective only when PREN (D3) is set to 1. If PREN (D3) = 0, settings of PMD do not have any effect.

D1 STPB: Stop Bit Select Bit

Selects a stop bit length.
1 (R/W): 2 bits
0 (R/W): 1 bit (default)

Two stop bits are selected by writing 1 to STPB, and one stop bit is selected by writing 0. The start bit is fixed at 1 bit.

D0 SSCK: Input Clock Select Bit

Selects the clock source.
1 (R/W): External clock (#SCLK0 pin)
0 (R/W): Internal clock (default)

This bit is used to select the clock source between the internal clock (CLG_T16U0 timer output clock) and an external clock (input from the #SCLK0 pin). An external clock is selected by writing 1 to this bit, and an internal clock is selected by writing 0.

0x4104: UART Control Register (UART_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Control Register (UART_CTL)	0x4104 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	REIEN	Receive error int. enable	1 Enable 0 Disable	0	R/W	
		D5	RIEN	Receive buffer full int. enable	1 Enable 0 Disable	0	R/W	
		D4	TIEN	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	RBFI	Receive buffer full int. condition	1 2 bytes 0 1 byte	0	R/W	
		D0	RXEN	UART enable	1 Enable 0 Disable	0	R/W	

D7 Reserved

D6 REIEN: Receive Error Interrupt Enable Bit

Enables an interrupt request to be output to the ITC when a receive error has occurred.
1 (R/W): Enable
0 (R/W): Disable (default)

Set this bit to 1 when processing receive errors in the interrupt handler routine.

D5 RIEN: Receive Buffer Full Interrupt Enable Bit

Enables an interrupt request to be output to the ITC when the receive data buffer receives the number of data specified by RBFI (D1).
1 (R/W): Enable
0 (R/W): Disable (default)

Set this bit to 1 when reading the received data in the interrupt handler routine.

D4 TIEN: Transmit Buffer Empty Interrupt Enable Bit

Enables an interrupt request to be output to the ITC when the transmit data written to the transmit data buffer is transferred to the shift register (when data transmission starts).
1 (R/W): Enable
0 (R/W): Disable (default)

Set this bit to 1 when writing transmit data to the transmit data buffer in the interrupt handler routine.

D[3:2] Reserved

D1 RBFI: Receive Buffer Full Interrupt Condition Bit

Sets the number of data in the receive data buffer to generate a receive-buffer full interrupt.
1 (R/W): 2 bytes
0 (R/W): 1 byte (default)

When the specified number of received data is loaded to the receive data buffer, the UART outputs an interrupt request pulse to the ITC if the receive buffer full interrupt has been enabled (RIEN = 1). If RBFI is 0, an interrupt request pulse is output when a received data is loaded to the receive data buffer (when the RDRY flag (D1/UART_ST register) goes 1). If RBFI is 1, an interrupt request pulse is output when two received data occupy the receive data buffer (when the RD2B flag (D3/UART_ST register) goes 1).

D0 RXEN: UART Enable Bit

Enables the UART to transmit/receive data.
1 (R/W): Enable
0 (R/W): Disable (default)

Before the UART can transmit/receive data, RXEN must be set to 1. When RXEN is set to 0, data transmission/reception is disabled.
Always make sure RXEN = 0 before setting the transfer conditions.
Writing 0 to RXEN also clears the transmit data buffer.

0x4105: UART Expansion Register (UART_EXP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
UART Expansion Register (UART_EXP)	0x4105 (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	IRCLK[2:0]	IrDA receive detection clock select	IRCLK[2:0]	Clock	0x0	R/W	
					0x7	PCLK•1/128			
					0x6	PCLK•1/64			
					0x5	PCLK•1/32			
					0x4	PCLK•1/16			
					0x3	PCLK•1/8			
					0x2	PCLK•1/4			
					0x1	PCLK•1/2			
					0x0	PCLK•1/1			
		D3–1	–	reserved	–		–	–	0 when being read.
		D0	IRMD	IrDA mode select	1 On	0 Off	0	R/W	

D7 Reserved

D[6:4] IRCLK[2:0]: IrDA Receive Detection Clock Select Bits

These bits select a prescaler output clock as the input pulse detection clock.

Table VI.1.9.2 Selecting the IrDA Receive Detection Clock

IRCLK[2:0]	Prescaler output clock
0x7	PCLK•1/128
0x6	PCLK•1/64
0x5	PCLK•1/32
0x4	PCLK•1/16
0x3	PCLK•1/8
0x2	PCLK•1/4
0x1	PCLK•1/2
0x0	PCLK•1/1

(Default: 0x0)

This clock must be faster than the transfer clock sclk supplied from the CLG_T16U0 timer or input from the #SCLK0 pin.

The demodulator regards a low pulse of which the width is longer than two cycles of the IrDA receive detection clock as a valid low pulse. Select an appropriate prescaler output clock that can detect a minimum 1.41 μs width of an input pulse.

D[3:1] Reserved

D0 IRMD: IrDA Mode Select Bit

Turns the IrDA interface function on and off.

1 (R/W): On

0 (R/W): Off (default)

Set this bit to 1 when using the IrDA interface. When set to 0, the module functions as a standard UART without IrDA.

VI.1.10 Precautions

- Before setting the bits listed below, make sure the transmit and receive operations are disabled (RXEN = 0).
  - All bits (SSCK, STPB, PMD, PREN, and CHLN) of the UART_MOD register (0x4103)
  - All bits (RBFI, TIEN, RIEN, and REIEN except RXEN) of the UART_CTL register (0x4104)
  - All bits (IRMD and IRCLK[2:0]) of the UART_EXP register (0x4105)

  ∗ **RXEN**: UART Enable Bit in the UART Control (UART_CTL) Register (D0/0x4104)
- When the UART is transmitting or receiving data, do not set RXEN to 0.
- The maximum transfer rate of the UART is limited to 460800 bps (115200 bps in IrDA mode). Do not set a transfer rate that exceeds the limit.
- When the RXEN bit is set to 0 to disable transmit/receive operations, the transmit data buffer is cleared (initialized). Therefore, make sure that the buffers do not contain any data waiting for transmission before writing 0 to the RXEN bit.
- The IrDA receive detection clock must be faster than the transfer clock sclk supplied from the CLG_T16U0 timer or input from the #SCLK0 pin.
- The demodulator regards a low pulse of which the width is longer than two cycles of the IrDA receive detection clock as a valid low pulse. Select an appropriate prescaler output clock as the IrDA receive detection clock so that it will be able to detect a minimum 1.41 μs width of an input pulse.

VI.2 I2C

VI.2.1 Configuration of the I2C Module

The S1C17501 equipped with an I2C bus interface module for high-speed synchronous serial communication. This I2C module operates as a master using the clock supplied from the CLG_T8I timer (supports single master mode only). It supports standard (100 kbps) and fast (400 kbps) modes, and 7-bit/10-bit slave addressing. The I2C module includes a noise remove function to secure reliable data transfer.
Also it can generate two types of interrupts (transmit buffer empty and receive buffer full interrupts), this makes it possible to process continuous serial data transfer simply in an interrupt handler.
Figure VI.2.1.1 shows the structure of the I2C module.

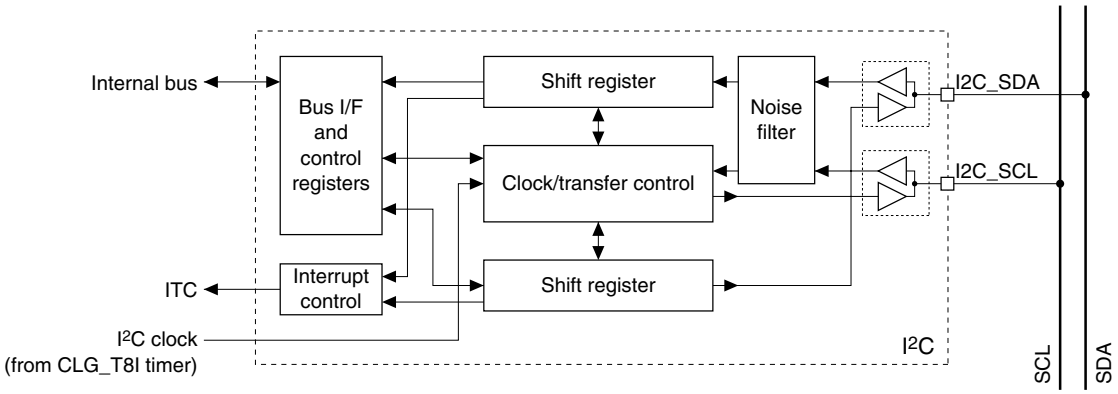


Figure VI.2.1.1 Structure of I2C Module

Note: The I2C module does not have a clock stretch function. Therefore, it does not support I2C slave devices that use clock stretch for synchronization of data communication.

VI.2.2 I2C I/O Pins

Table VI.2.2.1 lists the I2C pins.

Table VI.2.2.1 List of I2C Pins

Pin name	I/O	Size	Function
I2C_SDA	I/O	1	I2C data input/output pin This pin inputs serial data from the I2C bus and outputs serial data to the I2C bus.
I2C_SCL	I/O	1	I2C clock input/output pin This pin inputs the I2C_SCL line status and outputs the serial clock to the I2C bus.

The I2C input/output pins (I2C_SDA, I2C_SCL) are shared with the I/O ports and they are initialized as general-purpose I/O port pins by default. Before using these pins for the I2C, the pin functions must be switched using the Port Function Select Register.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.2.3 I2C Clock

The I2C module uses the internal clock output from the CLG_T8I timer as the synchronous clock. This clock drives the shift register and is output from the I2C_SCL pin to the slave I2C device.
Program the CLG_T8I timer so that it will output a clock according to the transfer rate. Refer to Section II.4, "Clock Generator (CLG)," for controlling the CLG_T8I timer.

The I2C module does not function as a slave device. The I2C_SCL input is used to check the I2C_SCL status of the I2C bus but it is not used to input synchronous clock.

VI.2.4 Setting before Starting Data Transfer

The I²C module has a noise remove function selectable in the application program.

Noise remove function

The I²C module contains a function to remove noise from the I2C_SDA and I2C_SCL input signals. This function is enabled by setting NSERM (D4/I2C_CTL register) to 1.
Note, however, that the I²C clock (CLG_T8I timer output clock) frequency must be a 1/6 of PCLK or lower to use the noise remove function.

* **NSERM**: Noise Remove On/Off Bit in the I²C Control (I2C_CTL) Register (D4/0x4342)

VI.2.5 Data Transmit/Receive Control

Before starting data transfer, set up the conditions by the procedure below.
(1) Set up the CLG_T8I timer to output the I2C clock. See Section II.4.
(2) Select optional functions. See Section VI.2.4.
(3) Set up the interrupt conditions if the I2C interrupt is used. See Section VI.2.6.

Note: Make sure that the I2C module is disabled (I2CEN/I2C_EN register = 0) before setting the conditions above.

∗ **I2CEN**: I2C Enable Bit in the I2C Enable (I2C_EN) Register (D0/0x4340)

Enabling data transmission/reception

First, set the I2CEN bit (D0/I2C_EN register) to 1 to enable I2C operation. This makes the I2C in ready-to-transmit/receive status and enables clock output.

Note: Do not set the I2CEN bit to 0 while the I2C module is transmitting/receiving data.

Starting data transmission/reception

To start data transmission, the I2C master (this module) must generate a START condition, and then send a slave address to establish the communication.

(1) Register setting procedure

To generate a START condition, set the following registers in the order shown below:
1. Set the slave address to RTDT[7:0] (D[7:0]/I2C_DAT register).
2. Set TXE (D9/I2C_DAT register) to 1.
3. Set STRT (D0/I2C_CTL register) to 1.

∗ **RTDT[7:0]**: Receive/Transmit Data Bits in the I2C Data (I2C_DAT) Register (D[7:0]/0x4344)
∗ **TXE**: Transmit Execution Bit in the I2C Data (I2C_DAT) Register (D9/0x4344)
∗ **STRT**: Start Control Bit in the I2C Control (I2C_CTL) Register (D0/0x4342)

This procedure generates the communication waveforms as shown in Items (2) and (3) below. Be sure to follow the register setting procedure.

(2) Generating a START condition

The START condition is a state in which the I2C_SDA line is pulled down to low with the I2C_SCL line held at high.

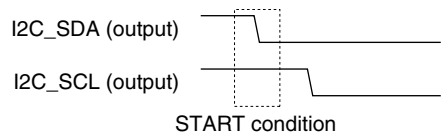


Figure VI.2.5.1 START Condition

Set the STRT bit (D0/I2C_CTL register) to 1 to generate a START condition.

∗ **STRT**: Start Control Bit in the I2C Control (I2C_CTL) Register (D0/0x4342)

After a START condition has been generated, STRT is automatically reset to 0.

(3) Sending a slave address

After a START condition has been generated, the I2C master (this module) sends the address of the slave to communicate and a bit to specify data transfer direction. The I2C module supports two slave address sizes: 7-bit slave address and 10-bit slave address. The I2C module sends the slave address bits with the transfer direction bit using the 8-bit transmit/receive data register. So one 7-bit slave address can be sent at a time. A 10-bit address should be sent in two parts or three parts with software control. Figure 20.5.2 shows the address data format.

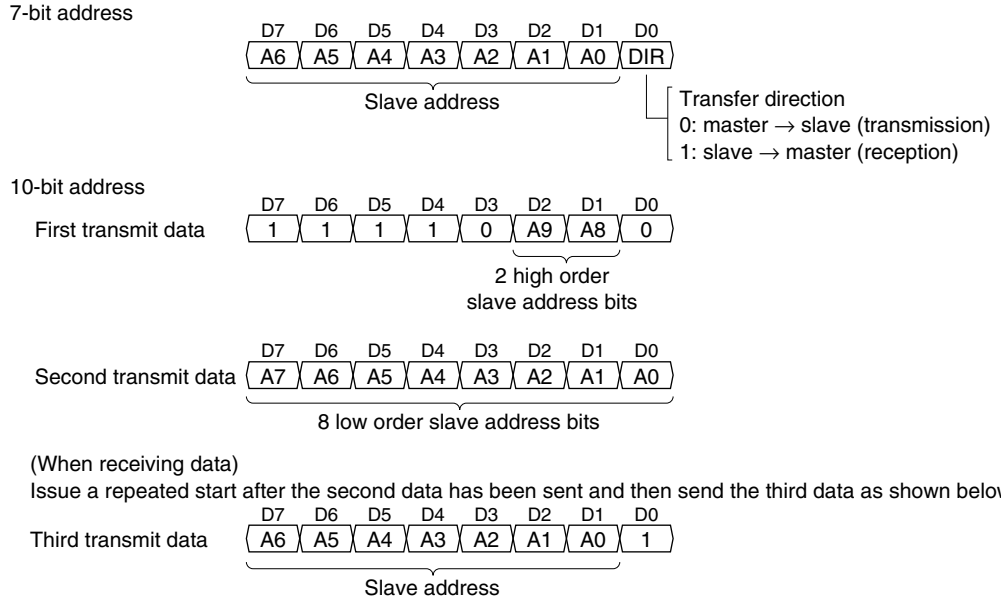


Figure VI.2.5.2 Transmit Data to Specify Slave Address and Data Direction

The transfer direction bit specifies the direction for the data transfer that follows the slave address transfer. Set the transfer direction bit to 0 when transmitting data from the master to the slave; set it to 1 when receiving data from the slave.

Configure an 8-bit data as above and set it into the transmit/receive data register. After that control data transmission as described below.

The slave address with a transfer direction bit must be sent once after a START condition has been generated.

After a slave address has been sent, perform data transmission or data reception as many times as necessary. It is necessary to perform data transmission or data reception according to the transfer direction specified with the slave address.

Data transmit control

The following explains how to transmit data. The slave address should be sent in the same way.

To transmit byte data, set the data to the RTDT[7:0] bits (D[7:0]/I2C_DAT register). At the same time, set the TXE bit (D9/I2C_DAT register) to 1 to execute one byte data transmission.

- **RTDT[7:0]**: Receive/Transmit Data Bits in the I2C Data (I2C_DAT) Register (D[7:0]/0x4344)
- **TXE**: Transmit Execution Bit in the I2C Data (I2C_DAT) Register (D9/0x4344)

When the TXE bit is set to 1, the I2C module starts data transmission in sync with the clock. If a START condition is being generated or the previous data is being transferred, the I2C module starts data transmission after waiting for completion of the process.

First, the I2C module transfer the written data to the shift register and starts outputting the clock from the I2C_SCL pin. At this time, TXE is reset to 0 and a cause of interrupt occurs. This allows the program to set the next transmit data and TXE again.

The data bits in the shift register are shifted one by one at the falling edge of the clock and are output from the I2C_SDA pin. The MSB is transmitted first.

The I2C module outputs nine clocks for one data transmission. In the ninth clock cycle, the I2C module sets the I2C_SDA signal into high-impedance status to input an ACK or NAK bit from the slave.

If the slave could receive byte data, it returns an ACK (0) bit to the master. If the slave could not receive byte data, the I2C_SDA line is not pulled down. The I2C module regards this status as a NAK (1) returned (transmission fails).

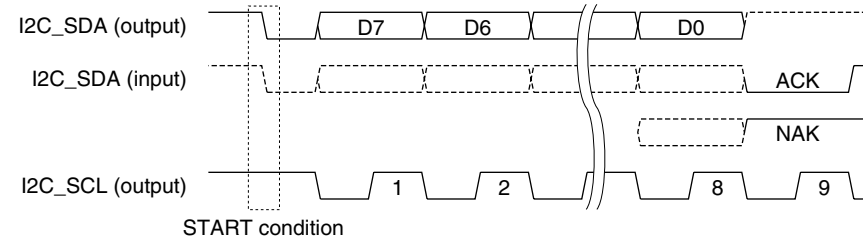


Figure VI.2.5.3 ACK and NAK

The I²C module provides two status bits for data transmit control, TBUSY flag (D8/I2C_CTL register) and RTACK bit (D8/I2C_DAT register).

- **TBUSY**: Transmit Busy Flag Bit in the I²C Control (I2C_CTL) Register (D8/0x4342)
- **RTACK**: Receive/Transmit ACK Bit in the I²C Data (I2C_DAT) Register (D8/0x4344)

The TBUSY flag indicates the data transmit status; it goes 1 when data transmission (including slave address transmission) starts and returns to 0 upon completion of data transmission. Also TBUSY returns to 0 in wait state. Read this flag to check whether the I²C module is busy or idle.
The RTACK bit indicates whether the slave returned ACK or not in a previous data transmission; it goes 0 when an ACK bit is received or goes 1 if an ACK bit is not received.

Data receive control

The following explains how to receive data. Even in data reception, a START condition must be generated and a slave address with the transfer direction bit set to 1 must be sent before starting data reception.
To receive byte data, set the RXE bit (D10/I2C_DAT register) to 1 to execute receiving one byte data. RXE can be set to 1 at the same time when TXE (D9/I2C_DAT register) is set to 1 for sending a slave address. If TXE and RXE are both set to 1, TXE is effective.

- **RXE**: Receive Execution Bit in the I²C Data (I2C_DAT) Register (D10/0x4344)

When the RXE bit is set to 1 and the I²C module is ready to receive, the I²C module sets the I2C_SDA line into high-impedance and starts outputting the clock from the I2C_SCL pin. The data bits are fetched in the shift register one by one at the rising edge of the clock. The MSB is received first.
RXE is reset to 0 during fetching D6.
When eight data bits are received in the shift register, the received data is loaded into RTDT[7:0].
The I²C module provides two status bits for data receive control, RBRDY (D11/I2C_DAT register) and RBUSY (D9/I2C_CTL register).

- **RBRDY**: Receive Buffer Ready Bit in the I²C Data (I2C_DAT) Register (D11/0x4344)
- **RBUSY**: Receive Busy Flag Bit in the I²C Control (I2C_CTL) Register (D9/0x4342)

The RBRDY flag indicates the received data status; it goes 1 when the received data in the shift register is loaded into RTDT[7:0] and returns to 0 when the received data is read from RTDT[7:0]. An interrupt can be generated when this flag goes 1. Use this interrupt or read the RBRDY flag to check that RTDT[7:0] contains valid data when reading received data. If the next data has been received before the previous received data in RTDT[7:0] is read, the previous received data is overwritten with the new data.
The RBUSY flag indicates the data receive operation status; it goes 1 when data reception starts and returns to 0 upon completion of data reception. Also RBUSY returns to 0 in wait state. Read this flag to check whether the I²C module is busy or idle.
The I²C module outputs nine clocks for one data transmission. In the 9th clock cycle, 0 or 1 set to RTACK(D8/I2C_DAT register) is sent from the I2C_SDA pin to the slave as ACK, NAK response, respectively. Do not change RTACK while ACK or NAK is in response.

Note: Receive buffer full interrupt occurs when ACK/NAK is in response (the 9th clock cycle). When the RTACK setting is rewritten without waiting for the end of the ACK/NAK period immediately after this interrupt is occured, ACK/NAK response output to the I2C_SDA pin is changed during the response time, the correct communication can not be performed.

Terminating data transmission/reception (generating a STOP condition)

To terminate data transfer after all data have been sent/received, the I2C master (this module) must generate a STOP condition. The STOP condition is a state in which the I2C_SDA line is pulled up from low to high with the I2C_SCL line held at high.

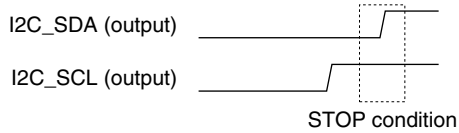


Figure VI.2.5.4 STOP Condition

Set the STP bit (D1/I2C_CTL register) to 1 to generate a STOP condition.

∗ **STP**: Stop Control Bit in the I2C Control (I2C_CTL) Register (D1/0x4342)

When STP is set to 1, the I2C module pulls up the I2C_SDA line from low to high with the I2C_SCL line held at high to generate a STOP condition on the I2C bus. This makes the I2C bus in free status.
Furthermore, the I2C module allows presetting for generating a STOP condition in advance. To do this, set STP to 1 after checking if the I2C is operating (TBUSY = 1 or RBUSY = 1). A STOP condition will be generated upon completion of data transmission/reception (including an ACK transfer).
STP is automatically reset to 0 after a STOP condition has been generated.

The I2C module does not support repeated START condition. A STOP condition cannot be omitted before generating a new START condition for starting the next data transfer.

Wait state by setting TXE, RXE, STRT, and STP

If TXE (D9/I2C_DAT register), RXE (D10/I2C_DAT register), STRT (D0/I2C_CTL register), and STP (D1/I2C_CTL register) have all been set to 0 when byte data and ACK transfer have finished, the I2C module fixes the I2C_SCL output at low and enters wait state. The wait state will be canceled by writing 1 to TXE or RXE to resume data transmission/reception or by writing 1 to STP to generate a STOP condition.

Disabling data transmission/reception

After the stop condition has been generated, write 0 to I2CEN to disable data transfers. To determine whether the stop condition has been generated, check to see if STP is automatically cleared to 0 after it is set to 1 by polling.
When I2CMEN is set to 0 while the I2C bus is in busy status, the SCL0 and SDA0 output levels and transfer data at that point cannot be guaranteed.

Timing charts

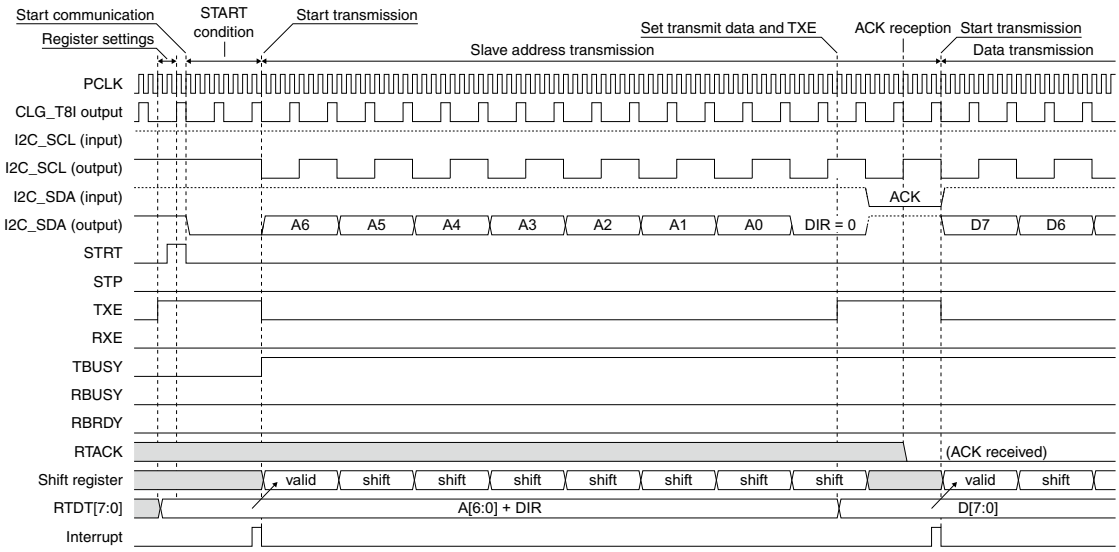


Figure VI.2.5.5 I²C Timing Chart 1 (START condition → data transmission)

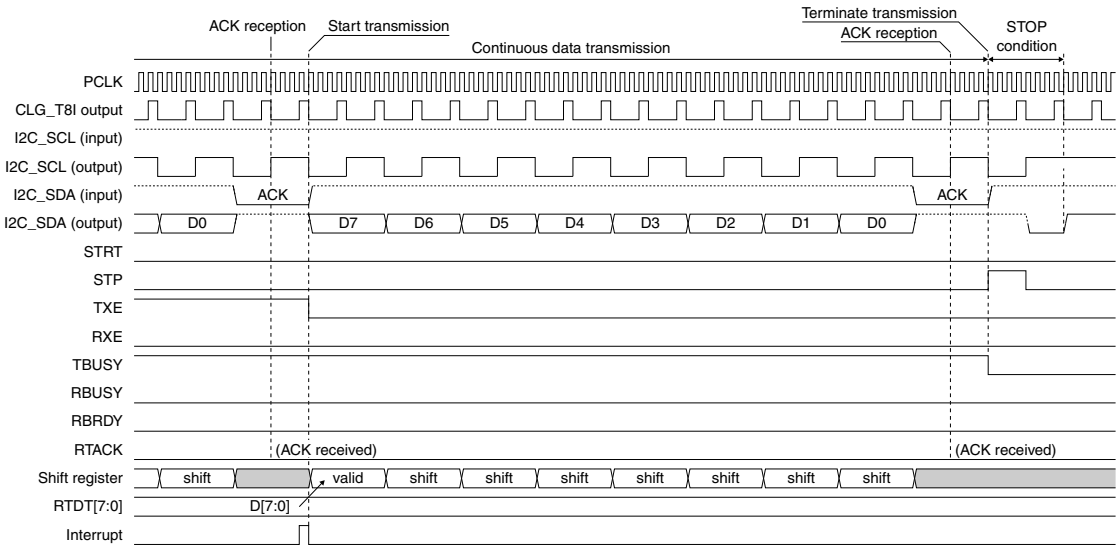


Figure VI.2.5.6 I²C Timing Chart 2 (data transmission → STOP condition)

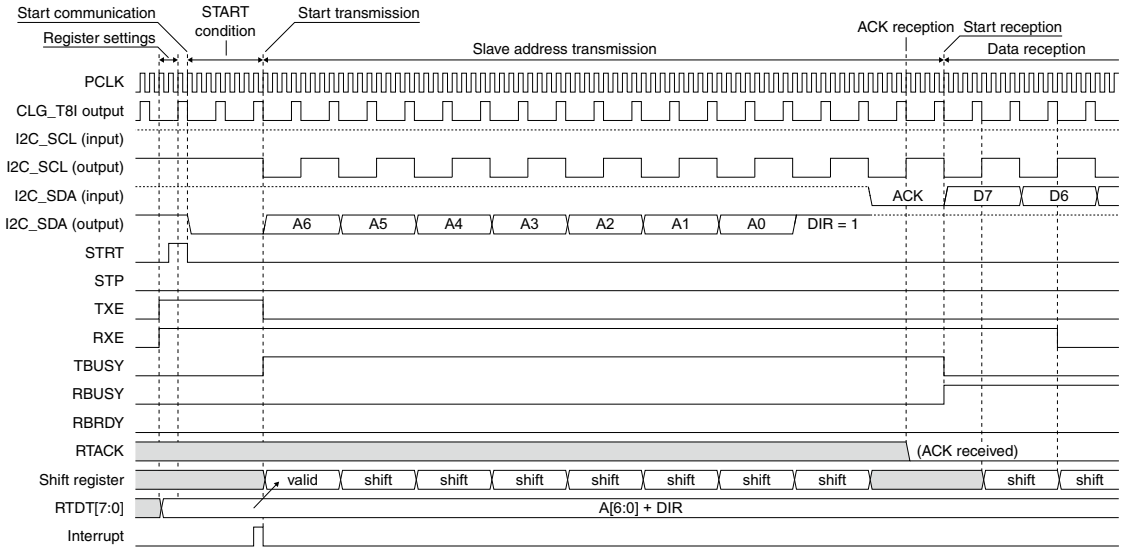


Figure VI.2.5.7 I²C Timing Chart 3 (START condition → data reception)

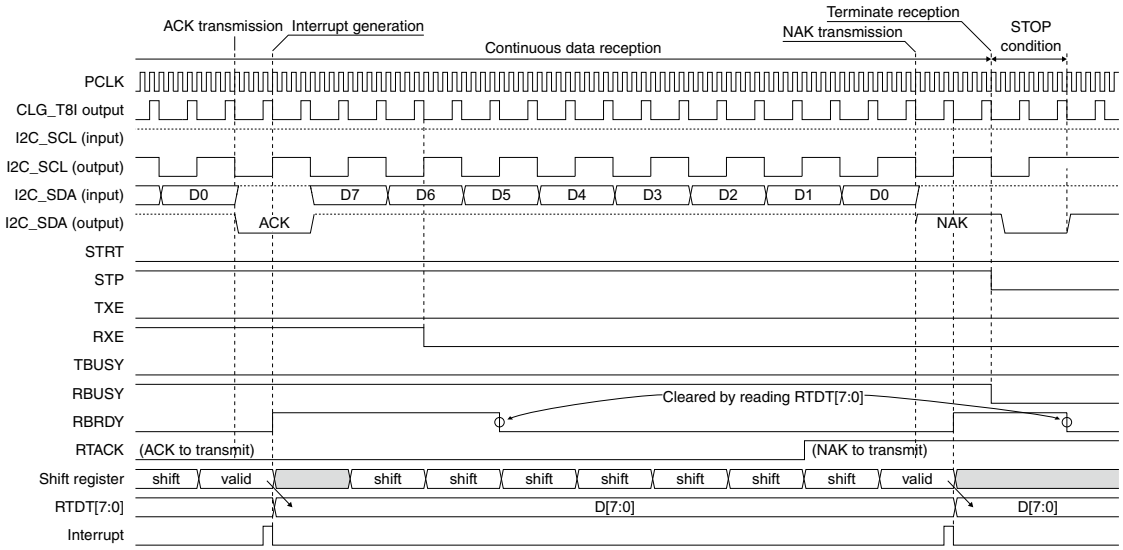


Figure VI.2.5.8 I²C Timing Chart 4 (data reception → STOP condition)

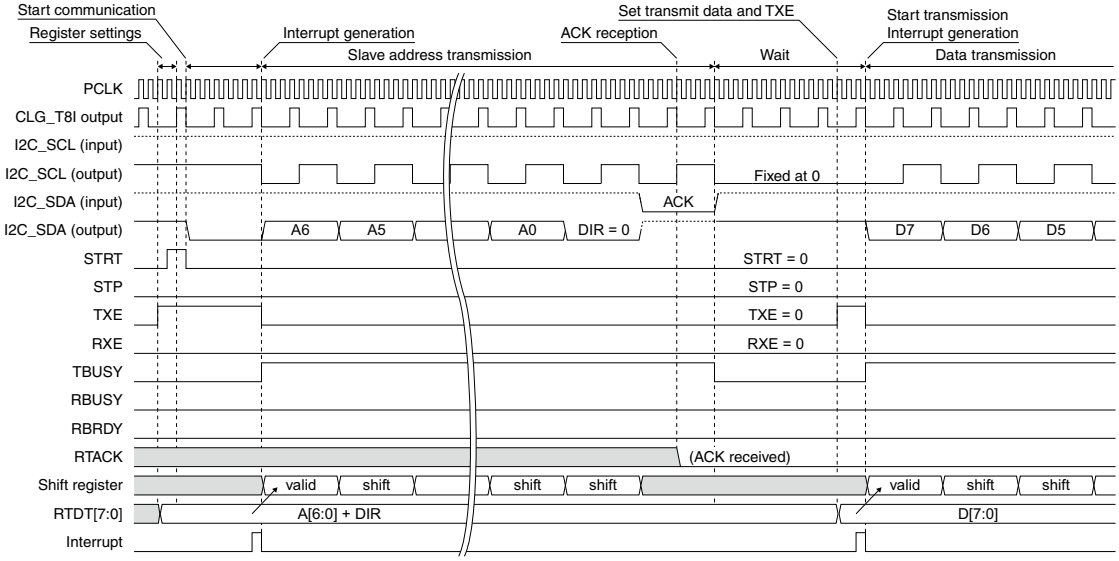


Figure VI.2.5.9 I²C Timing Chart 5 (wait state)

VI.2.6 I2C Interrupt

The I2C module can generate the following two types of interrupts:
• Transmit buffer empty interrupt
• Receive buffer full interrupt

The I2C module has one interrupt signal to be output to the interrupt controller (ITC) and it is shared with the two causes of interrupt.

Transmit buffer empty interrupt

Set the TINTE bit (D0/I2C_ICTL register) to 1 when using this interrupt. If TINTE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **TINTE**: Transmit Interrupt Enable Bit in the I2C Interrupt Control (I2C_ICTL) Register (D0/0x4346)

When the transmit data set in RTDT[7:0] (D[7:0]/I2C_DAT register) is transferred to the shift register, the I2C module outputs an interrupt request pulse to the ITC if the transmit buffer empty interrupt has been enabled (TINTE = 1).

∗ **RTDT[7:0]**: Receive/Transmit Data Bits in the I2C Data (I2C_DAT) Register (D[7:0]/0x4344)

If other interrupt conditions are satisfied, an interrupt is generated.

Receive buffer full interrupt

Set the RINTE bit (D1/I2C_ICTL register) to 1 when using this interrupt. If RINTE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **RINTE**: Receive Interrupt Enable Bit in the I2C Interrupt Control (I2C_ICTL) Register (D1/0x4346)

When data received in the shift register is loaded to RTDT[7:0], the I2C module outputs an interrupt request pulse to the ITC if the receive buffer full interrupt has been enabled (RINTE = 1).

If other interrupt conditions are satisfied, an interrupt is generated.

ITC registers for I2C interrupts

The following shows the control bits of the ITC provided for the I2C module:

Interrupt flag
∗ **IIFT7**: I2C Interrupt Flag Bit in the Interrupt Flag (ITC_IFLG) Register (D15/0x4300)

Interrupt enable bit
∗ **IIEN7**: I2C Interrupt Enable Bit in the Interrupt Enable (ITC_EN) Register (D15/0x4302)

Interrupt level setup bits
∗ **IILV7[2:0]**: I2C Interrupt Level Bits in the Internal Interrupt Level Setup (ITC_ILV3) Register 3 (D[10:8]/0x4314)

When the I2C outputs an interrupt request pulse, the interrupt flag IIFT7 is set to 1.
If the interrupt enable bit IIEN7 has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the I2C interrupt, set the IIEN7 bit to 0.
The IIFT7 flag is always set to 1 by the I2C interrupt request pulse, regardless of how the IIEN7 bit is set (even when set to 0).
The interrupt level setup bits IILV7[2:0] set the interrupt level (0 to 7) of the I2C interrupt.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register PSR) in the S1C17 Core is set to 1.
- The I2C interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, "Interrupt Controller (ITC)."

Interrupt vector

The following shows the vector number and vector address for the I2C interrupt:

Vector number: 19 (0x13)
Vector address: TTBR + 0x4c

VI.2.7 Details of Control Registers

Table VI.2.7.1 List of I2C Registers

Address	Register name		Function
0x4340	I2C_EN	I2C Enable Register	Enables the I2C module.
0x4342	I2C_CTL	I2C Control Register	Controls the I2C operation and indicates transfer status.
0x4344	I2C_DAT	I2C Data Register	Transmit/receive data
0x4346	I2C_ICTL	I2C Interrupt Control Register	Controls the I2C interrupt.

The following describes each I2C register. These are all 16-bit registers.

Notes: • When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

• Be sure to use 16-bit access instructions for reading/writing from/to the I2C registers. The I2C registers do not allow reading/writing using 32-bit and 8-bit access instructions.

0x4340: I2C Enable Register (I2C_EN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I2C Enable Register (I2C_EN)	0x4340 (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	I2CEN	I2C enable	1 Enable 0 Disable	0	R/W	

D[15:1] Reserved

D0 I2CEN: I2C Enable Bit

Enables/disables operation of the I2C module.

1 (R/W): Enable

0 (R/W): Disable (default)

When I2CEN is set to 1, the I2C module is activated and data transfer is enabled.

When I2CEN is set to 0, the I2C module goes off.

0x4342: I2C Control Register (I2C_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I2C Control Register (I2C_CTL)	0x4342 (16 bits)	D15–10	–	reserved	–	–	–	0 when being read.
		D9	RBUSY	Receive busy flag	1 Busy 0 Idle	0	R	
		D8	TBUSY	Transmit busy flag	1 Busy 0 Idle	0	R	
		D7–5	–	reserved	–	–	–	0 when being read.
		D4	NSERM	Noise remove on/off	1 On 0 Off	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	STP	Stop control	1 Stop 0 Ignored	0	R/W	
		D0	STRT	Start control	1 Start 0 Ignored	0	R/W	

D[15:10] Reserved

D9 RBUSY: Receive Busy Flag Bit

Indicates the I2C receive operation status.

1 (R): Busy

0 (R): Idle (default)

RBUSY is set to 1 when the I2C starts data reception and stays 1 while data reception is in progress. RBUSY is reset to 0 upon completion of receive operation. Also RBUSY returns to 0 in wait state.

D8 TBUSY: Transmit Busy Flag Bit

Indicates the I2C transmit operation status.

1 (R): Busy

0 (R): Idle (default)

TBUSY is set to 1 when the I2C starts data transmission and stays 1 while data transmission is in progress. TBUSY is reset to 0 upon completion of transmit operation. Also TBUSY returns to 0 in wait state.

D[7:5] Reserved

D4 NSERM: Noise Remove On/Off Bit

Turns the noise remove function on and off.

1 (R/W): On

0 (R/W): Off (default)

The I2C module contains a function to remove noise from the I2C_SDA and I2C_SCL input signals. This function is enabled by setting NSERM to 1.

Note, however, that the I2C clock (CLG_T8I timer output clock) frequency must be 1/6 of PCLK or lower to use the noise remove function.

D[3:2] Reserved

D1 STP: Stop Control Bit

Generates a STOP condition.

1 (R/W): Generate STOP condition

0 (R/W): Ignore (default)

When STP is set to 1, the I2C module pulls up the I2C_SDA line from low to high with the I2C_SCL line held at high to generate a STOP condition on the I2C bus. This makes the I2C bus in free status.

Furthermore, the I2C module allows presetting for generating a STOP condition in advance. To do this, set STP to 1 after checking if the I2C is operating (TBUSY = 1 or RBUSY = 1). A STOP condition will be generated upon completion of data transmission/reception (including an ACK transfer).

STP is automatically reset to 0 after a STOP condition has been generated.

D0 STRT: Start Control Bit

Generates a START condition.
1 (R/W): Generate START condition
0 (R/W): Ignore (default)

When STRT is set to 1, the I2C module pulls down the I2C_SDA line to low with the I2C_SCL line held at high to generate a START condition on the I2C bus. This makes the I2C bus in busy status.
To generate a START condition, set the following registers in the order shown below:

1. Set the slave address to RTDT[7:0] (D[7:0]/I2C_DAT register).
2. Set TXE (D9/I2C_DAT register) to 1.
3. Set STRT to 1.

STRT is automatically reset to 0, after a START condition has been generated.

0x4344: I2C Data Register (I2C_DAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I2C Data Register (I2C_DAT)	0x4344 (16 bits)	D15–12	–	reserved	–	–	–	0 when being read.
		D11	RBRDY	Receive buffer ready	1 Ready 0 Empty	0	R	
		D10	RXE	Receive execution	1 Receive 0 Ignored	0	R/W	
		D9	TXE	Transmit execution	1 Transmit 0 Ignored	0	R/W	
		D8	RTACK	Receive/transmit ACK	1 Error 0 ACK	0	R/W	
		D7–0	RTDT[7:0]	Receive/transmit data RTDT7 = MSB RTDT0 = LSB	0x0 to 0xff	0x0	R/W	

D[15:12] Reserved

D11 RBRDY: Receive Buffer Ready Bit

Indicates the receive buffer status.

1 (R): Received data is present

0 (R): No received data is present (default)

The RBRDY flag goes 1 when the received data in the shift register is loaded into RTDT[7:0] (D[7:0]) and returns to 0 when the received data is read from RTDT[7:0]. An interrupt can be generated when this flag goes 1. Use this interrupt or read the RBRDY flag to check that RTDT[7:0] contains valid data when reading received data.

D10 RXE: Receive Execution Bit

Execute a data reception for one byte.

1 (R/W): Start data reception

0 (R/W): Ignore (default)

The I2C module starts data reception for one byte by setting RXE to 1 and TXE (D9) to 0. RXE can be set to 1 for the next data reception even if a slave address is being transmitted or data is being received. RXE is reset to 0 when D6 is input to the shift register.

D9 TXE: Transmit Execution Bit

Execute a data transmission for one byte.

1 (R/W): Start data transmission

0 (R/W): Ignore (default)

Set the transmit data to RTDT[7:0] (D[7:0]) and write 1 to TXE to start data transmission. TXE can be set to 1 for the next data transmission even if a slave address or data is being transmitted. TXE is reset to 0 when the data set in RTDT[7:0] is transferred to the shift register.

D8 RTACK: Receive/Transmit ACK Bit

In data transmission

Indicates the acknowledge bit status.

1 (R/W): Error (NAK)

0 (R/W): ACK (default)

This bit is set to 0 when the slave returned ACK after one-byte data has been transmitted. This indicates that the slave could receive the data normally. If this bit is set to 1, the slave may be inactive or it could not receive the data normally.

In data reception

Set the acknowledge bit to be sent to the slave.

1 (R/W): Error (NAK)

0 (R/W): ACK (default)

To return ACK to the slave after data is received, set RTACK to 0 before the I2C module sends the acknowledge bit.

To return NAK, set RTACK to 1.

D[7:0] RTDT[7:0]: Receive/Transmit Data Bits

In data transmission

Set a transmit data in this register. (Default: 0x0)

Data transmission begins when TXE (D9) is set to 1. If a slave address or data is being transmitted, a new transmission starts after the current transmission has completed. The serial-converted data is output from the I2C_SDA pin beginning with the MSB, in which the bits set to 0 are output as low-level signals. When the data set in this register is transferred to the shift register, a cause of transmit buffer empty interrupt occurs. The next transmit data can be written to the register after that.

In data reception

The received data can be read from this register. (Default: 0x0)

Data reception begins when RXE (D10) is set to 1. If a slave address is being transmitted or data is being received, a new reception starts after the current reception has completed. When a receive operation is completed and the data received in the shift register is loaded to this register, the RBRDY flag (D11) is set and a cause of receive buffer full interrupt occurs. Thereafter, the data can be read out at any time before a receive operation for the next data is completed.

If the next data receive operation is completed before this register is read out, the data in it is overwritten with the newly received data.

The serial data input from the I2C_SDA pin is converted into parallel data beginning with the MSB, with the high-level signals changed to 1s and the low-level signals changed to 0s. The resulting data is stored in this register.

0x4346: I2C Interrupt Control Register (I2C_ICTL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
I2C Interrupt Control Register (I2C_ICTL)	0x4346 (16 bits)	D15–2	–	reserved	–				–	–	0 when being read.
		D1	RINTE	Receive interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	TINTE	Transmit interrupt enable	1	Enable	0	Disable	0	R/W	

D[15:2] Reserved

D1 RINTE: Receive Interrupt Enable Bit

Enables/disables the I2C receive buffer full interrupt.
1 (R/W): Enable
0 (R/W): Disable (default)

When RINTE is set to 1, I2C receive buffer full interrupt requests to the ITC are enabled. A receive buffer full interrupt request occurs when the data received in the shift register is loaded to RTDT[7:0] (D[7:0]/I2C_DAT register) (receive operation completed).
When RINTE is set to 0, an I2C receive buffer full interrupt is not generated.

D0 TINTE: Transmit Interrupt Enable Bit

Enables/disables the I2C transmit buffer empty interrupt.
1 (R/W): Enable
0 (R/W): Disable (default)

When TINTE is set to 1, I2C transmit buffer empty interrupt requests to the ITC are enabled. A transmit buffer empty interrupt request occurs when the data written to RTDT[7:0] (D[7:0]/I2C_DAT register) is transferred to the shift register.
When TINTE is set to 0, an I2C transmit buffer empty interrupt is not generated.

VI.3 SPI (SPI CH.0)

VI.3.1 Configuration of the SPI CH.0

The S1C17501 equipped with a synchronous serial interface module (hereafter SPI CH.0). The SPI CH.0 supports both master and slave modes and performs 8-bit serial data transfer. Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
The SPI CH.0 includes a transmit data buffer and a receive data buffer separately from the shift registers, and can generate two types of interrupts (transmit data buffer empty and receive data buffer full), this makes it possible to process continuous serial data transfers simply in an interrupt handler.
Figure VI.3.1.1 shows the structure of the SPI CH.0.

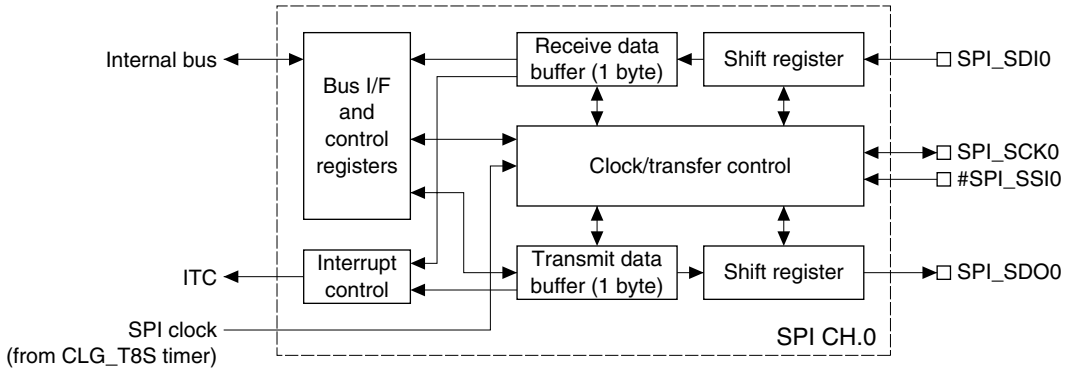


Figure VI.3.1.1 Structure of SPI CH.0

VI.3.2 SPI CH.0 I/O Pins

Table VI.3.2.1 lists the SPI CH.0 pins.

Table VI.3.2.1 List of SPI CH.0 Pins

Pin name	I/O	Size	Function
SPI_SDI0	I	1	SPI CH.0 data input pin This pin inputs serial data from the SPI bus.
SPI_SDO0	O	1	SPI CH.0 data output pin This pin outputs serial data to the SPI bus.
SPI_SCK0	I/O	1	SPI CH.0 external clock input/output pin This pin outputs the SPI clock when the SPI CH.0 is in master mode. This pin inputs an external clock when the SPI CH.0 is in slave mode.
#SPI_SSI0	I	1	SPI CH.0 slave select signal (active low) input pin A low level input to this pin selects this SPI CH.0 device in slave mode.

The SPI CH.0 input/output pins (SPI_SDI0, SPI_SDO0, SPI_SCK0, #SPI_SSI0) are shared with the I/O ports and they are initialized as general-purpose I/O port pins by default. Before using these pins for the SPI CH.0, the pin functions must be switched using the Port Function Select registers.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.3.3 SPI Clock

In master mode, the SPI CH.0 uses the internal clock output from the CLG_T8S timer as the SPI clock. This clock drives the shift register and is output from the SPI_SCK0 pin to the slave device.
Program the CLG_T8S timer so that it will output a clock according to the transfer rate. See Section II.4 "Clock Generator (CLG)," for controlling the timer.

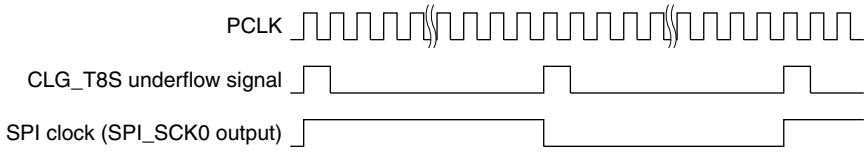


Figure VI.3.3.1 SPI Clock in Master Mode

In slave mode, the SPI CH.0 inputs the SPI clock from the SPI_SCK0 pin.

Note: The duty ratio of the clock input via the SPI_SCK0 pin must be 50%.

VI.3.4 Setting the Data Transfer Conditions

The SPI CH.0 can be set in master or slave mode and the SPI clock polarity and phase can be set using the SPI_CTL0 register.
The data length is fixed at eight bits.

Note: Make sure that the SPI CH.0 is disabled (SPEN/SPI_CTL0 register = 0) before selecting master/slave mode and setting the clock conditions.

∗ **SPEN**: SPI CH.0 Enable Bit in the SPI CH.0 Control (SPI_CTL0) Register (D0/0x4326)

Selecting master/slave mode

Use MSSL (D1/SPI_CTL0 register) to select whether the SPI CH.0 is set in master mode or slave mode. Setting MSSL to 1 selects master mode, and setting to 0 (default) selects slave mode. In master mode, the SPI CH.0 performs data transfer using the internal clock. In slave mode, the SPI CH.0 performs data transfer using a clock input from the master device.

∗ **MSSL**: Master/Slave Mode Select Bit in the SPI CH.0 Control (SPI_CTL0) Register (D1/0x4326)

Setting the SPI clock polarity and phase

Use CPOL (D2/SPI_CTL0 register) to select the SPI clock polarity. The SPI clock is configured as active low when CPOL is set to 1 or active high when CPOL is set to 0 (default).

∗ **CPOL**: Clock Polarity Select Bit in the SPI CH.0 Control (SPI_CTL0) Register (D2/0x4326)

The SPI clock phase is selected with CPHA (D3/SPI_CTL0 register).

∗ **CPHA**: Clock Phase Select Bit in the SPI CH.0 Control (SPI_CTL0) Register (D3/0x4326)

Setting these control bits determines the transfer timing as in the figure shown below.

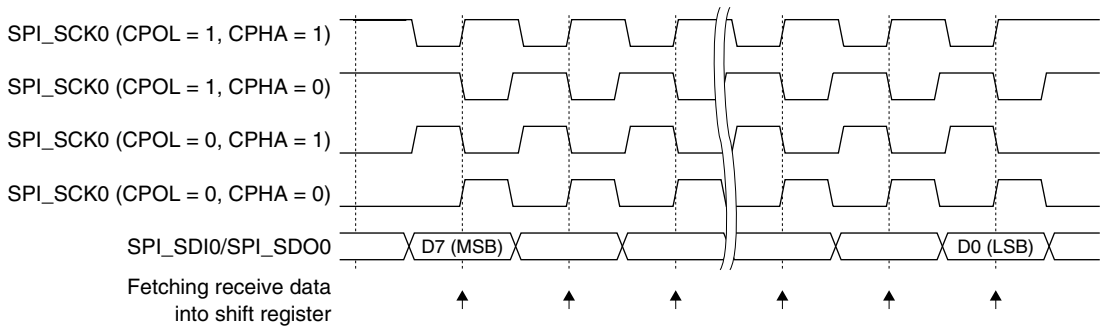


Figure VI.3.4.1 Clock and Data Transfer Timing

Note: When the SPI module is used in master mode with CPHA set to 0, the clock may change a minimum of one system clock cycle time from change of the first transmit data bit.

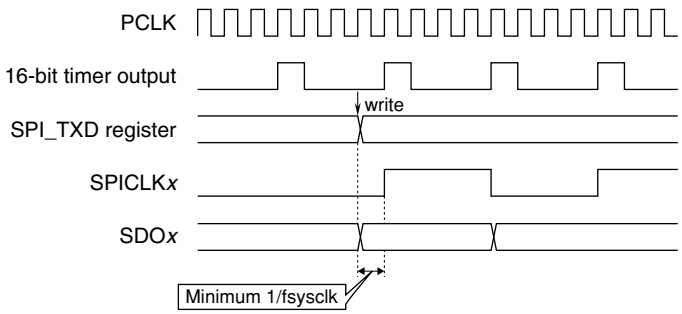


Figure VI.3.4.2 SDO*x* and SPICLK*x* Change Timings when CPHA = 0

The half SPICLK*x* cycle will be secured from change of data to change of the clock for the second and following transmit data bits and the second and following bytes during continuous transfer.

VI.3.5 Data Transmit/Receive Control

Before starting data transfer, set up the conditions as shown below.
(1) Set up the CLG_T8S timer to output the SPI clock. See Section II.4.
(2) Select either master or slave mode. See Section VI.3.4.
(3) Set up the clock conditions. See Section VI.3.4.
(4) Set up the interrupt conditions if the SPI CH.0 interrupt is used. See Section VI.3.6.

Note: Make sure that the SPI CH.0 is disabled (SPEN/SPI_CTL0 register = 0) before setting the conditions above.

* **SPEN**: SPI CH.0 Enable Bit in the SPI CH.0 Control (SPI_CTL0) Register (D0/0x4326)

Enabling data transmission/reception

First, set the SPEN bit (D0/SPI_CTL0 register) to 1 to enable SPI CH.0 operation. This puts the SPI CH.0 in ready-to-transmit/receive status and enables clock input/output.

Note: Do not set the SPEN bit to 0 while the SPI CH.0 is transmitting/receiving data.

Data transmit control

To start transmission, write transmit data to the SPI_TXD0 register (0x4322).

* **SPI_TXD0**: SPI CH.0 Transmit Data Register (0x4322)

Data is written to the transmit data buffer and the SPI CH.0 starts data transmission.
The buffered data is sent to the shift register for transmission. In master mode, the SPI CH.0 starts outputting the clock from the SPI_SCK0 pin. In slave mode, the SPI CH.0 waits for clock input from the SPI_SCK0 pin. The data bits in the shift register are shifted one by one at the rising or falling edge of the clock configured with CPHA (D3/SPI_CTL0 register) and CPOL (D2/SPI_CTL0 register) (see Figure VI.3.4.1), and are output from the SPI_SDO0 pin. The MSB of data is transmitted first.

* **CPHA**: Clock Phase Select Bit in the SPI CH.0 Control (SPI_CTL0) Register (D3/0x4326)
* **CPOL**: Clock Polarity Select Bit in the SPI CH.0 Control (SPI_CTL0) Register (D2/0x4326)

The SPI CH.0 provides two status flags for data transmit control, SPTBE (D0/SPI_ST0 register) and SPBSY (D2/SPI_ST0 register).

* **SPTBE**: Transmit Data Buffer Empty Flag Bit in the SPI CH.0 Status (SPI_ST0) Register (D0/0x4320)
* **SPBSY**: Transfer Busy Flag (Master)/ss Signal Low Flag (Slave) Bit in the SPI CH.0 Status (SPI_ST0) Register (D2/0x4320)

The SPTBE flag indicates the transmit data buffer status; it goes 0 when the application program writes data to the SPI_TXD0 register (transmit data buffer) and returns to 1 when the data in the transmit data buffer is sent to the shift register for transmitting. An interrupt can be generated when this flag goes 1 (see Section VI.3.6). Use this interrupt or read the SPTBE flag to check that the transmit data buffer becomes empty when transmitting the next data. Although the transmit data buffer size is one byte, transmit data can be written while the previous data is being transmitted as the shift register is separately provided. However, make sure that the transmit data buffer is empty before writing transmit data. If data is written when the SPTBE flag is 0, the previous transmit data in the transmit data buffer is overwritten with the new data.

In master mode, the SPBSY flag indicates the shift register status; it goes 1 when transmit data is loaded from the transmit data buffer and returns to 0 upon completion of data transmission. Read this flag to check whether the SPI CH.0 is busy or idle.
In slave mode, the SPBSY flag indicates the SPI CH.0 slave select signal (#SPI_SSI0 pin) status; it goes 1 when this SPI CH.0 is selected as a slave or goes 0 when this SPI CH.0 is deselected.

Data receive control

In master mode, write dummy data to the SPI_TXD0 register (0x4322). Writing to the SPI_TXD0 register is used as the trigger for data receiving as well as starting data transmission. Also actual data to be transmitted can be written as the SPI CH.0 performs data transmission and reception simultaneously.
The SPI CH.0 starts output of the SPI clock from the SPI_SCK0 pin.

In slave mode, the SPI CH.0 waits for clock input from the SPI_SCK0 pin. When receiving data in slave mode without any data transmission, it is not necessary to write data to the SPI_TXD0 register. The receive process activates by the clock input from the master device. When performing data transmission and reception simultaneously, the transmit data should be written to the SPI_TXD0 register before a clock is input.

The data bits are fetched in the shift register one by one at the rising or falling edge of the clock configured with CPHA (D3/SPI_CTL0 register) and CPOL (D2/SPI_CTL0 register) (see Figure VI.3.4.1). The MSB of data is received first.
When eight data bits are received in the shift register, the received data is loaded into the receive data buffer.

The received data in the buffer can be read from the SPI_RXD0 register (0x4324).

* **SPI_RXD0**: SPI CH.0 Receive Data Register (0x4324)

The SPI CH.0 provides the SPRBF flag (D1/SPI_ST0 register) for data receive control.

* **SPRBF**: Receive Data Buffer Full Flag Bit in the SPI CH.0 Status (SPI_ST0) Register (D1/0x4320)

The SPRBF flag indicates the receive data buffer status; it goes 1 when the data received in the shift register is loaded to the receive data buffer to indicate that the receive data can be read and returns to 0 when the data in the receive data buffer is read out from the SPI_RXD0 register. An interrupt can be generated when this flag goes 1 (see Section VI.3.6). Use this interrupt or read the SPRBF flag to check that the receive data buffer contains valid data when reading received data. Although the receive data buffer size is one byte, the previous received data can be maintained while the next data is being received as the shift register is separately provided. However, be sure to read the receive data before the next data has been received. If the next data is received before the previous received data in the receive data buffer has been read, the previous received data is overwritten with the new data.

In master mode, the SPBSY flag that indicates the shift register status can be used as in data transmission.

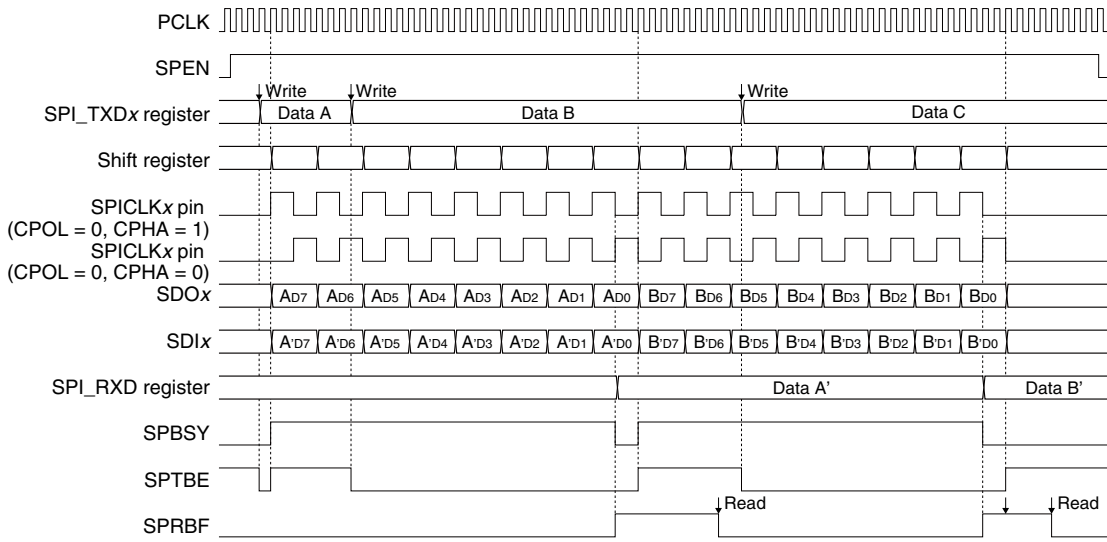


Figure VI.3.5.1 Data Transmission/Receiving Timing Chart (MSB first)

Disabling data transmission/reception

After data transfer (both transmission and reception) has finished, write 0 to the SPEN bit to disable data transmission/reception.

Always make sure that the SPTBE flag is 1 and SPBSY flag is 0 before data transmission/reception is disabled. The data being transferred cannot be guaranteed if SPEN is set to 0 during transmitting/receiving.

VI.3.6 SPI CH.0 Interrupt

The SPI CH.0 can generate the following two types of interrupts:
• Transmit buffer empty interrupt
• Receive buffer full interrupt

The SPI CH.0 has one interrupt signal to be output to the interrupt controller (ITC) and it is shared with the two causes of interrupt. To determine the cause of interrupt that has occurred, read the status flags.

Transmit buffer empty interrupt

Set the SPTIE bit (D4/SPI_CTL0 register) to 1 when using this interrupt. If SPTIE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **SPTIE**: Transmit Data Buffer Empty Interrupt Enable Bit in the SPI CH.0 Control (SPI_CTL0) Register (D4/0x4326)

When the transmit data set in the transmit data buffer is transferred to the shift register, the SPI CH.0 sets the SPTBE bit (D0/SPI_ST0 register) to 1 to indicate that the transmit data buffer is empty. At the same time, the SPI CH.0 outputs an interrupt request pulse to the ITC if the transmit buffer empty interrupt has been enabled (SPTIE = 1).

∗ **SPTBE**: Transmit Data Buffer Empty Flag Bit in the SPI CH.0 Status (SPI_ST0) Register (D0/0x4320)

If other interrupt conditions are satisfied, an interrupt is generated.
The SPI CH.0 interrupt handler routine should read the SPTBE flag to check if the interrupt has occurred due to a transmit buffer empty or another cause. When SPTBE = 1, the SPI CH.0 interrupt handler routine can write the next transmit data to the transmit data buffer.

Receive buffer full interrupt

Set the SPRIE bit (D5/SPI_CTL0 register) to 1 when using this interrupt. If SPRIE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **SPRIE**: Receive Data Buffer Full Interrupt Enable Bit in the SPI CH.0 Control (SPI_CTL0) Register (D5/0x4326)

When data received in the shift register is loaded to the receive data buffer, the SPI CH.0 sets the SPRBF bit (D1/SPI_ST0 register) to 1 to indicate that the received data buffer is full. At the same time, the SPI CH.0 outputs an interrupt request pulse to the ITC if the receive buffer full interrupt has been enabled (SPRIE = 1).

∗ **SPRBF**: Receive Data Buffer Full Flag Bit in the SPI CH.0 Status (SPI_ST0) Register (D1/0x4320)

If other interrupt conditions are satisfied, an interrupt is generated.
The SPI CH.0 interrupt handler routine should read the SPRBF flag to check if the interrupt has occurred due to a receive buffer full or another cause. When SPRBF = 1, the SPI CH.0 interrupt handler routine can read the received data from the receive data buffer.

ITC registers for SPI CH.0 interrupts

The following shows the control bits of the ITC provided for the SPI CH.0:

Interrupt flag
∗ **IIFT6**: SPI CH.0 Interrupt Flag Bit in the Interrupt Flag (ITC_IFLG) Register (D14/0x4300)

Interrupt enable bit
∗ **IIEN6**: SPI CH.0 Interrupt Enable Bit in the Interrupt Enable (ITC_EN) Register (D14/0x4302)

Interrupt level setup bits
∗ **IILV6[2:0]**: SPI CH.0 Interrupt Level Bits in the Internal Interrupt Level Setup (ITC_ILV3) Register 3 (D[2:0]/0x4314)

When the SPI CH.0 outputs an interrupt request pulse, the interrupt flag IIFT6 is set to 1.
If the interrupt enable bit IIEN6 has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the SPI CH.0 interrupt, set the IIEN6 bit to 0.
The IIFT6 flag is always set to 1 by the SPI CH.0 interrupt request pulse, regardless of how the IIEN6 bit is set (even when set to 0).
The interrupt level setup bits IILV6[2:0] set the interrupt level (0 to 7) of the SPI CH.0 interrupt.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The SPI CH.0 interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section VI.1, "Interrupt Controller (ITC)."

Interrupt vector

The following shows the vector number and vector address for the SPI CH.0 interrupt:

Vector number: 18 (0x12)
Vector address: TTBR + 0x48

VI.3.7 Details of Control Registers

Table VI.3.7.1 List of SPI CH.0 Registers

Address	Register name		Function
0x4320	SPI_ST0	SPI CH.0 Status Register	Indicates transfer and buffer statuses.
0x4322	SPI_TXD0	SPI CH.0 Transmit Data Register	Transmit data
0x4324	SPI_RXD0	SPI CH.0 Receive Data Register	Receive data
0x4326	SPI_CTL0	SPI CH.0 Control Register	Sets the SPI CH.0 mode and enables data transfer.

The following describes each SPI CH.0 register. These are all 16-bit registers.

Notes: • When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

• Be sure to use 16-bit access instructions for reading/writing from/to the SPI CH.0 registers. The SPI CH.0 registers do not allow reading/writing using 32-bit and 8-bit access instructions.

0x4320: SPI CH.0 Status Register (SPI_ST0)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SPI CH.0 Status Register (SPI_ST0)	0x4320 (16 bits)	D15–3	–	reserved	–				–	–	0 when being read.
		D2	SPBSY	Transfer busy flag (master)	1	Busy	0	Idle	0	R	
				ss signal low flag (slave)	1	ss = L	0	ss = H			
		D1	SPRBF	Receive data buffer full flag	1	Full	0	Not full	0	R	
		D0	SPTBE	Transmit data buffer empty flag	1	Empty	0	Not empty	1	R	

D[15:3] Reserved

D2 SPBSY: Transfer Busy Flag (Master)/ss Signal Low Flag (Slave) Bit

Master mode

Indicates the SPI CH.0 transmit/receive operation status.

1 (R): Busy

0 (R): Idle (default)

SPBSY is set to 1 when the SPI CH.0 starts data transmission/reception in master mode and stays 1 while data transmission/reception is in progress. SPBSY is reset to 0 upon completion of the transmit/receive operation.

Slave mode

Indicates the slave select (#SPI_SSI0) signal status.

1 (R): Low level (SPI CH.0 is selected)

0 (R): High level (SPI CH.0 is deselected) (default)

SPBSY is set to 1 when the master device activates the #SPI_SSI0 signal to select this SPI CH.0 (slave device), and is reset to 0 when the master device negates the #SPI_SSI0 signal to deselect this SPI CH.0.

D1 SPRBF: Receive Data Buffer Full Flag Bit

Indicates the receive data buffer status.

1 (R): Full

0 (R): Not full (default)

SPRBF is set to 1 when the data received in the shift register is loaded to the receive data buffer (receive operation completed), indicating that the received data can be read out. This bit is reset to 0 when the data is read out from the SPI_RXD0 register (0x4324).

D0 SPTBE: Transmit Data Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Empty (default)

0 (R): Not empty

SPTBE is reset to 0 when transmit data is written to the SPI_TXD0 register (transmit data buffer, 0x4322) and is set to 1 when the written data is transferred to the shift register (transmit operation started).

Transmit data should be written to the SPI_TXD0 register when this bit = 1.

0x4322: SPI CH.0 Transmit Data Register (SPI_TXD0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SPI CH.0 Transmit Data Register (SPI_TXD0)	**0x4322** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**SPTDB[7:0]**	SPI CH.0 transmit data buffer SPTDB7 = MSB SPTDB0 = LSB	0x0 to 0xff	0x0	R/W	

D[15:8] Reserved

D[7:0] SPTDB[7:0]: SPI CH.0 Transmit Data Buffer Bits

Set transmit data to be written to the transmit data buffer. (Default: 0x0)

In master mode, data transmission begins by writing data to this register. In slave mode, the register contents are transferred to the shift register to start data transmission when a clock is input from the master device.

SPTBE (D0/SPI_ST0 register) is set to 1 (empty) when the data is transferred to the shift register. At the same time, a cause of transmit data buffer empty interrupt occurs. The next transmit data can be written to the register at any time thereafter, even when the SPI CH.0 is sending data.

The serial-converted data is output from the SPI_SDO0 pin beginning with the MSB, in which the bits set to 1 are output as high-level signals and those set to 0 output as low-level signals.

Note: Make sure that SPEN is set to 1 before writing data to the SPI_TXD register to start data transmission/reception.

0x4324: SPI CH.0 Receive Data Register (SPI_RXD0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SPI CH.0 Receive Data Register (SPI_RXD0)	**0x4324** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	**SPRDB[7:0]**	SPI CH.0 receive data buffer SPRDB7 = MSB SPRDB0 = LSB	0x0 to 0xff	0x0	R	

D[15:8] Reserved

D[7:0] SPRDB[7:0]: SPI CH.0 Receive Data Buffer Bits

Stores received data. (Default: 0x0)

When a receive operation is completed and the data received in the shift register is loaded to the receive data buffer, SPRBF (D1/SPI_ST0 register) is set to 1 (buffer full). At the same time, a cause of receive data buffer full interrupt occurs. Thereafter, the data can be read out at any time before a receive operation for the next data is completed.

If the next data receive operation is completed before this register is read out, the data in it is overwritten with the newly received data.

The serial data input from the SPI_SDI0 pin is converted into parallel data beginning with the MSB, with the high-level signals changed to 1s and the low-level signals changed to 0s. The resulting data is stored in this register.

SPI_RXD0 is a read-only register, so no data can be written to it.

0x4326: SPI CH.0 Control Register (SPI_CTL0)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SPI CH.0 Control Register (SPI_CTL0)	0x4326 (16 bits)	D15–6	–	reserved	–				–	–	0 when being read.
		D5	SPRIE	Receive data buffer full int. enable	1	Enable	0	Disable	0	R/W	
		D4	SPTIE	Transmit data buffer empty int. enable	1	Enable	0	Disable	0	R/W	
		D3	CPHA	Clock phase select	1	Data out	0	Data in	0	R/W	These bits must be set before setting SPEN to 1.
		D2	CPOL	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	MSSL	Master/slave mode select	1	Master	0	Slave	0	R/W	
		D0	SPEN	SPI CH.0 enable	1	Enable	0	Disable	0	R/W	

D[15:6] Reserved

D5 SPRIE: Receive Data Buffer Full Interrupt Enable Bit

Enables/disables SPI CH.0 interrupt caused by receive data buffer full.
1 (R/W): Enable
0 (R/W): Disable (default)

When SPRIE is set to 1, SPI CH.0 (receive data buffer full) interrupt requests to the ITC are enabled. A receive data buffer full interrupt request occurs when the data received in the shift register is loaded to the receive data buffer (receive operation completed).
When SPRIE is set to 0, SPI CH.0 interrupts caused by receive data full are not generated.

D4 SPTIE: Transmit Data Buffer Empty Interrupt Enable Bit

Enables/disables SPI CH.0 interrupt caused by transmit data buffer empty.
1 (R/W): Enable
0 (R/W): Disable (default)

When SPTIE is set to 1, SPI CH.0 (transmit data buffer empty) interrupt requests to the ITC are enabled. A transmit data buffer empty interrupt request occurs when the data written to the transmit data buffer is transferred to the shift register (transmit operation started).
When SPTIE is set to 0, SPI CH.0 interrupts caused by transmit data buffer empty are not generated.

D3 CPHA: Clock Phase Select Bit

Selects the phase of the SPI clock. (Default: 0)
This bit controls the data transfer timing in conjunction with the CPOL (D2) bit (see Figure VI.3.7.1).

D2 CPOL: Clock Polarity Select Bit

Selects the polarity of the SPI clock.
1 (R/W): Active low
0 (R/W): Active high (default)

This bit controls the data transfer timing in conjunction with the CPHA (D3) bit (see Figure VI.3.7.1).

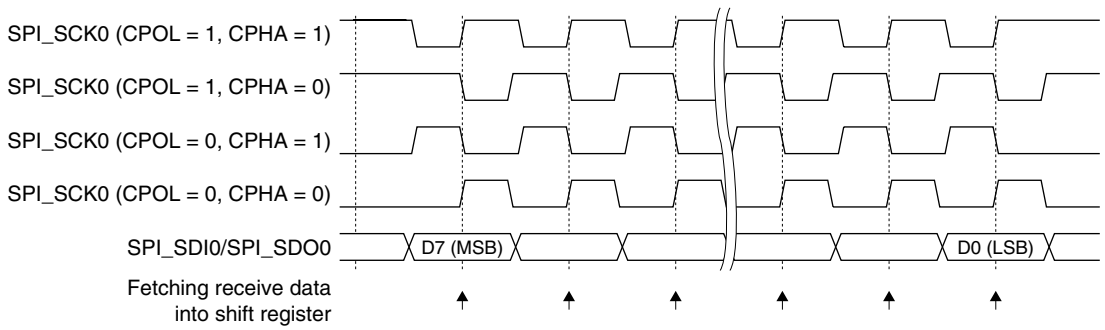


Figure VI.3.7.1 Clock and Data Transfer Timing

D1 MSSL: Master/Slave Mode Select Bit

Sets the SPI CH.0 in master or slave mode.
1 (R/W): Master mode
0 (R/W): Slave mode (default)

Setting MSSL to 1 selects master mode, and setting to 0 selects slave mode. In master mode, the SPI CH.0 performs data transfer using the clock generated by the CLG_T8S timer. In slave mode, the SPI CH.0 performs data transfer using a clock input from the master device.

D0 SPEN: SPI CH.0 Enable Bit

Enables/disables operation of the SPI CH.0.
1 (R/W): Enable
0 (R/W): Disable (default)

When SPEN is set to 1, the SPI CH.0 is activated and data transfer is enabled.
When SPEN is set to 0, the SPI CH.0 goes off.

Note: Make sure that the SPEN bit is 0 before setting the CPHA, CPOL, and MSSL bits.

VI.3.8 Precautions

- Be sure to use 16-bit access instructions for reading/writing from/to the SPI CH.0 registers (0x4320 to 0x4326). The SPI CH.0 registers do not allow reading/writing using 32-bit and 8-bit access instructions.
- Do not access the SPI_CTL0 register (0x4326), while the SPBSY flag (D2/SPI_ST0 register) is set to 1 (during data transfer).

  * **SPBSY**: Transfer Busy Flag (Master)/ss Signal Low Flag (Slave) Bit in the SPI CH.0 Status (SPI_ST0) Register (D2/0x4320)

VI.4 Extended SPI (SPI CH.1)

VI.4.1 Configuration of the SPI CH.1

The S1C17501 equipped with a synchronous serial interface module (hereafter SPI CH.1). The SPI CH.1 supports both master and slave modes and performs 8-bit serial data transfer. Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
The SPI CH.1 includes a transmit data buffer and a receive data buffer separately from the shift registers, and can generate two types of interrupts (transmit data buffer empty and receive data buffer full), this makes it possible to process continuous serial data transfers simply in an interrupt handler.
Figure VI.4.1.1 shows the structure of the SPI CH.1.

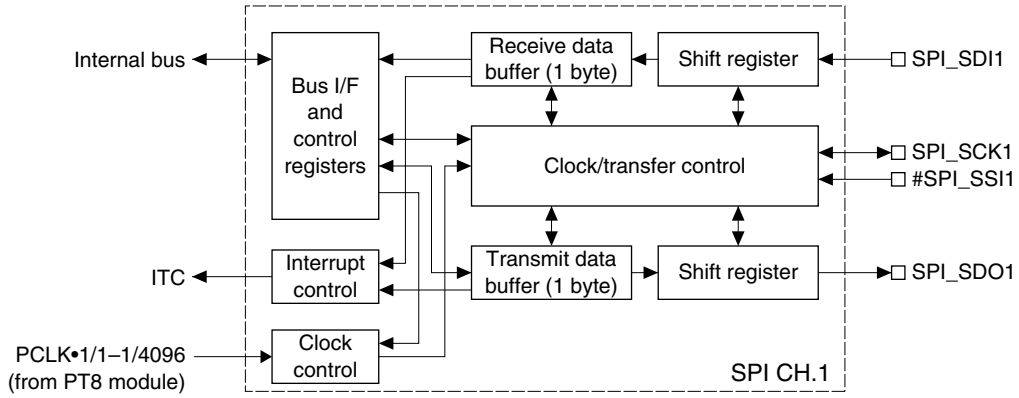


Figure VI.4.1.1 Structure of SPI CH.1

VI.4.2 SPI CH.1 I/O Pins

Table VI.4.2.1 lists the SPI CH.1 pins.

Table VI.4.2.1 List of SPI CH.1 Pins

Pin name	I/O	Size	Function
SPI_SDI1	I	1	SPI CH.1 data input pin This pin inputs serial data from the SPI bus.
SPI_SDO1	O	1	SPI CH.1 data output pin This pin outputs serial data to the SPI bus.
SPI_SCK1	I/O	1	SPI CH.1 external clock input/output pin This pin outputs the SPI clock when the SPI CH.1 is in master mode. This pin inputs an external clock when the SPI CH.1 is in slave mode.
#SPI_SSI1	I	1	SPI CH.1 slave select signal (active low) input pin A low level input to this pin selects this SPI CH.1 device in slave mode.

The SPI CH.1 input/output pins (SPI_SDI1, SPI_SDO1, SPI_SCK1, #SPI_SSI1) are shared with the I/O ports and they are initialized as general-purpose I/O port pins by default. Before using these pins for the SPI CH.1, the pin functions must be switched using the Port Function Select registers.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.4.3 SPI Clock

In master mode, the SPI CH.1 uses the internal clock output from the prescaler in the 8-bit timer (PT8) module. To operate the PT8 prescaler for SPI CH.1, the operating clocks for both the PT8 and ESPI modules must be supplied from the CMU by setting the control bits shown below to 1.

- **PT8_CLK_EN**: 8-bit Programmable Timer Clock Control Bit in the Gated Clock Control 1 (CMU_GATEDCLK1) Register (D1/0x4907)
- **SPI_CLK_EN**: SPI CH.1 Module Clock Control Bit in the Gated Clock Control 2 (CMU_GATEDCLK2) Register (D5/0x4908)

The PT8 prescaler divides PCLK (with the same frequency as the system clock) by 1 to 4096 to generate 13 clocks. Select one of the prescaler output clocks using SPI_CLK[3:0] (D[3:0]/SPI_CLK1 register).

- **SPI_CLK[3:0]**: SPI CH.1 Clock Division Ratio Selection Bits in the SPI CH.1 Clock Control (SPI_CLK1) Register (D[3:0]/0x5708)

Table VI.4.3.1 Selecting the SPI Clock

SPI_CLK[3:0]	Prescaler output clock	SPI_CLK[3:0]	Prescaler output clock
0xf	Reserved	0x7	PCLK•1/128
0xe	Reserved	0x6	PCLK•1/64
0xd	Reserved	0x5	PCLK•1/32
0xc	PCLK•1/4096	0x4	PCLK•1/16
0xb	PCLK•1/2048	0x3	PCLK•1/8
0xa	PCLK•1/1024	0x2	PCLK•1/4
0x9	PCLK•1/512	0x1	PCLK•1/2
0x8	PCLK•1/256	0x0	PCLK•1/1

(Default: 0x0)

The selected clock is input to the SPI CH.1 by setting SPI_CKE (D4/SPI_CLK1 register) to 1.

- **SPI_CKE**: SPI CH.1 Clock Enable Bit in the SPI CH.1 Clock Control (SPI_CLK1) Register (D4/0x5708)

In slave mode, the SPI CH.1 inputs the SPI clock from the SPI_SCK1 pin. Since the internal circuit operates with the internal PCLK clock, the input clock is differentiated and used to sync with the PCLK clock.

Note: The duty ratio of the clock input via the SPI_SCK1 pin must be 50%.

VI.4.4 Setting the Data Transfer Conditions

The SPI CH.1 can be set in master or slave mode and the SPI clock polarity and phase can be set using the SPI_CTL1 register.
The data length is fixed at eight bits.

Note: Make sure that the SPI CH.1 is disabled (SPEN/SPI_CTL1 register = 0) before selecting master/slave mode and setting the clock conditions.

∗ **SPEN**: SPI CH.1 Enable Bit in the SPI CH.1 Control (SPI_CTL1) Register (D0/0x5706)

Selecting master/slave mode

Use MSSL (D1/SPI_CTL1 register) to select whether the SPI CH.1 is set in master mode or slave mode. Setting MSSL to 1 selects master mode, and setting to 0 (default) selects slave mode. In master mode, the SPI CH.1 performs data transfer using the internal clock. In slave mode, the SPI CH.1 performs data transfer using a clock input from the master device.

∗ **MSSL**: Master/Slave Mode Select Bit in the SPI CH.1 Control (SPI_CTL1) Register (D1/0x5706)

Setting the SPI clock polarity and phase

Use CPOL (D2/SPI_CTL1 register) to select the SPI clock polarity. The SPI clock is configured as active low when CPOL is set to 1 or active high when CPOL is set to 0 (default).

∗ **CPOL**: Clock Polarity Select Bit in the SPI CH.1 Control (SPI_CTL1) Register (D2/0x5706)

The SPI clock phase is selected with CPHA (D3/SPI_CTL1 register).

∗ **CPHA**: Clock Phase Select Bit in the SPI CH.1 Control (SPI_CTL1) Register (D3/0x5706)

Setting these control bits determines the transfer timing as in the figure shown below.

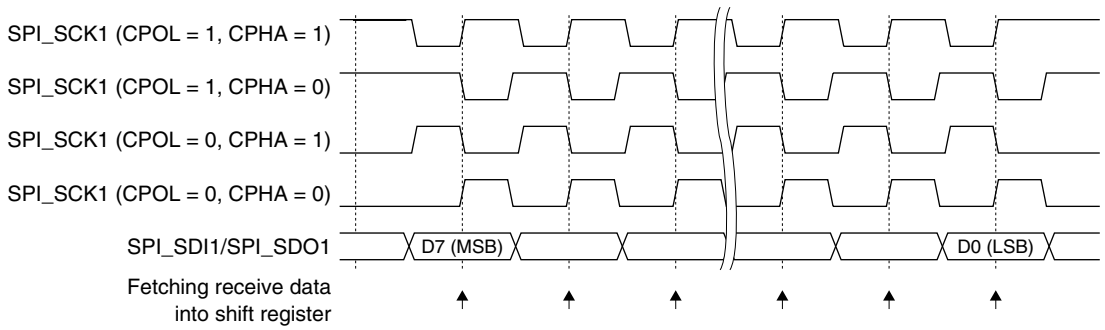


Figure VI.4.4.1 Clock and Data Transfer Timing

VI.4.5 Data Transmit/Receive Control

Before starting data transfer, set up the conditions as shown below.
(1) Set up the PT8 prescaler to output the SPI clock. See Section VI.4.3.
(2) Select either master or slave mode. See Section VI.4.4.
(3) Set up the clock conditions. See Section VI.4.4.
(4) Set up the interrupt conditions if the SPI CH.1 interrupt is used. See Section VI.4.6.

Note: Make sure that the SPI CH.1 is disabled (SPEN/SPI_CTL1 register = 0) before setting the conditions above.

* **SPEN**: SPI CH.1 Enable Bit in the SPI CH.1 Control (SPI_CTL1) Register (D0/0x5706)

Enabling data transmission/reception

First, set the SPEN bit (D0/SPI_CTL1 register) to 1 to enable SPI CH.1 operation. This puts the SPI CH.1 in ready-to-transmit/receive status and enables clock input/output.

Note: Do not set the SPEN bit to 0 while the SPI CH.1 is transmitting/receiving data.

Data transmit control

To start transmission, write transmit data to the SPI_TXD1 register (0x5702).

* **SPI_TXD1**: SPI CH.1 Transmit Data Register (0x5702)

Data is written to the transmit data buffer and the SPI CH.1 starts data transmission.
The buffered data is sent to the shift register for transmission. In master mode, the SPI CH.1 starts outputting the clock from the SPI_SCK1 pin. In slave mode, the SPI CH.1 waits for clock input from the SPI_SCK1 pin. The data bits in the shift register are shifted one by one at the rising or falling edge of the clock configured with CPHA (D3/SPI_CTL1 register) and CPOL (D2/SPI_CTL1 register) (see Figure VI.4.4.1), and are output from the SPI_SDO1 pin. The MSB of data is transmitted first.

* **CPHA**: Clock Phase Select Bit in the SPI CH.1 Control (SPI_CTL1) Register (D3/0x5706)
* **CPOL**: Clock Polarity Select Bit in the SPI CH.1 Control (SPI_CTL1) Register (D2/0x5706)

The SPI CH.1 provides two status flags for data transmit control, SPTBE (D0/SPI_ST1 register) and SPBSY (D2/SPI_ST1 register).

* **SPTBE**: Transmit Data Buffer Empty Flag Bit in the SPI CH.1 Status (SPI_ST1) Register (D0/0x5700)
* **SPBSY**: Transfer Busy Flag (Master)/ss Signal Low Flag (Slave) Bit in the SPI CH.1 Status (SPI_ST1) Register (D2/0x5700)

The SPTBE flag indicates the transmit data buffer status; it goes 0 when the application program writes data to the SPI_TXD1 register (transmit data buffer) and returns to 1 when the data in the transmit data buffer is sent to the shift register for transmitting. An interrupt can be generated when this flag goes 1 (see Section VI.4.6). Use this interrupt or read the SPTBE flag to check that the transmit data buffer becomes empty when transmitting the next data. Although the transmit data buffer size is one byte, transmit data can be written while the previous data is being transmitted as the shift register is separately provided. However, make sure that the transmit data buffer is empty before writing transmit data. If data is written when the SPTBE flag is 0, the previous transmit data in the transmit data buffer is overwritten with the new data.

In master mode, the SPBSY flag indicates the shift register status; it goes 1 when transmit data is loaded from the transmit data buffer and returns to 0 upon completion of data transmission. Read this flag to check whether the SPI CH.1 is busy or idle.
In slave mode, the SPBSY flag indicates the SPI CH.1 slave select signal (#SPI_SSI1 pin) status; it goes 1 when this SPI CH.1 is selected as a slave or goes 0 when this SPI CH.1 is deselected.

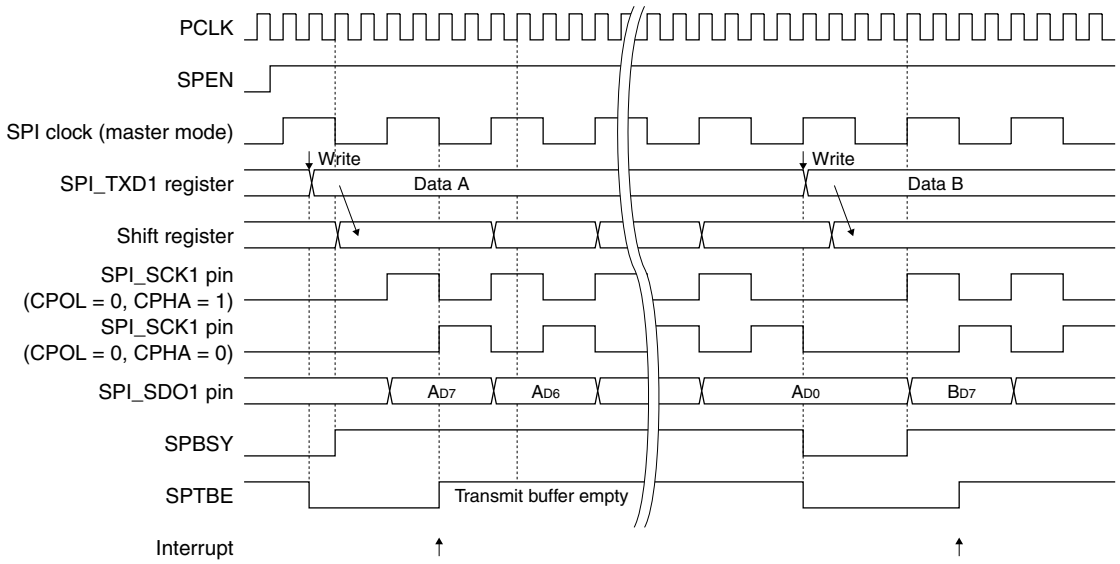


Figure VI.4.5.1 Data Transmit Timing Chart

Data receive control

In master mode, write dummy data to the SPI_TXD1 register (0x5702). Writing to the SPI_TXD1 register is used as the trigger for data receiving as well as starting data transmission. Also actual data to be transmitted can be written as the SPI CH.1 performs data transmission and reception simultaneously.
The SPI CH.1 starts output of the SPI clock from the SPI_SCK1 pin.

In slave mode, the SPI CH.1 waits for clock input from the SPI_SCK1 pin. When receiving data in slave mode without any data transmission, it is not necessary to write data to the SPI_TXD1 register. The receive process activates by the clock input from the master device. When performing data transmission and reception simultaneously, the transmit data should be written to the SPI_TXD1 register before a clock is input.

The data bits are fetched in the shift register one by one at the rising or falling edge of the clock configured with CPHA (D3/SPI_CTL1 register) and CPOL (D2/SPI_CTL1 register) (see Figure VI.4.4.1). The MSB of data is received first.
When eight data bits are received in the shift register, the received data is loaded into the receive data buffer.

The received data in the buffer can be read from the SPI_RXD1 register (0x5704).

* **SPI_RXD1**: SPI CH.1 Receive Data Register (0x5704)

The SPI CH.1 provides the SPRBF flag (D1/SPI_ST1 register) for data receive control.

* **SPRBF**: Receive Data Buffer Full Flag Bit in the SPI CH.1 Status (SPI_ST1) Register (D1/0x5700)

The SPRBF flag indicates the receive data buffer status; it goes 1 when the data received in the shift register is loaded to the receive data buffer to indicate that the receive data can be read and returns to 0 when the data in the receive data buffer is read out from the SPI_RXD1 register. An interrupt can be generated when this flag goes 1 (see Section VI.4.6). Use this interrupt or read the SPRBF flag to check that the receive data buffer contains valid data when reading received data. Although the receive data buffer size is one byte, the previous received data can be maintained while the next data is being received as the shift register is separately provided. However, be sure to read the receive data before the next data has been received. If the next data is received before the previous received data in the receive data buffer has been read, the previous received data is overwritten with the new data.

In master mode, the SPBSY flag that indicates the shift register status can be used as in data transmission.

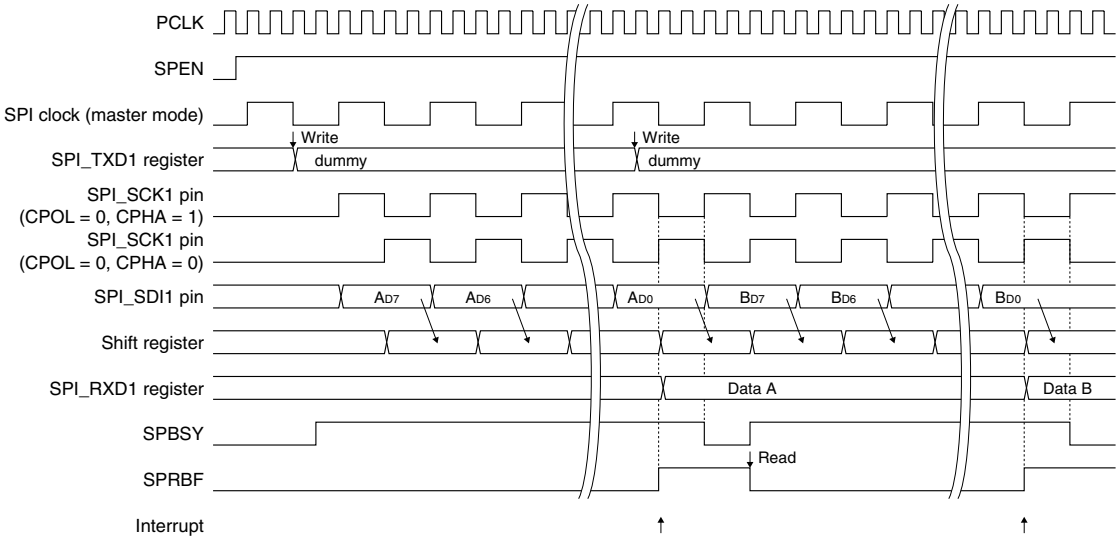


Figure VI.4.5.2 Data Receive Timing Chart

Disabling data transmission/reception

After data transfer (both transmission and reception) has finished, write 0 to the SPEN bit to disable data transmission/reception.

Always make sure that the SPTBE flag is 1 and SPRBF flag is 0 before data transmission/reception is disabled.

When the SPEN bit is set to 0, the transmit and receive data buffers are placed in empty status (data is cleared if any remains). Furthermore, the data being transferred cannot be guaranteed if SPEN is set to 0 during transmitting/receiving.

VI.4.6 SPI CH.1 Interrupt

The SPI CH.1 can generate the following two types of interrupts:
• Transmit buffer empty interrupt
• Receive buffer full interrupt

The SPI CH.1 has one interrupt signal to be output to the interrupt controller (ITC) and it is shared with the two causes of interrupt. To determine the cause of interrupt that has occurred, read the status flags.

Transmit buffer empty interrupt

Set the SPTIE bit (D4/SPI_CTL1 register) to 1 when using this interrupt. If SPTIE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **SPTIE**: Transmit Data Buffer Empty Interrupt Enable Bit in the SPI CH.1 Control (SPI_CTL1) Register (D4/0x5706)

When the transmit data set in the transmit data buffer is transferred to the shift register, the SPI CH.1 sets the SPTBE bit (D0/SPI_ST1 register) to 1 to indicate that the transmit data buffer is empty. At the same time, the SPI CH.1 outputs an interrupt request pulse to the ITC if the transmit buffer empty interrupt has been enabled (SPTIE = 1).

∗ **SPTBE**: Transmit Data Buffer Empty Flag Bit in the SPI CH.1 Status (SPI_ST1) Register (D0/0x5700)

If other interrupt conditions are satisfied, an interrupt is generated.
The SPI CH.1 interrupt handler routine should read the SPTBE flag to check if the interrupt has occurred due to a transmit buffer empty or another cause. When SPTBE = 1, the SPI CH.1 interrupt handler routine can write the next transmit data to the transmit data buffer.

Receive buffer full interrupt

Set the SPRIE bit (D5/SPI_CTL1 register) to 1 when using this interrupt. If SPRIE is set to 0 (default), an interrupt request by this cause will not be sent to the ITC.

∗ **SPRIE**: Receive Data Buffer Full Interrupt Enable Bit in the SPI CH.1 Control (SPI_CTL1) Register (D5/0x5706)

When data received in the shift register is loaded to the receive data buffer, the SPI CH.1 sets the SPRBF bit (D1/SPI_ST1 register) to 1 to indicate that the received data buffer is full. At the same time, the SPI CH.1 outputs an interrupt request pulse to the ITC if the receive buffer full interrupt has been enabled (SPRIE = 1).

∗ **SPRBF**: Receive Data Buffer Full Flag Bit in the SPI CH.1 Status (SPI_ST1) Register (D1/0x5700)

If other interrupt conditions are satisfied, an interrupt is generated.
The SPI CH.1 interrupt handler routine should read the SPRBF flag to check if the interrupt has occurred due to a receive buffer full or another cause. When SPRBF = 1, the SPI CH.1 interrupt handler routine can read the received data from the receive data buffer.

ITC registers for SPI CH.1 interrupts

The following shows the control bits of the ITC provided for the SPI CH.1:

Interrupt flag
∗ **AIFT10**: SPI CH.1 Interrupt Flag Bit in the Additional Interrupt Flag (ITC_AIFLG) Register (D10/0x42e0)

Interrupt enable bit
∗ **AIEN10**: SPI CH.1 Interrupt Enable Bit in the Additional Interrupt Enable (ITC_AEN) Register (D10/0x42e2)

Interrupt level setup bits
∗ **AILV10[2:0]**: SPI CH.1 Interrupt Level Bits in the Additional Interrupt Level Setup (ITC_AILV5) Register 5 (D[2:0]/0x42f0)

When the SPI CH.1 outputs an interrupt request pulse, the interrupt flag AIFT10 is set to 1.
If the interrupt enable bit AIEN10 has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the SPI CH.1 interrupt, set the AIEN10 bit to 0.
The AIFT10 flag is always set to 1 by the SPI CH.1 interrupt request pulse, regardless of how the AIEN10 bit is set (even when set to 0).
The interrupt level setup bits AILV10[2:0] set the interrupt level (0 to 7) of the SPI CH.1 interrupt.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The SPI CH.1 interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, "Interrupt Controller (ITC)."

Interrupt vector

The following shows the vector number and vector address for the SPI CH.1 interrupt:

Vector number: 26 (0x1a)
Vector address: TTBR + 0x68

VI.4.7 Details of Control Registers

Table VI.4.7.1 List of SPI CH.1 Registers

Address	Register name		Function
0x5700	SPI_ST1	SPI CH.1 Status Register	Indicates transfer and buffer statuses.
0x5702	SPI_TXD1	SPI CH.1 Transmit Data Register	Transmit data
0x5704	SPI_RXD1	SPI CH.1 Receive Data Register	Receive data
0x5706	SPI_CTL1	SPI CH.1 Control Register	Sets the SPI CH.1 mode and enables data transfer.
0x5708	SPI_CLK1	SPI CH.1 Clock Control Register	Sets up the SPI clock.

The following describes each SPI CH.1 register. These are all 16-bit registers.

Notes: • When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

• Be sure to use 16-bit access instructions for reading/writing from/to the SPI CH.1 registers. The SPI CH.1 registers do not allow reading/writing using 32-bit and 8-bit access instructions.

0x5700: SPI CH.1 Status Register (SPI_ST1)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SPI CH.1 Status Register (SPI_ST1)	0x5700 (16 bits)	D15–3	–	reserved	–				–	–	0 when being read.
		D2	SPBSY	Transfer busy flag (master)	1	Busy	0	Idle	0	R	
				ss signal low flag (slave)	1	ss = L	0	ss = H			
		D1	SPRBF	Receive data buffer full flag	1	Full	0	Not full	0	R	
		D0	SPTBE	Transmit data buffer empty flag	1	Empty	0	Not empty	1	R	

D[15:3] Reserved

D2 SPBSY: Transfer Busy Flag (Master)/ss Signal Low Flag (Slave) Bit

Master mode

Indicates the SPI CH.1 transmit/receive operation status.

1 (R): Busy

0 (R): Idle (default)

SPBSY is set to 1 when the SPI CH.1 starts data transmission/reception in master mode and stays 1 while data transmission/reception is in progress. SPBSY is reset to 0 upon completion of the transmit/receive operation.

Slave mode

Indicates the slave select (#SPI_SSI1) signal status.

1 (R): Low level (SPI CH.1 is selected)

0 (R): High level (SPI CH.1 is deselected) (default)

SPBSY is set to 1 when the master device activates the #SPI_SSI1 signal to select this SPI CH.1 (slave device), and is reset to 0 when the master device negates the #SPI_SSI1 signal to deselect this SPI CH.1.

D1 SPRBF: Receive Data Buffer Full Flag Bit

Indicates the receive data buffer status.

1 (R): Full

0 (R): Not full (default)

SPRBF is set to 1 when the data received in the shift register is loaded to the receive data buffer (receive operation completed), indicating that the received data can be read out. This bit is reset to 0 when the data is read out from the SPI_RXD1 register (0x5704).

D0 SPTBE: Transmit Data Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Empty (default)

0 (R): Not empty

SPTBE is reset to 0 when transmit data is written to the SPI_TXD1 register (transmit data buffer, 0x5702) and is set to 1 when the written data is transferred to the shift register (transmit operation started).

Transmit data should be written to the SPI_TXD1 register when this bit = 1.

0x5702: SPI CH.1 Transmit Data Register (SPI_TXD1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SPI CH.1 Transmit Data Register (SPI_TXD1)	**0x5702** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**SPTDB[7:0]**	SPI CH.1 transmit data buffer SPTDB7 = MSB SPTDB0 = LSB	0x0 to 0xff	0x0	R/W	

D[15:8] Reserved

D[7:0] SPTDB[7:0]: SPI CH.1 Transmit Data Buffer Bits

Set transmit data to be written to the transmit data buffer. (Default: 0x0)

In master mode, data transmission begins by writing data to this register. In slave mode, the register contents are transferred to the shift register to start data transmission when a clock is input from the master device.

SPTBE (D0/SPI_ST1 register) is set to 1 (empty) when the data is transferred to the shift register. At the same time, a cause of transmit data buffer empty interrupt occurs. The next transmit data can be written to the register at any time thereafter, even when the SPI CH.1 is sending data.

The serial-converted data is output from the SPI_SDO1 pin beginning with the MSB, in which the bits set to 1 are output as high-level signals and those set to 0 output as low-level signals.

0x5704: SPI CH.1 Receive Data Register (SPI_RXD1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SPI CH.1 Receive Data Register (SPI_RXD1)	**0x5704** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	**SPRDB[7:0]**	SPI CH.1 receive data buffer SPRDB7 = MSB SPRDB0 = LSB	0x0 to 0xff	0x0	R	

D[15:8] Reserved

D[7:0] SPRDB[7:0]: SPI CH.1 Receive Data Buffer Bits

Stores received data. (Default: 0x0)

When a receive operation is completed and the data received in the shift register is loaded to the receive data buffer, SPRBF (D1/SPI_ST1 register) is set to 1 (buffer full). At the same time, a cause of receive data buffer full interrupt occurs. Thereafter, the data can be read out at any time before a receive operation for the next data is completed.

If the next data receive operation is completed before this register is read out, the data in it is overwritten with the newly received data.

The serial data input from the SPI_SDI1 pin is converted into parallel data beginning with the MSB, with the high-level signals changed to 1s and the low-level signals changed to 0s. The resulting data is stored in this register.

SPI_RXD1 is a read-only register, so no data can be written to it.

0x5706: SPI CH.1 Control Register (SPI_CTL1)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SPI CH.1 Control Register (SPI_CTL1)	0x5706 (16 bits)	D15–6	–	reserved	–				–	–	0 when being read.
		D5	SPRIE	Receive data buffer full int. enable	1	Enable	0	Disable	0	R/W	
		D4	SPTIE	Transmit data buffer empty int. enable	1	Enable	0	Disable	0	R/W	
		D3	CPHA	Clock phase select	1	Data out	0	Data in	0	R/W	These bits must be set before setting SPEN to 1.
		D2	CPOL	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	MSSL	Master/slave mode select	1	Master	0	Slave	0	R/W	
		D0	SPEN	SPI CH.1 enable	1	Enable	0	Disable	0	R/W	

D[15:6] Reserved

D5 SPRIE: Receive Data Buffer Full Interrupt Enable Bit

Enables/disables SPI CH.1 interrupt caused by receive data buffer full.
1 (R/W): Enable
0 (R/W): Disable (default)

When SPRIE is set to 1, SPI CH.1 (receive data buffer full) interrupt requests to the ITC are enabled. A receive data buffer full interrupt request occurs when the data received in the shift register is loaded to the receive data buffer (receive operation completed).
When SPRIE is set to 0, SPI CH.1 interrupts caused by receive data full are not generated.

D4 SPTIE: Transmit Data Buffer Empty Interrupt Enable Bit

Enables/disables SPI CH.1 interrupt caused by transmit data buffer empty.
1 (R/W): Enable
0 (R/W): Disable (default)

When SPTIE is set to 1, SPI CH.1 (transmit data buffer empty) interrupt requests to the ITC are enabled. A transmit data buffer empty interrupt request occurs when the data written to the transmit data buffer is transferred to the shift register (transmit operation started).
When SPTIE is set to 0, SPI CH.1 interrupts caused by transmit data buffer empty are not generated.

D3 CPHA: Clock Phase Select Bit

Selects the phase of the SPI clock. (Default: 0)
This bit controls the data transfer timing in conjunction with the CPOL (D2) bit (see Figure VI.4.7.1).

D2 CPOL: Clock Polarity Select Bit

Selects the polarity of the SPI clock.
1 (R/W): Active low
0 (R/W): Active high (default)

This bit controls the data transfer timing in conjunction with the CPHA (D3) bit (see Figure VI.4.7.1).

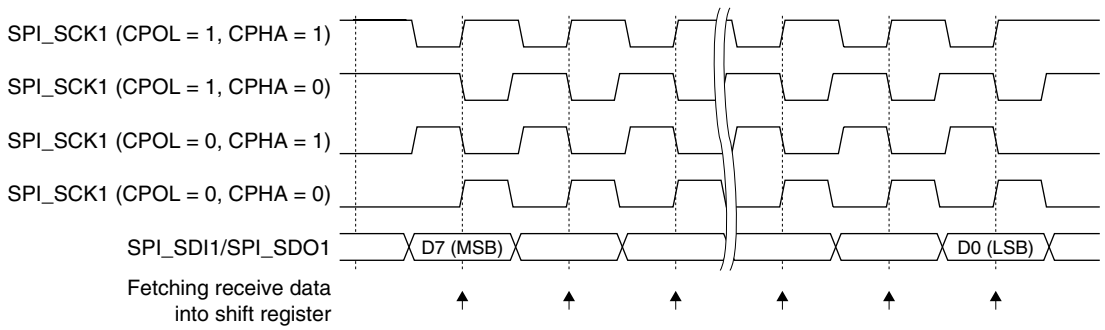


Figure VI.4.7.1 Clock and Data Transfer Timing

D1 MSSL: Master/Slave Mode Select Bit

Sets the SPI CH.1 in master or slave mode.
1 (R/W): Master mode
0 (R/W): Slave mode (default)

Setting MSSL to 1 selects master mode, and setting to 0 selects slave mode. In master mode, the SPI CH.1 performs data transfer using the clock generated by the PT8 prescaler. In slave mode, the SPI CH.1 performs data transfer using a clock input from the master device.

D0 SPEN: SPI CH.1 Enable Bit

Enables/disables operation of the SPI CH.1.
1 (R/W): Enable
0 (R/W): Disable (default)

When SPEN is set to 1, the SPI CH.1 is activated and data transfer is enabled.
When SPEN is set to 0, the SPI CH.1 goes off.

Note: Make sure that the SPEN bit is 0 before setting the CPHA, CPOL, and MSSL bits.

0x5708: SPI CH.1 Clock Control Register (SPI_CLK1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SPI CH.1 Clock Control Register (SPI_CLK1)	0x5708 (16 bits)	D15–5	–	reserved	–	–	–	0 when being read.
		D4	SPI_CKE	SPI CH.1 clock enable	1 Enable 0 Disable	0	R/W	
		D3–0	SPI_CLK [3:0]	SPI CH.1 clock division ratio selection (Prescaler output clock)	SPI_CLK[3:0] / Clock: 0xf–0xd reserved; 0xc PCLK•1/4096; 0xb PCLK•1/2048; 0xa PCLK•1/1024; 0x9 PCLK•1/512; 0x8 PCLK•1/256; 0x7 PCLK•1/128; 0x6 PCLK•1/64; 0x5 PCLK•1/32; 0x4 PCLK•1/16; 0x3 PCLK•1/8; 0x2 PCLK•1/4; 0x1 PCLK•1/2; 0x0 PCLK•1/1	0x0	R/W	

D[15:5] Reserved

D4 SPI_CKE: SPI CH.1 Clock Enable Bit

Enables the internally generated SPI clock input to the SPI CH.1.
1 (R/W): Enable
0 (R/W): Disable (default)

Write 1 to this bit before the SPI CH.1 can operate in master mode.

D[3:0] SPI_CLK[3:0]: SPI CH.1 Clock Division Ratio Selection Bits

These bits select the SPI clock for the SPI CH.1 from 13 PT8 prescaler output clocks.

Table VI.4.7.2 Selecting the SPI Clock

SPI_CLK[3:0]	Prescaler output clock	SPI_CLK[3:0]	Prescaler output clock
0xf	Reserved	0x7	PCLK•1/128
0xe	Reserved	0x6	PCLK•1/64
0xd	Reserved	0x5	PCLK•1/32
0xc	PCLK•1/4096	0x4	PCLK•1/16
0xb	PCLK•1/2048	0x3	PCLK•1/8
0xa	PCLK•1/1024	0x2	PCLK•1/4
0x9	PCLK•1/512	0x1	PCLK•1/2
0x8	PCLK•1/256	0x0	PCLK•1/1

(Default: 0x0)

VI.4.8 Precautions

- When using SPI CH.1 in master mode, the operating clocks for both the PT8 and ESPI modules must be supplied from the CMU, since the PT8 module includes the prescaler for generating the SPI clock for the SPI CH.1.
- Be sure to use 16-bit access instructions for reading/writing from/to the SPI CH.1 registers (0x5700 to 0x5708). The SPI CH.1 registers do not allow reading/writing using 32-bit and 8-bit access instructions.
- Do not access the SPI_CTL1 register (0x5706), while the SPBSY flag (D2/SPI_ST1 register) is set to 1 (during data transfer).

  * **SPBSY**: Transfer Busy Flag (Master)/ss Signal Low Flag (Slave) Bit in the SPI CH.1 Status (SPI_ST1) Register (D2/0x5700)

VI.5 I2S

VI.5.1 Overview of the I2S Module

The S1C17501 has a built-in bi-directional I2S module that inputs/outputs PCM data in the I2S (Inter-IC Sound) format. An audio input/output circuit can be simply configured by connecting external devices such as an audio DAC and ADC to the I2S bus.
The following shows the features of the I2S module:

Output channel (CH.0)

- Operates as an I2S master device.
- Generates the bit clock, word-select clock, and master clock.
- 16-bit or 24-bit resolution is selectable for PCM data to be output.
- A 24-byte transmit FIFO (24 bits × 2 channels × 4) is included.
- Stereo, mono (L and R), and mute modes are software selectable.
- FIFO data empty (half empty, whole empty, or one empty) can issue an interrupt request.

Input channel (CH.1)

- Operates as an I2S slave device.
- Supports 16-bit resolution for PCM data to be input.
- A 16-byte receive FIFO (16 bits × 2 channels × 4) is included.
- FIFO data full (half full, whole full, or one data) can issue an interrupt request.
- Supports bypass mode to send the input clock and data directly to the output channel.

Data supports

- Clock polarity is software configurable.
- Data shift direction (MSB first/LSB first) is software selectable.
- Supports I2S mode, left justified mode, and right justified mode.

Figure VI.5.1.1 shows the structure of the I2S module.

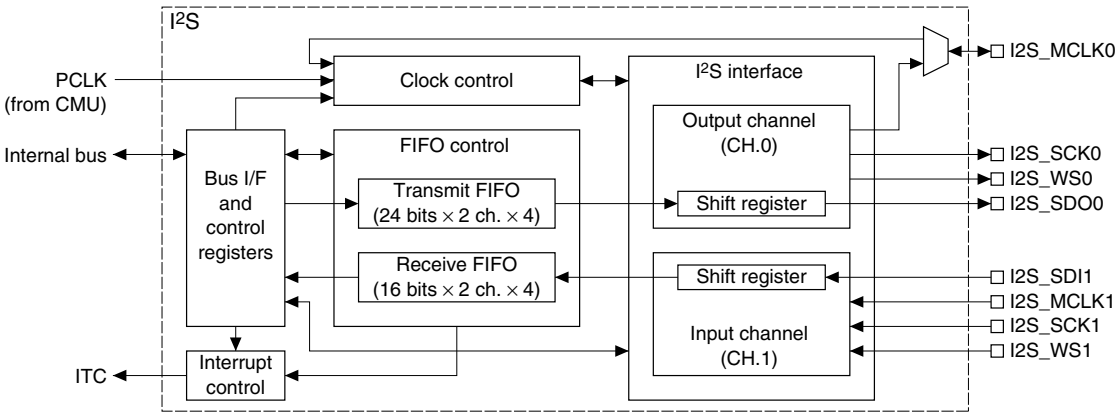


Figure VI.5.1.1 Structure of I2S Module

VI.5.2 I²S I/O Pins

Table VI.5.2.1 lists the I²S pins.

Table VI.5.2.1 List of I²S Pins

Pin name	I/O	Size	Function
I2S_SDO0	O	1	I²S data output pin This pin outputs serial PCM data.
I2S_WS0	O	1	I²S word-select signal (LRCLK) output pin This pin outputs the word-select signal that indicates the channel (L or R) of the data being output.
I2S_SCK0	O	1	I²S synchronous clock (bit clock) output pin This pin outputs the synchronous clock (bit clock) for serial data.
I2S_MCLK0	I/O	1	I²S master clock input/output pin for I²S output channel This pin outputs the I²S master clock when the I²S module uses the internal clock as the source of MCLK. When an external clock is used as the source of MCLK, this pin inputs the I²S master clock to the clock generator of the I²S module.
I2S_SDI1	I	1	I²S data input pin This pin inputs serial PCM data.
I2S_WS1	I	1	I²S word-select signal (LRCLK) input pin This pin inputs the word-select signal that indicates the channel (L or R) of the data being input.
I2S_SCK1	I	1	I²S synchronous clock (bit clock) input pin This pin inputs the synchronous clock (bit clock) for serial data.
I2S_MCLK1	I	1	I²S master clock input pin This pin inputs the I²S master clock.

The I²S input/output pins (I2S_SDO0, I2S_SDI1, I2S_WS*x*, I2S_SCK*x*, I2S_MCLK*x*) are shared with the I/O ports and they are initialized as general-purpose I/O port pins by default. Before using these pins for the I²S, the pin functions must be switched using the Port Function Select Register.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.5.3 I²S Module Operating Clock

The I²S module use PCLK (= system clock) generated by the CMU as the operating clock.
PCLK is always supplied to the I²S module in normal mode and HALT mode. In SLEEP mode, the clock supply stops. Therefore, the I²S module also stops operating in SLEEP mode.

The transfer clocks for CH.0 are generated in the I²S module by dividing the source clock (PCLK or external MCLK selectable with software). To generate the CH.0 clocks, set I2SEN0 (D8/I2S_CTL_OUT register) to 1. When I2SEN0 is set to 0, the clock goes off.

* **I2SEN0**: I²S Ch.0 Enable Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D8/0x5300)

The transfer clocks for CH.1 are input from the clock input pins. The clock input is enabled by setting I2SEN1 (D0/ I2S_CTL_IN register) to 1.

* **I2SEN1**: I²S CH.1 Enable Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D0/0x5302)

VI.5.4 Setting the I²S Module

When performing data transfers via the I²S bus, the following settings must be made before data transfer is actually begun:

1. Setting the I/O pins
2. Setting the I²S interface clocks
3. Setting the data format and timing
4. Setting interrupts

The following explains the content of each setting.

Note: Always make sure the I²S module is not started (I2SSTART0 (D0/I2S_START register)/ I2SSTART1 (D8/I2S_START register) = 0) before these settings are made. A change of settings during operation may cause a malfunction.

∗ **I2SSTART0**: I²S CH.0 Start/Stop Control Bit in the I²S Start/Stop (I2S_START) Register (D0/0x5308)
∗ **I2SSTART1**: I²S CH.1 Start/Stop Control Bit in the I²S Start/Stop (I2S_START) Register (D8/0x5308)

Setting the I/O pins

Configure the Port Function Select Registers to enable the I²S input/output functions. For details of pin functions and how to switch over, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

Setting the I²S interface clocks

The I²S module inputs/outputs the following three clocks:

1. I2S_MCLK (master clock)
2. I2S_SCK (bit clock)
3. I2S_WS (word-select clock)

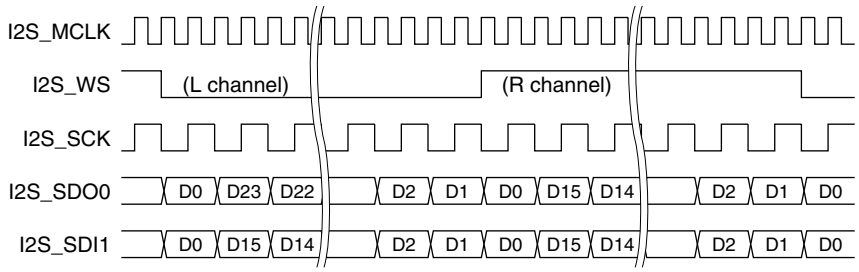


Figure VI.5.4.1 I²S Interface Clocks

The following shows the configurable clock conditions and their control bits. For more information on clock setting, see Section VI.5.10, "Setting the I²S Clocks."

Source clock for I2S_MCLK (master clock)

Either the internal clock (PCLK) or the external clock input from the I2S_MCLK0 pin of I²S CH.0 can be selected as the source clock for the master clock (I2S_MCLK) using MCLKSEL (D15/I2S_DV_MCLK register).

∗ **MCLKSEL**: I2S_MCLK Source Clock Select Bit in the I²S MCLK Divide Ratio (I2S_DV_MCLK) Register (D15/0x5304)

When MCLKSEL is set to 0 (default), the I²S module generates the master clock (I2S_MCLK) from PCLK using a frequency divider and outputs the clock from the I2S_MCLK0 pin of I²S CH.0. When MCLKSEL is set to 1, the clock input from the I2S_MCLK0 pin is directly send to the clock generation circuit in the I²S module to generate the bit clock and the word select clock.

Divide ratio for I2S_MCLK (master clock)

When the internal clock is selected as the source clock for I2S_MCLK, the I²S module generates I2S_MCLK to be output from the I2S_MCLK0 pin by dividing the PCLK (= system clock) generated by the CMU. Specify the divide ratio using MCLKDIV[5:0] (D[5:0]/I2S_DV_MCLK register).

* **MCLKDIV[5:0]**: I2S_MCLK Divide Ratio Select Bits in the I²S MCLK Divide Ratio (I2S_DV_MCLK) Register (D[5:0]/0x5304)

Table VI.5.4.1 Setting I2S_MCLK (Master Clock)

MCLKDIV[5:0]	I2S_MCLK
0x3f	PCLK•1/64
0x3e	PCLK•1/63
0x3d	PCLK•1/62
:	:
0x2	PCLK•1/3
0x1	PCLK•1/2
0x0	PCLK•1/1

(Default: 0x0)

Divide ratio for I2S_SCK (bit clock)

The I²S module generates the bit clock to be output from the I2S_SCK0 pin of the I²S CH.0 by dividing the source clock selected for I2S_MCLK.
Specify the divide ratio using BCLKDIV[7:0] (D[7:0]/I2S_DV_AUDIO_CLK register).

* **BCLKDIV[7:0]**: I²S CH.0 Bit Clock Divide Ratio Select Bits in the I²S Audio Clock Divide Ratio (I2S_DV_AUDIO_CLK) Register (D[7:0]/0x5306)

Table VI.5.4.2 Setting the Bit Clock

BCLKDIV[7:0]	Bit clock (I2S_SCK0)
0xff	SRC_CLK•1/512
0xfe	SRC_CLK•1/510
0xfd	SRC_CLK•1/508
:	:
0x2	SRC_CLK•1/6
0x1	SRC_CLK•1/4
0x0	SRC_CLK•1/2

(SRC_CLK = PCLK or I2S_MCLK0 input clock, default: 0x0)

The I²S CH.0 bit clock frequency is calculated as below.

$$f_{I2S\_SCK0} = \frac{f_{SRC\_CLK}}{(BCLKDIV + 1) \times 2} \text{ [Hz]}$$

f_{I2S_SCK0}: I²S CH.0 bit clock frequency [Hz]
f_{SRC_CLK}: PCLK or I2S_MCLK0 input clock frequency [Hz]
BCLKDIV: BCLKDIV[7:0] set value (0x0–0xff)

I²S CH.1 uses the bit clock input from the I2S_SCK1 pin, therefore the above setting is not applied to CH.1.

Sample clock (I2S_WS) period

The I²S CH.0 generates the sample clock (word-select clock) to be output from the I2S_WS0 pin by counting the bit clock configured with BCLKDIV[7:0]. Specify the half cycle (a high or low level period) of the I2S_WS clock with the number of bit clock cycles using WSCLKCYC0[3:0] (D[11:8]/I2S_DV_AUDIO_CLK register).
The I²S CH.1 inputs the sample clock (word-select clock) from the I2S_WS1 pin. The clock period must be specified with the number of bit clock cycles using WSCLKCYC1[3:0] (D[15:12]/I2S_DV_AUDIO_CLK register) similar to CH.0.

* **WSCLKCYC0[3:0]**: I²S CH.0 WS Clock Cycle Setup Bits in the I²S Audio Clock Divide Ratio (I2S_DV_AUDIO_CLK) Register (D[11:8]/0x5306)
* **WSCLKCYC1[3:0]**: I²S CH.1 WS Clock Cycle Setup Bits in the I²S Audio Clock Divide Ratio (I2S_DV_AUDIO_CLK) Register (D[15:12]/0x5306)

Table VI.5.4.3 Setting the Sample Clock Period

WSCLKCYC*x*[3:0]	Sample clock period (number of bit clock cycles)
0xf	31 clocks
0xe	30 clocks
0xd	29 clocks
0xc	28 clocks
0xb	27 clocks
0xa	26 clocks
0x9	25 clocks
0x8	24 clocks
0x7	23 clocks
0x6	22 clocks
0x5	21 clocks
0x4	20 clocks
0x3	19 clocks
0x2	18 clocks
0x1	17 clocks
0x0	16 clocks

(Default: 0x0)

The sampling clock frequency is calculated as below.

$$fs = \frac{f_{I2S\_SCK}}{n \times 2}[Hz]$$

fs: Sampling clock frequency [Hz]
f_{I2S_SCK}: Bit clock frequency [Hz] (CH.0: See Table VI.5.4.2. CH.1: I2S_SCK1 input clock frequency)
n: Number of bit clocks selected by WSCLKCYC*x*[3:0] (See Table VI.5.4.3.)

Note: The value to be set to the WSCLKCYC*x*[3:0] is not the number of audio data bits, but the number of bit clock cycles that is used to adjust the sample clock period. It must be equal to or greater than the number of audio data bits (24 bits or 16 bits).

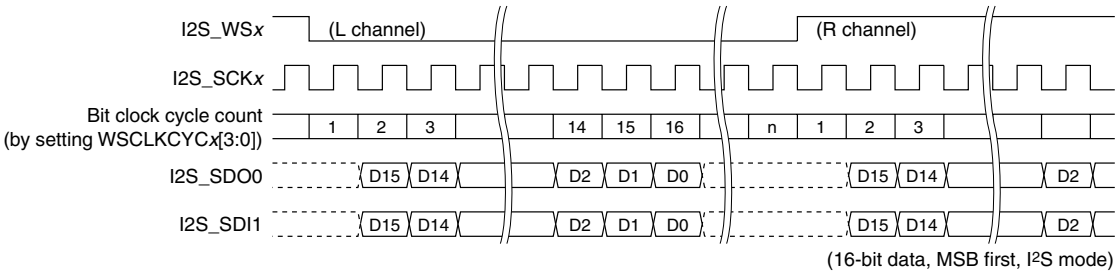


Figure VI.5.4.2 Sample Clock Period

Selecting the word clock mode

The I2S_WS signal represents the current output channel (L or R) with its level (low or high). Use WCLKMD0 (D7/I2S_CTL_OUT register) for CH.0 and WCLKMD1 (D5/I2S_CTL_IN register) for CH.1 to select the relationship between the signal level and the L/R channel.

- **WCLKMD0**: I²S CH.0 Output Word Clock Mode Select Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D7/0x5300)
- **WCLKMD1**: I²S CH.1 Input Word Clock Mode Select Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D5/0x5302)

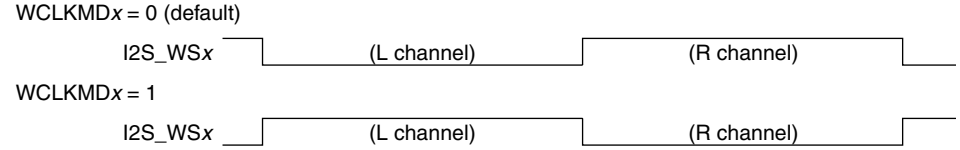


Figure VI.5.4.3 Selecting Word Clock Mode

I2S_SCK (bit clock) polarity

Use BCLKPOL0 (D6/I2S_CTL_OUT register) for CH.0 and BCLKPOL1 (D4/I2S_CTL_IN register) for CH.1 to select the bit clock polarity.

- **BCLKPOL0**: I²S CH.0 Output Bit Clock Polarity Select Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D6/0x5300)
- **BCLKPOL1**: I²S CH.1 Input Bit Clock Polarity Select Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D4/0x5302)

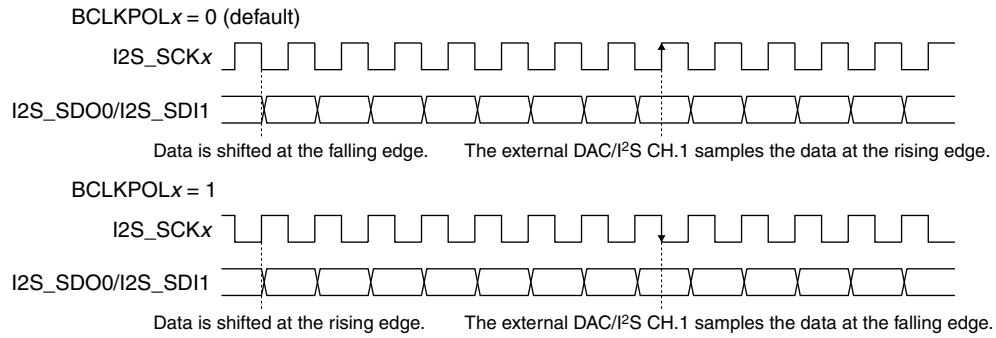


Figure VI.5.4.4 Selecting the Bit Clock Polarity

Setting the output data format and timing

Output data resolution

Use DATRES0 (D9/I2S_CTL_OUT register) to select either 16 bits or 24 bits as the output data resolution. Setting DATRES0 to 0 (default) selects 16 bits and setting 1 selects 24 bits.

- **DATRES0**: I²S CH.0 Output Data Resolution Select Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D9/0x5300)

The I²S CH.1 (input channel) supports 16-bit resolution only.

Data format (MSB first/LSB first)

Use DTFORM (D5/ I2S_CTL_OUT register) to select either MSB first or LSB first as the data output direction.

Setting DTFORM to 0 (default) selects MSB first and setting 1 selects LSB first.

- **DTFORM**: I²S CH.0 Output Data Format Select Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D5/0x5300)

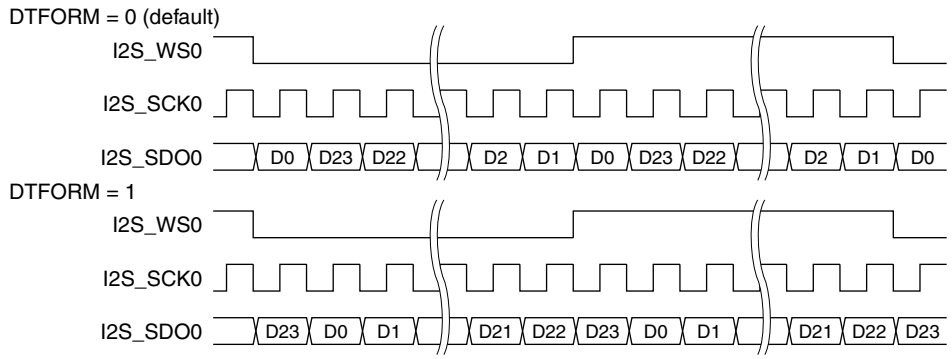


Figure VI.5.4.5 Selecting Output Data Format

Signed/unsigned format

When right justified mode is selected as the data output timing condition, output data can be configured to the signed or unsigned format using DTSIGN (D10/I2S_CTL_OUT register).

- **DTSIGN**: I²S CH.0 Signed/Unsigned Data Format Select Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D10/0x5300)

Setting DTSIGN to 0 (default) selects the unsigned format. The high-order bits that exceed the valid data size are set to 0. Setting 1 selects the signed format. The high-order bits that exceed the valid data size are set to the sign bit value (D23 or D15) of the valid data.

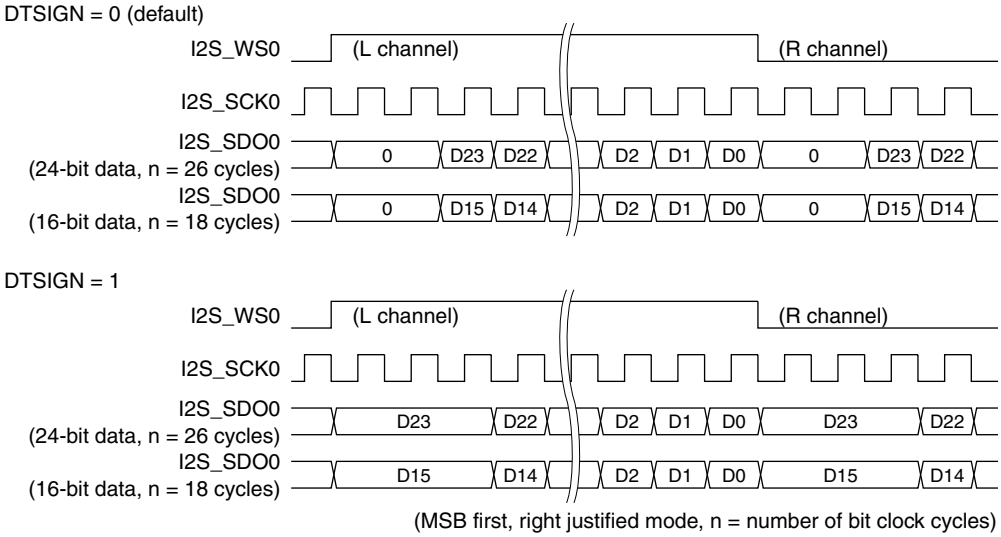


Figure VI.5.4.6 Unsigned and Signed Format

This setting is effective only in right justified mode. The other modes output only the unsigned data regardless of how DTSIGN is set.

Data output timing

Use DTTMG0[1:0] (D[3:2]/I2S_CTL_OUT register) for CH.0 and DTTMG1[1:0] (D[3:2]/I2S_CTL_IN register) for CH.1 to select the data output timing.

- **DTTMG0[1:0]**: I²S CH.0 Output Data Timing Select Bits in the I²S CH.0 Control (I2S_CTL_OUT) Register (D[3:2]/0x5300)
- **DTTMG1[1:0]**: I²S CH.1 Input Data Timing Select Bits in the I²S CH.1 Control (I2S_CTL_IN) Register (D[3:2]/0x5302)

Table VI.5.4.4 Data Input/Output Timing

DTTMG*x*[1:0]	Data output timing mode
0x3	Reserved
0x2	Right justified mode
0x1	Left justified mode
0x0	I²S mode

(Default: 0x0)

When DTTMG*x*[1:0] is set to 0x0 (default), I²S mode is selected. In this mode, the first bit of each data is input/output after one I2S_SCK clock delay from the I2S_WS signal edge.

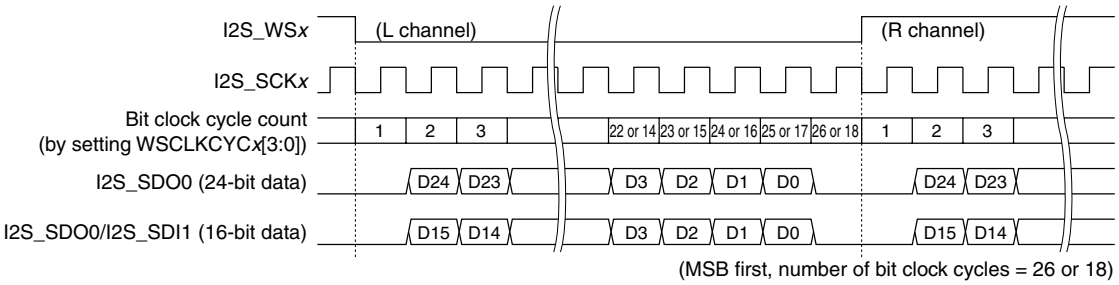


Figure VI.5.4.7 Data Input/Output Timing 1 (I²S Mode)

When DTTMG*x*[1:0] is set to 0x1, left justified mode is selected. In this mode, each data input/output starts at the I2S_WS signal edge.

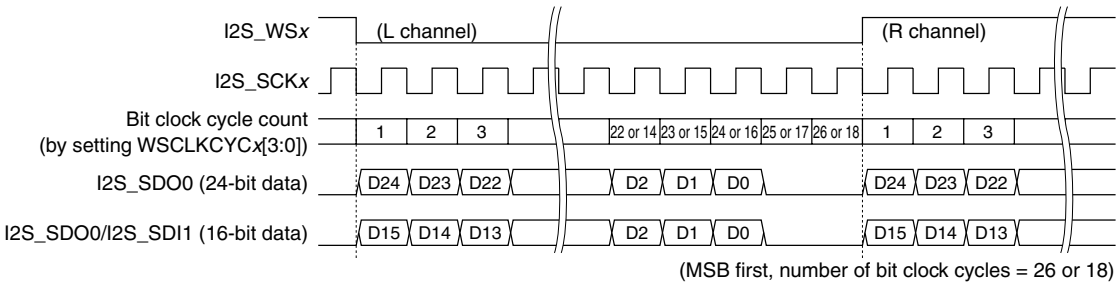


Figure VI.5.4.8 Data Input/Output Timing 2 (Left Justified Mode)

When DTTMG*x*[1:0] is set to 0x2, right justified mode is selected. In this mode, input/output data is right justified to the I2S_WS signal edge.

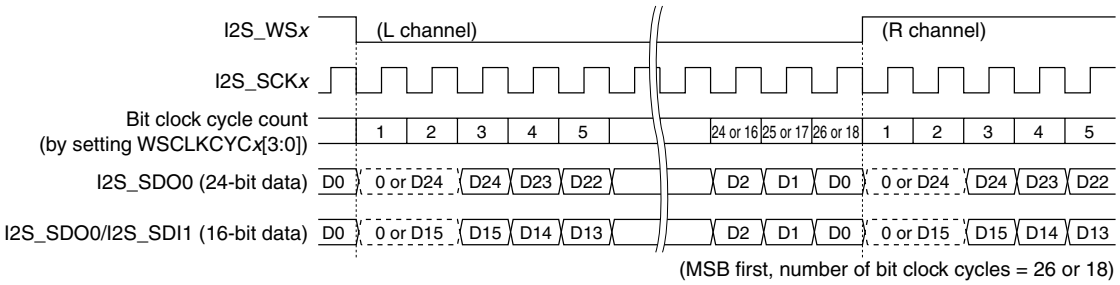


Figure VI.5.4.9 Data Input/Output Timing 3 (Right Justified Mode)

Note: When using right justified mode, the number of bit clock cycles (sample clock period) must be equal to or greater than [Data bit size + 2].

Setting interrupt conditions in the I²S module

The following explains settings of the interrupt mode in the I²S module. For the interrupt settings in the ITC, refer to Section VI.5.7, "I²S Interrupt."

Interrupt mode for I²S output (CH.0)

The I²S CH.0 has an embedded FIFO (24 bits × 2 channels (L & R) × 4) for storing four stereo data to be output. The I²S module can generate interrupts to request the application program to write output data into the FIFO when it reads the data written into the FIFO to output. The I²S CH.0 provides three interrupt modes with different interrupt timings: half empty interrupt mode, whole empty interrupt mode, and one empty interrupt mode. Use I2SINTMD0[1:0] (D[3:2]/I2S_INT_MOD register) to select an interrupt mode. Furthermore, set I2SINTEN0 (D0/I2S_INT_MOD register) to 1 to enable the I²S CH.0 interrupt.

- **I2SINTMD0[1:0]**: I²S CH.0 Interrupt Mode Select Bits in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D[3:2]/0x530c)
- **I2SINTEN0**: I²S CH.0 Interrupt Enable Bit in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D0/0x530c)

Table VI.5.4.5 Selecting I²S CH.0 Interrupt Mode

I2SINTMD0[1:0]	Interrupt mode
0x3	Reserved
0x2	One empty interrupt mode
0x1	Whole empty interrupt mode
0x0	Half empty interrupt mode

(Default: 0x0)

Whole empty interrupt mode

While audio data is being output in this mode, the I²S CH.0 generates an interrupt after all data (four stereo data) has been read out from the FIFO to transmit. In other words, the FIFO is empty when an interrupt occurs. Therefore, the application program needs to fill the FIFO with four stereo data (24 or 16 bits × 2 channels (L & R) × 4) at once after an interrupt occurs.

Half empty interrupt mode (default)

In this mode, the I²S CH.0 generates an interrupt after two stereo data has been read out from the FIFO to transmit. In this case, the FIFO may be empty or it may still contain one or two data remained (the FIFO status can be checked using the status bits). The application program needs to fill the FIFO with two stereo data (24 or 16 bits × 2 channels (L & R) × 2) at once after an interrupt occurs.

One empty interrupt mode

In this mode, the I²S CH.0 generates an interrupt after one stereo data has been read out from the FIFO to transmit. In this case, the FIFO may be empty or it may still contain one to three data remained (the FIFO status can be checked using the status bits). The application program needs to fill the FIFO with one stereo data (24 or 16 bits × 2 channels (L & R) × 1) at once after an interrupt occurs.

Interrupt mode for I²S input (CH.1)

The I²S CH.1 has an embedded FIFO (16 bits × 2 channels (L & R) × 4) for storing four received stereo data. The I²S module can generate interrupts to request the application program to read data in the FIFO when the received data is written to the FIFO. The I²S CH.1 provides three interrupt modes with different interrupt timings: half full interrupt mode, whole full interrupt mode, and one data interrupt mode. Use I2SINTMD1[1:0] (D[5:4]/I2S_INT_MOD register) to select an interrupt mode. Furthermore, set I2SINTEN1 (D1/I2S_INT_MOD register) to 1 to enable the I²S CH.1 interrupt.

* **I2SINTMD1[1:0]**: I²S CH.1 Interrupt Mode Select Bits in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D[5:4]/0x530c)
* **I2SINTEN1**: I²S CH.1 Interrupt Enable Bit in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D1/0x530c)

Table VI.5.4.6 Selecting I²S CH.1 Interrupt Mode

I2SINTMD1[1:0]	Interrupt mode
0x3	Reserved
0x2	One data interrupt mode
0x1	Whole full interrupt mode
0x0	Half full interrupt mode

(Default: 0x0)

Whole full interrupt mode

While audio data is being input in this mode, the I²S CH.1 generates an interrupt after four received stereo data have been written to the FIFO. In other words, the FIFO is full when an interrupt occurs. Therefore, the application program needs to read four stereo data (16 bits × 2 channels (L & R) × 4) from the FIFO at once after an interrupt occurs.

Half full interrupt mode (default)

In this mode, the I²S CH.1 generates an interrupt after two received stereo data have been written to the FIFO. In this case, the FIFO may be full or it may contain two or three received data (the FIFO status can be checked using the status bits). The application program needs to read two stereo data (16 bits × 2 channels (L & R) × 2) from the FIFO at once after an interrupt occurs.

One data interrupt mode

In this mode, the I²S CH.1 generates an interrupt after one received stereo data has been written to the FIFO. In this case, the FIFO may be full or it may contain one to three data (the FIFO status can be checked using the status bits). The application program needs to read one stereo data (16 bits × 2 channels (L & R) × 1) from the FIFO at once after an interrupt occurs.

VI.5.5 Data Output Control (CH.0)

The following shows audio data output procedure:

1. Set up the I²S conditions as described in the previous section.
2. Set up the interrupt conditions as described in the previous section. Also the ITC registers must be set up (explained later).
3. Write 1 to the I2SEN0 (D8/I2S_CTL_OUT register) to turn the I²S CH.0 circuit on.
   The I²S CH.0 circuit starts frequency division of the source clock.

   * **I2SEN0**: I²S Ch.0 Enable Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D8/0x5300)

4. Set the output channel mode using CHMD[1:0] (D[1:0]/I2S_CTL_OUT register).

   * **CHMD[1:0]**: I²S CH.0 Output Channel Mode Select Bits in the I²S CH.0 Control (I2S_CTL_OUT) Register (D[1:0]/0x5300)

   Table VI.5.5.1 Selecting Output Channel Mode

CHMD[1:0]	Output channel mode	L channel	R channel
0x3	Mute	0	0
0x2	Mono (L)	Data output	0
0x1	Mono (R)	0	Data output
0x0	Stereo	Data output	Data output

   (Default: 0x0)

   The output channel mode can be switched even if data is being output. In this case, the mode changes after the current word output has finished.

5. Write the first audio data to the FIFO.
   The 16-bit register I2S_FIFO_OUT (0x5310) is used to write the output data to the FIFO. Up to four stereo data (24 or 16 bits × 2 channels (L & R) × 4) can be written to the FIFO regardless of the data size. Before starting audio data output, fill the FIFO with the first four stereo data.
   When the data size is 16 bits, use a 16-bit memory write (`ld [%rb],%rs`) instruction for writing data. Note that 8-bit and 24-bit memory write instructions cannot be used when the data size is 16 bits.
   When the data size is 24 bits, use a 24-bit memory write (`ld.a [%rb],%rs`) instruction for writing data. In this case, two memory write access cycles are generated for writing one channel data. Note that 8-bit and 16-bit memory write instructions cannot be used when the data size is 24 bits.

   First write L-channel data, then R-channel data. Both channel data must be written as a pair even if "mono" is selected as the output channel mode.

   When four stereo data is written to the FIFO, the FIFO becomes full and the I2SFIFOFF0 flag (D1/I2S_FIFO_STAT register) is set to 1. Note that the newest data of the FIFO is overwritten if data is written to I2S_FIFO_OUT in this status.

   * **I2SFIFOFF0**: I²S CH.0 FIFO Full Flag Bit in the I²S FIFO Status (I2S_FIFO_STAT) Register (D1/0x530a)

6. Write 1 to I2SOUTEN (D4/I2S_CTL_OUT register) to enable I²S output.

   * **I2SOUTEN**: I²S CH.0 Output Enable Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D4/0x5300)

   When I2SOUTEN = 0, the I2S_MCLK0 and I2S_WS0 pins are fixed at 0. The I2S_SDO0 pin is left unchanged. The I2S_SCK0 pin is fixed at 0 (when BCLKPOL0 (D6/I2S_CTL_OUT register) = 0) or 1 (when BCLKPOL0 = 1).
   When I2SOUTEN is set to 1, all output pins enter standby status.

   * **BCLKPOL0**: I²S CH.0 Output Bit Clock Polarity Select Bit in the I²S CH.0 Control (I2S_CTL_OUT) Register (D6/0x5300)

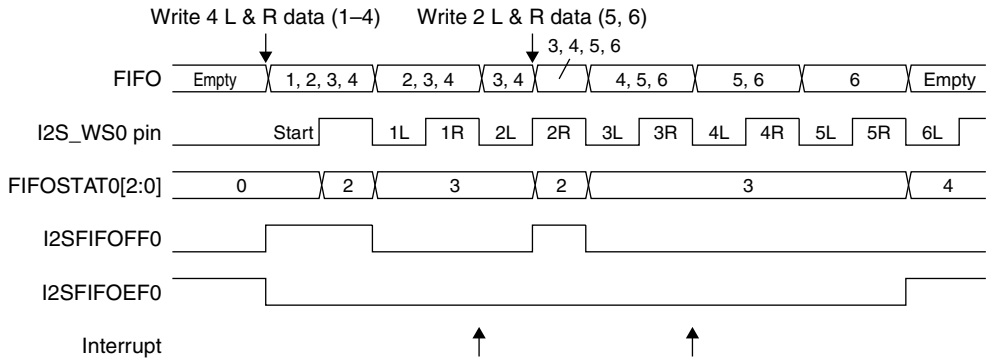
7. Write 1 to I2SSTART0 (D0/I2S_START register) to start output.

   * **I2SSTART0**: I²S CH.0 Start/Stop Control Bit in the I²S Start/Stop (I2S_START) Register (D0/0x5308)

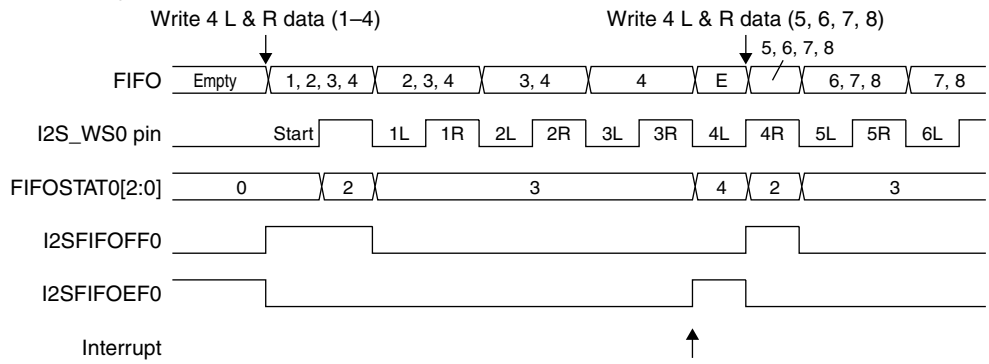
   When I2SSTART0 is 0, the bit clock is stopped with pulled down to low. The word select clock is also stopped with pulled up to high if WCLKMD0 = 0 or pulled down to low if WCLKMD0 = 1.
   When I2SSTART0 is set to 1, the I²S module loads one data (L & R) in the FIFO to the shift register and it starts serial output in sync with the I2S_WS signal.
   The data in the shift register is shifted at the I2S_SCK clock edge and is output from the L channel first. When an output of one data (L & R) has finished, the next data is read out from the FIFO and the same operation repeats.

   When the number of data according to the interrupt mode has been read out from the FIFO, an interrupt can be generated.
   In half empty interrupt mode (default), the I²S module generates an interrupt after two stereo data has been read out from the FIFO. In this case, write the next two stereo data (24 or 16 bits × 2 channels (L & R) × 2) to the FIFO.
   In whole empty interrupt mode, the I²S module generates an interrupt after all data (four stereo data) has been read out from the FIFO. In this case, write the next four stereo data (24 or 16 bits × 2 channels (L & R) × 4) to the FIFO.
   In one empty interrupt mode, the I²S module generates an interrupt after one stereo data has been read out from the FIFO. In this case, write the next one stereo data (24 or 16 bits × 2 channels (L & R) × 1) to the FIFO.

In half empty interrupt mode



In whole empty interrupt mode



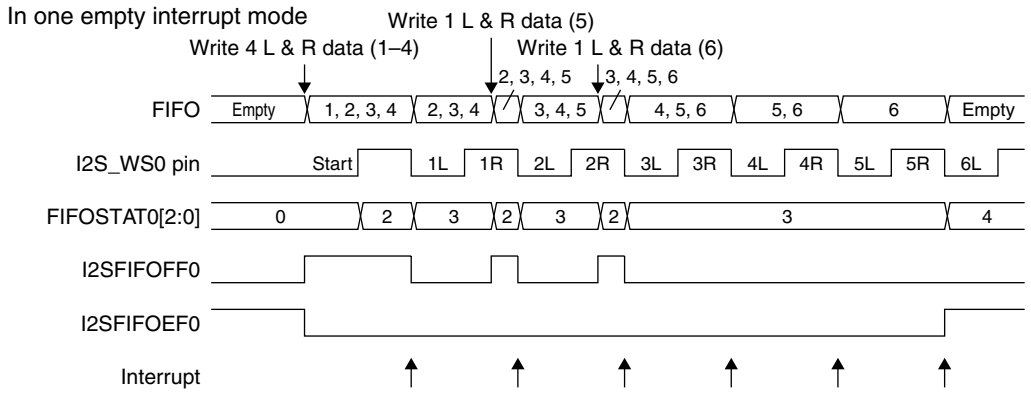


Figure VI.5.5.1 FIFO Data and Interrupts

When the FIFO becomes empty, I2SFIFOEF0 (D0/I2S_FIFO_STAT register) is set to 1.

∗ **I2SFIFOEF0**: I²S CH.0 FIFO Empty Flag Bit in the I²S FIFO Status (I2S_FIFO_STAT) Register (D0/0x530a)

When data is written to the FIFO, I2SFIFOEF0 is reset to 0 and the data output continues.

Furthermore, the I²S CH.0 provides the status bits FIFOSTAT0[2:0] (D[4:2]/I2S_FIFO_STAT register) that indicate the FIFO state machine.

∗ **FIFOSTAT0[2:0]**: I²S CH.0 FIFO State Machine Bits in the I²S FIFO Status (I2S_FIFO_STAT) Register (D[4:2]/0x530a)

Table VI.5.5.2 Monitoring the FIFO State Machine

FIFOSTAT0[2:0]	State
0x7–0x6	Reserved
0x5	FLUSH: FIFO is flushing the remained audio data before it stops.
0x4	EMPTY: FIFO is empty.
0x3	LACK: FIFO is not full and not empty.
0x2	FULL: FIFO is full.
0x1	INIT: Initialize all four entries of FIFO.
0x0	STOP: FIFO is idle.

(Default: 0x0)

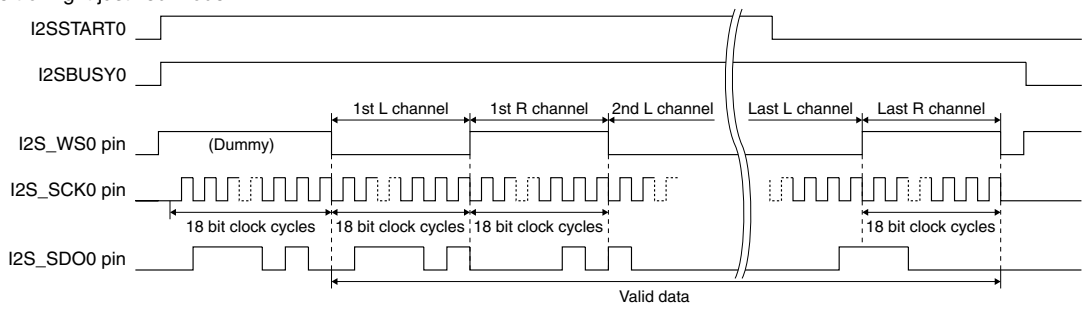
I2SBUSY0 (D7/I2S_START register) is set to 1 while data is being output. This flag can be used to check the output status.

∗ **I2SBUSY0**: I²S CH.0 Busy Flag Bit in the I²S Start/Stop (I2S_START) Register (D7/0x5308)

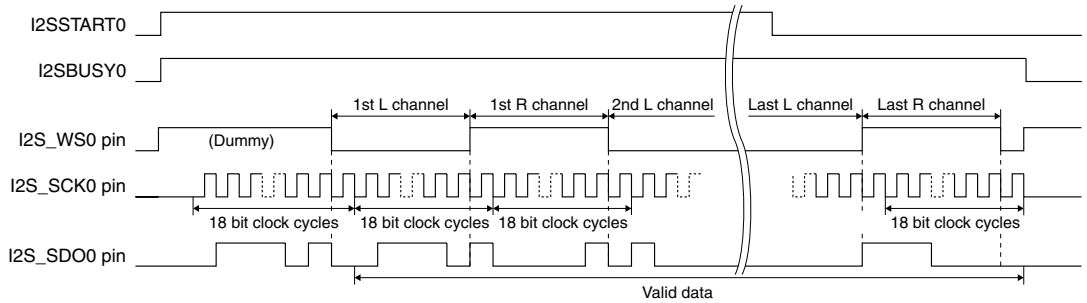
8. To stop output, write 0 to I2SSTART0 (D0/I2S_START register).
   When I2SSTART0 is set to 0, the I²S module will stop data output after the remaining data stored in the FIFO are all output. When the I²S stops, I2SBUSY0 is reset to 0.

9. To disable output, write 0 to I2SOUTEN (D4/I2S_CTL_OUT register) after writing 0 to I2SSTART0 (D0/I2S_START register) to stop the current output. Finally, write 0 to I2SEN0 (D8/I2S_CTL_OUT register) to turn the I²S CH.0 circuit off. You can disable output and turn off I²S CH.0 at the same time.

In left or right justified mode



In I²S mode



Conditions: DATRES0 = 0 (16-bit data), CHMD[1:0] = 0x0 (stereo), WCLKMD0 = 0 (L ch = low),
BCLKPOL0 = 0 (rising edge), WSCLKCYC0[3:0] = 0x2 (18 clocks)

Figure VI.5.5.2 Data Output Timing Chart

∗ Output when mute or mono mode is selected

When mute mode is selected using CHMD[1:0] (D[1:0]/I2S_CTL_OUT register), the I2S_SDO0 pin is fixed at 0. However, the FIFO and shift register run the same as stereo mode and three clock signals are output normally. Also in mono mode, the I2S_SDO0 pin is fixed at 0 during the output period for the unselected channel.

The FIFO data is read out normally, therefore an interrupt caused by a FIFO empty occurs.

If CHMD[1:0] is changed when data is being output, the mode changes after the current L & R data output has finished.

VI.5.6 Data Input Control (CH.1)

The following shows audio data input procedure:

1. Set up the I²S conditions as described in Section VI.5.4.
2. Set up the interrupt conditions as described in Section VI.5.4. Also the ITC registers must be set up (explained later).
3. Write 1 to the I2SEN1 (D0/I2S_CTL_IN register) to turn the I²S CH.1 circuit on.

   * **I2SEN1**: I²S CH.1 Enable Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D0/0x5302)

4. Write 1 to I2SSTART1 (D8/I2S_START register) to start receiving.
   The I²S CH.1 circuit enables the clock input from the I2S_WS1 and I2S_SCK1 pins.

   * **I2SSTART1**: I²S CH.1 Start/Stop Control Bit in the I²S Start/Stop (I2S_START) Register (D8/0x5308)

   The I²S CH.1 starts data receiving at the first falling or rising edge of the I2S_WS1 input clock that goes to the L channel level according to the WCLKMD1 (D5/I2S_CTL_IN register) setting.

   * **WCLKMD1**: I²S CH.1 Input Word Clock Mode Select Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D5/0x5302)

   The data bits are sampled at the rising or falling edge of the I2S_SCK1 input clock specified by BCLKPOL1 (D4/I2S_CTL_IN register) and received in the receive shift register.

   * **BCLKPOL1**: I²S CH.1 Input Bit Clock Polarity Select Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D4/0x5302)

   After a 32-bit data is received in the shift register, the received data is loaded to the receive FIFO assuming that the first received 16 bits are L channel data and the following 16 bits are R channel data. Up to four stereo data (16 bits × 2 channels (L & R) × 4) can be stored in the FIFO.

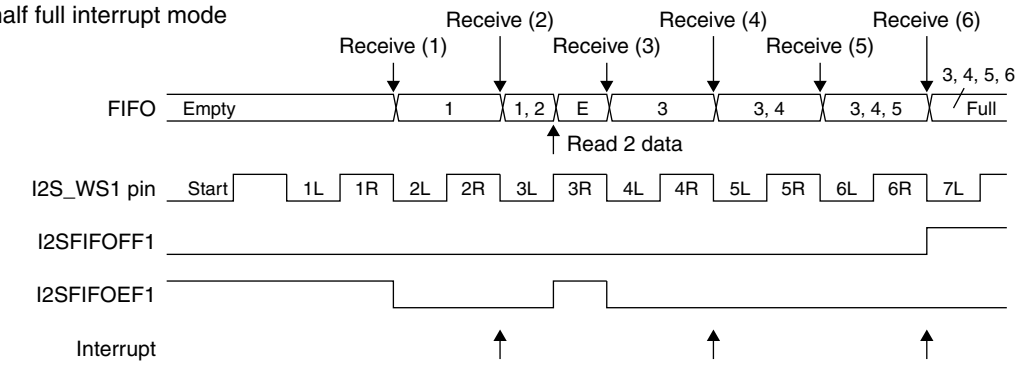
   When the number of data according to the interrupt mode has been loaded to the FIFO, an interrupt can be generated.
   In half full interrupt mode (default), the I²S module generates an interrupt after two stereo data has been received in the FIFO. In this case, read two stereo data (16 bits × 2 channels (L & R) × 2) from the FIFO.
   In whole full interrupt mode, the I²S module generates an interrupt after four stereo data has been received in the FIFO (the FIFO becomes full). In this case, read four stereo data (16 bits × 2 channels (L & R) × 4) from the FIFO.
   In one data interrupt mode, the I²S module generates an interrupt after one stereo data has been received in the FIFO. In this case, read one stereo data (16 bits × 2 channels (L & R) × 1) from the FIFO.

   If the FIFO becomes full, the I2SFIFOFF1 flag (D9/I2S_FIFO_STAT register) is set to 1. Note that the latest stereo data in the FIFO will be overwritten with the newly-received data if it has not been read out within 1 word clock cycle.

   * **I2SFIFOFF1**: I²S CH.1 FIFO Full Flag Bit in the I²S FIFO Status (I2S_FIFO_STAT) Register (D9/0x530a)

   When data is read out from the FIFO, I2SFIFOFF1 is reset to 0.

In half full interrupt mode



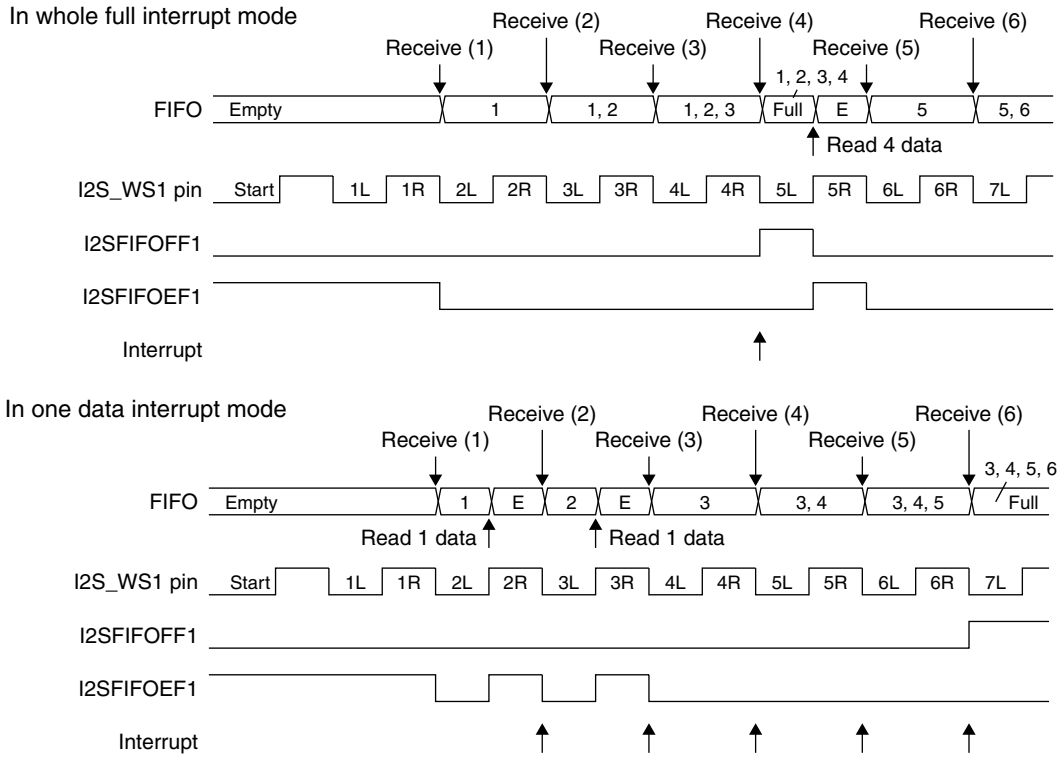


Figure VI.5.6.1 FIFO Data and Interrupts

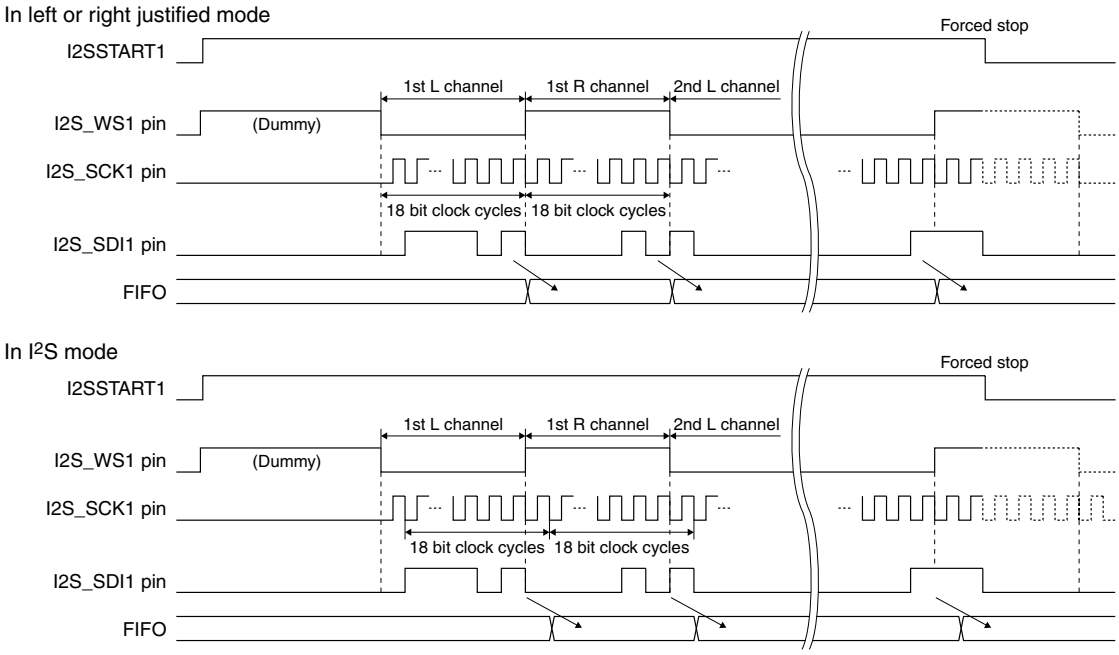
5. Read the received audio data from the FIFO.
   By using an interrupt described above, read the received data from the I2S_FIFO_IN register (0x5314).
   Use a 16-bit memory read (`ld %rd,[%rb]`) instruction for reading data from the FIFO. Note that 8-bit and 24-bit memory read instructions cannot be used.
   Both channel data should be read as a pair. L-channel data is read first, then R channel data.

   When the FIFO becomes empty by reading, I2SFIFOEF1 (D8/I2S_FIFO_STAT register) is set to 1.

   * **I2SFIFOEF1**: I²S CH.1 FIFO Empty Flag Bit in the I²S FIFO Status (I2S_FIFO_STAT) Register (D8/0x530a)

6. To stop input, write 0 to I2SSTART1 (D8/I2S_START register).
   When I2SSTART1 is set to 0, the I²S module stops data input immediately. After that, the data sent from the external I²S device will be ignored.

7. Finally, write 0 to I2SEN1 (D0/I2S_CTL_IN register) to turn the I²S CH.1 circuit off.



Conditions: WCLKMD1 = 0 (L ch = low), BCLKPOL1 = 0 (rising edge), WSCLKCYC1[3:0] = 0x2 (18 clocks)

Figure VI.5.6.2 Data Input Timing Chart

I²S bypass mode

The I²S module supports bypass mode. In this mode, the signals input from the I2S_MCLK1, I2S_SCK1, I2S_WS1, and I2S_SDI1 pins can be directly output from the I2S_MCLK0, I2S_SCK0, I2S_WS0, and I2S_SDO0 pins, respectively. Set I2SBYPASS (D1/I2S_CTL_IN register) to 1 to set the I²S module to bypass mode. When I2SBYPASS is 0 (default), I²S CH.0 and I²S CH.1 can be used independently.

* **I2SBYPASS**: I²S Bypass Mode Select Bit in the I²S CH.1 Control (I2S_CTL_IN) Register (D1/0x5302)

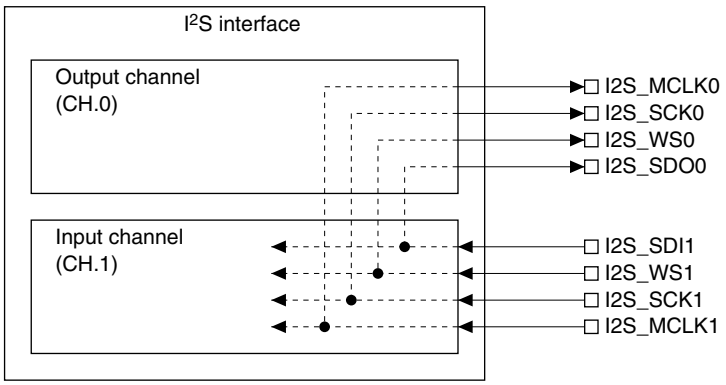


Figure VI.5.6.3 Bypass Mode

VI.5.7 I²S Interrupt

The I²S module can generate the following two types of interrupts:
- I²S FIFO empty interrupt (CH.0)
- I²S FIFO full interrupt (CH.1)

I²S FIFO empty interrupt (CH.0)

The I²S CH.0 has an embedded FIFO (24 bits × 2 channels (L & R) × 4) for storing four stereo data to be output. The I²S module can generate interrupts to request the application program to write output data into the FIFO when it reads the data written into the FIFO to output. The I²S CH.0 provides three interrupt modes with different interrupt timings: half empty interrupt mode, whole empty interrupt mode, and one empty interrupt mode. Use I2SINTMD0[1:0] (D[3:2]/I2S_INT_MOD register) to select an interrupt mode. Furthermore, set I2SINTEN0 (D0/I2S_INT_MOD register) to 1 to enable the I²S FIFO empty interrupt.

∗ **I2SINTMD0[1:0]**: I²S CH.0 Interrupt Mode Select Bits in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D[3:2]/0x530c)
∗ **I2SINTEN0**: I²S CH.0 Interrupt Enable Bit in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D0/0x530c)

Table VI.5.7.1 Selecting I²S CH.0 Interrupt Mode

I2SINTMD0[1:0]	Interrupt Mode
0x3	Reserved
0x2	One empty interrupt mode
0x1	Whole empty interrupt mode
0x0	Half empty interrupt mode

(Default: 0x0)

Whole empty interrupt mode

While audio data is being output in this mode, the I²S CH.0 generates an interrupt after all data (four stereo data) has been read out from the FIFO to transmit. In other words, the FIFO is empty when an interrupt occurs. Therefore, the application program needs to fill the FIFO with four stereo data (24 or 16 bits × 2 channels (L & R) × 4) at once after an interrupt occurs.

Half empty interrupt mode (default)

In this mode, the I²S CH.0 generates an interrupt after two stereo data has been read out from the FIFO to transmit. In this case, the FIFO may be empty or it may still contain one or two data remained (the FIFO status can be checked using the status bits). The application program needs to fill the FIFO with two stereo data (24 or 16 bits × 2 channels (L & R) × 2) at once after an interrupt occurs.

One empty interrupt mode

In this mode, the I²S CH.0 generates an interrupt after one stereo data has been read out from the FIFO to transmit. In this case, the FIFO may be empty or it may still contain one to three data remained (the FIFO status can be checked using the status bits). The application program needs to fill the FIFO with one stereo data (24 or 16 bits × 2 channels (L & R) × 1) at once after an interrupt occurs.

Note: Fill the FIFO with four stereo data (24 or 16 bits × 2 channels (L & R) × 4) at the beginning of transfer via I²S CH.0 (at the time of start by setting I2SSTART0 to 1) regardless of the interrupt mode (I2SINTMD0[1:0]).

- When filling the FIFO before enabling the interrupt
  Write four stereo data to the FIFO, then enable the interrupt. When a FIFO empty interrupt occurs, the interrupt handler routine must write one, two or four stereo data to the FIFO according to the interrupt mode you set.
- When filling the FIFO after enabling the interrupt
  The FIFO can also be filled in the interrupt handler routine after enabling the interrupt. Note, however, that you must fill the FIFO with four stereo data at the first time the interrupt handler routine is executed. For the second and subsequent interrupts, write one, two or four stereo data to the FIFO according to the interrupt mode you set.

I²S FIFO full interrupt (CH.1)

The I²S CH.1 has an embedded FIFO (16 bits × 2 channels (L & R) × 4) for storing four received stereo data. The I²S module can generate interrupts to request the application program to read data in the FIFO when the received data is written to the FIFO. The I²S CH.1 provides three interrupt modes with different interrupt timings: half full interrupt mode, whole full interrupt mode, and one data interrupt mode. Use I2SINTMD1[1:0] (D[5:4]/I2S_INT_MOD register) to select an interrupt mode. Furthermore, set I2SINTEN1 (D1/I2S_INT_MOD register) to 1 to enable the I²S FIFO full interrupt.

* **I2SINTMD1[1:0]**: I²S CH.1 Interrupt Mode Select Bits in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D[5:4]/0x530c)
* **I2SINTEN1**: I²S CH.1 Interrupt Enable Bit in the I²S Interrupt Mode Select (I2S_INT_MOD) Register (D1/0x530c)

Table VI.5.7.2 Selecting I²S CH.1 Interrupt Mode

I2SINTMD1[1:0]	Interrupt Mode
0x3	Reserved
0x2	One data interrupt mode
0x1	Whole full interrupt mode
0x0	Half full interrupt mode

(Default: 0x0)

Whole full interrupt mode

While audio data is being input in this mode, the I²S CH.1 generates an interrupt after four received stereo data have been written to the FIFO. In other words, the FIFO is full when an interrupt occurs. Therefore, the application program needs to read four stereo data (16 bits × 2 channels (L & R) × 4) from the FIFO at once after an interrupt occurs.

Half full interrupt mode (default)

In this mode, the I²S CH.1 generates an interrupt after two received stereo data have been written to the FIFO. In this case, the FIFO may be full or it may contain two or three received data (the FIFO status can be checked using the status bits). The application program needs to read two stereo data (16 bits × 2 channels (L & R) × 2) from the FIFO at once after an interrupt occurs.

One data interrupt mode

In this mode, the I²S CH.1 generates an interrupt after one received stereo data has been written to the FIFO. In this case, the FIFO may be full or it may contain one to three data (the FIFO status can be checked using the status bits). The application program needs to read one stereo data (16 bits × 2 channels (L & R) × 1) from the FIFO at once after an interrupt occurs.

ITC registers for I²S interrupts

Table VI.5.7.3 shows the control registers of the ITC provided for each I²S interrupt.

Table VI.5.7.3 ITC Registers

Cause of interrupt	Interrupt flag	Interrupt enable bit	Interrupt level setup bits
I²S FIFO empty	AIFT12 (D12/ITC_AIFLG)	AIEN12 (D12/ITC_AEN)	AILV12[2:0] (D[2:0]/ITC_AILV6)
I²S FIFO full	AIFT13 (D13/ITC_AIFLG)	AIEN13 (D13/ITC_AEN)	AILV13[2:0] (D[10:8]/ITC_AILV6)

ITC_AIFLG register (0x42e0)
ITC_AEN register (0x42e2)
ITC_AILV6 register (0x42f2)

When the I²S module outputs an interrupt signal, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the I²S interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the interrupt signal, regardless of how the interrupt enable bit is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the I²S interrupt. If the same interrupt level is set, the I²S FIFO empty interrupt has higher priority than the I²S FIFO full interrupt.
An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The I²S interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vectors

The following shows the vector numbers and vector addresses for the I²S interrupts:

Table VI.5.7.4 I²S Interrupt Vectors

Cause of interrupt	Vector number	Vector address
I²S FIFO empty	28 (0x1c)	TTBR + 0x70
I²S FIFO full	29 (0x1d)	TTBR + 0x74

VI.5.8 Details of Control Registers

Table VI.5.8.1 List of I²S Registers

Address	Register name		Function
0x5300	I2S_CTL_OUT	I²S CH.0 Control Register	Sets output conditions.
0x5302	I2S_CTL_IN	I²S CH.1 Control Register	Sets input conditions.
0x5304	I2S_DV_MCLK	I²S MCLK Divide Ratio Register	Configures MCLK.
0x5306	I2S_DV_AUDIO_CLK	I²S Audio Clock Divide Ratio Register	Configures the audio clock.
0x5308	I2S_START	I²S Start/Stop Register	Controls/indicates I²S start/stop status.
0x530a	I2S_FIFO_STAT	I²S FIFO Status Register	Indicates the FIFO status.
0x530c	I2S_INT_MOD	I²S Interrupt Mode Select Register	Sets the I²S interrupt conditions.
0x5310	I2S_FIFO_OUT	I²S CH.0 FIFO Register	L & R channel output data (16- or 24-bit access)
0x5314	I2S_FIFO_IN	I²S CH.1 FIFO Register	L & R channel input data (16-bit access)

The following describes each I²S register. These are all 16-bit registers except 0x5310.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x5300: I²S CH.0 Control Register (I2S_CTL_OUT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
I²S CH.0 Control Register (I2S_CTL_OUT)	0x5300 (16 bits)	D15–11	–	reserved	–				–	–	0 when being read.
		D10	DTSIGN	I²S CH.0 signed/unsigned data format select	1	Signed	0	Unsigned	0	R/W	
		D9	DATRES0	I²S CH.0 output data resolution select	1	24 bits	0	16 bits	0	R/W	
		D8	I2SEN0	I²S CH.0 enable	1	Enable	0	Disable	0	R/W	
		D7	WCLKMD0	I²S CH.0 output word clock mode select	1	L: high R: low	0	L: low R: high	0	R/W	
		D6	BCLKPOL0	I²S CH.0 output bit clock polarity select	1	Negative	0	Positive	0	R/W	
		D5	DTFORM	I²S CH.0 output data format select	1	LSB first	0	MSB first	0	R/W	
		D4	I2SOUTEN	I²S CH.0 output enable	1	Enable	0	Disable	0	R/W	
		D3–2	DTTMG0 [1:0]	I²S CH.0 output data timing select	DTTMG0[1:0]	Timing mode			0x0	R/W	
					0x3	reserved					
					0x2	Right justified					
					0x1	Left justified					
					0x0	I²S					
		D1–0	CHMD[1:0]	I²S CH.0 output channel mode select	CHMD[1:0]	Channel mode			0x0	R/W	
					0x3	Mute					
					0x2	Mono left					
					0x1	Mono right					
					0x0	Stereo					

Note: All the data transfer conditions must be set using this register before setting I2SSTART0 (D0/ I2S_START register) to start data output from the I²S CH.0.

D[15:11] Reserved

D10 DTSIGN: I²S CH.0 Signed/Unsigned Data Format Select Bit

Selects the data format in right justified mode.

1 (R/W): Signed

0 (R/W): Unsigned (default)

Setting DTSIGN to 0 (default) selects the unsigned format. The high-order bits that exceed the valid data size are set to 0. Setting 1 selects the signed format. The high-order bits that exceed the valid data size are set to the sign bit value (D23 or D15) of the valid data.

This setting is effective only in right justified mode. The other modes output only the unsigned data regardless of how DTSIGN is set.

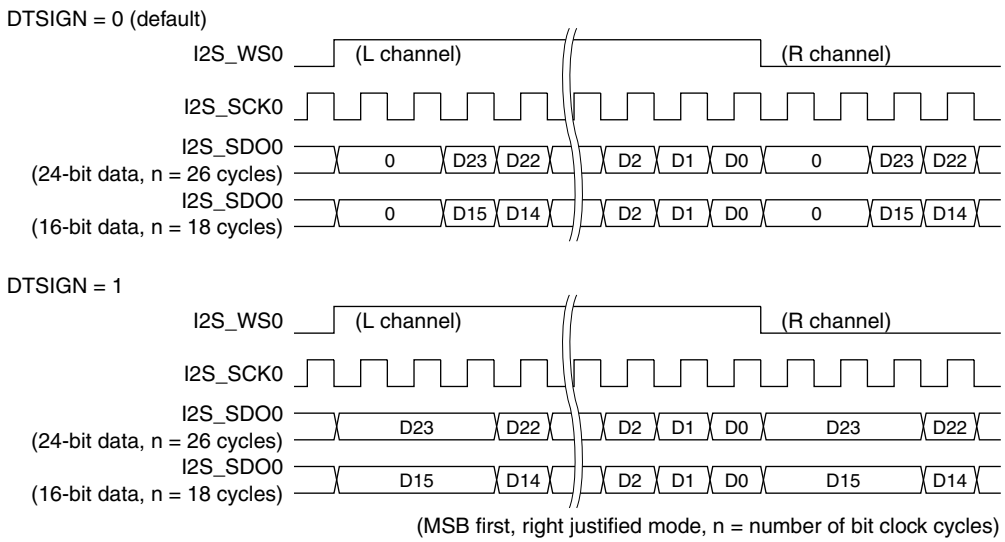


Figure VI.5.8.1 Unsigned and Signed Format

D9 DATRES0: I²S CH.0 Output Data Resolution Select Bit

Selects the output audio data resolution for I²S CH.0.
1 (R/W): 24 bits
0 (R/W): 16 bits (default)

Setting DATRES0 to 0 (default) selects 16 bits and setting 1 selects 24 bits.
The I²S CH.1 (input channel) supports 16-bit resolution only.

D8 I2SEN0: I²S CH.0 Enable Bit

Enables/disables operation of the I²S CH.0.
1 (R/W): Enable (on)
0 (R/W): Disable (off) (default)

When I2SEN0 is set to 1, the I²S CH.0 starts operating and data transfer is enabled.
When I2SEN0 is set to 0, the I²S CH.0 goes off.

D7 WCLKMD0: I²S CH.0 Output Word Clock Mode Select Bit

Selects the I2S_WS output signal level for indicating a channel.
1 (R/W): High = L channel, Low = R channel
0 (R/W): High = R channel, Low = L channel (default)

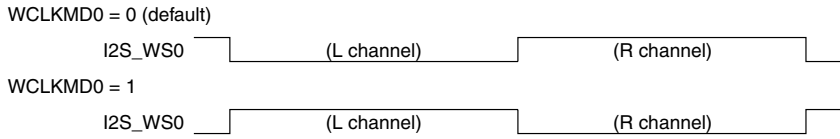


Figure VI.5.8.2 Selecting Word Clock Mode

D6 BCLKPOL0: I²S CH.0 Output Bit Clock Polarity Select Bit

Selects the bit clock polarity.
1 (R/W): Negative
0 (R/W): Positive (default)

When BCLKPOL0 is 0, the I2S_SDO0 output changes at the falling edge of the I2S_SCK clock (bit clock) and the external DAC samples the data bit at the rising edge of I2S_SCK.
When BCLKPOL0 is set to 1, the I2S_SDO0 output changes at the rising edge of I2S_SCK and the external DAC samples the data bit at the falling edge of I2S_SCK.

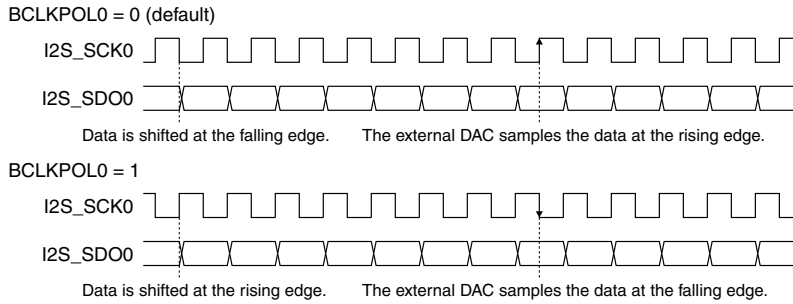


Figure VI.5.8.3 Selecting Bit Clock Polarity

D5 DTFORM: I²S CH.0 Output Data Format Select Bit

Selects either MSB first or LSB first as the data output direction.
1 (R/W): LSB first
0 (R/W): MSB first (default)

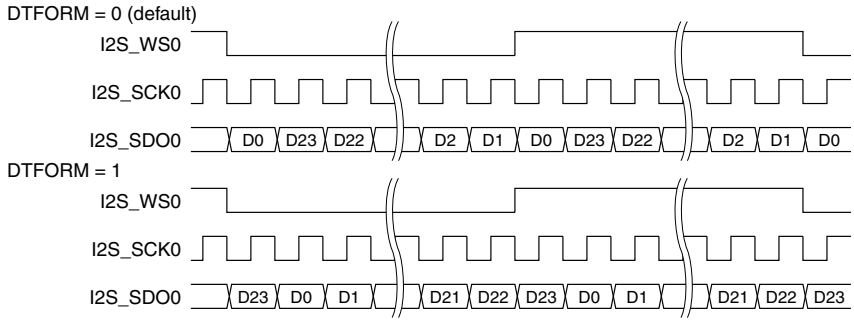


Figure VI.5.8.4 Selecting Data Format

D4 I2SOUTEN: I²S CH.0 Output Enable Bit

Enables/disables output of the I²S CH.0 signals.
1 (R/W): Enable (on)
0 (R/W): Disable (off) (default)

When I2SOUTEN = 0, the I2S_MCLK0 and I2S_WS0 pins are fixed at 0. The I2S_SDO0 pin is left unchanged. The I2S_SCK0 pin is fixed at 0 (when BCLKPOL0 (D6/I2S_CTL_OUT register) = 0) or 1 (when BCLKPOL0 = 1).
When I2SOUTEN is set to 1, all output pins enter standby status.

D[3:2] DTTMG0[1:0]: I²S CH.0 Output Data Timing Select Bits

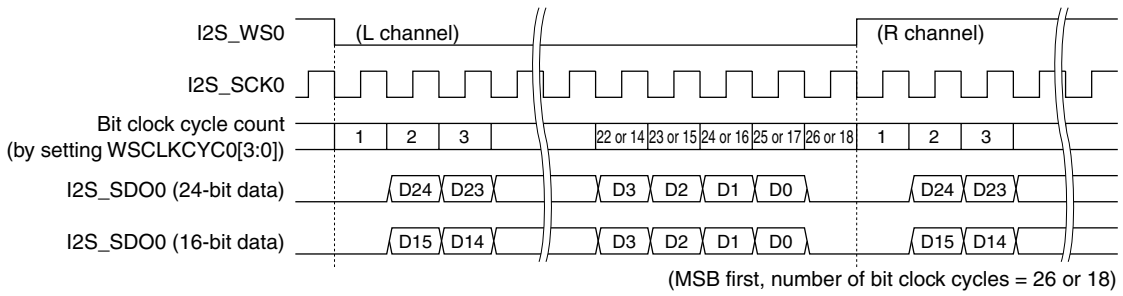
Selects the data bit output timing.

Table VI.5.8.2 Data Output Timing

DTTMG0[1:0]	Data output timing mode
0x3	Reserved
0x2	Right justified mode
0x1	Left justified mode
0x0	I²S mode

(Default: 0x0)

When DTTMG0[1:0] is set to 0x0 (default), I²S mode is selected. In this mode, the first bit of each data is output after one I2S_SCK clock delay from the I2S_WS signal edge.



(MSB first, number of bit clock cycles = 26 or 18)

Figure VI.5.8.5 Data Output Timing 1 (I²S Mode)

When DTTMG0[1:0] is set to 0x1, left justified mode is selected. In this mode, each data output will start at the I2S_WS signal edge.

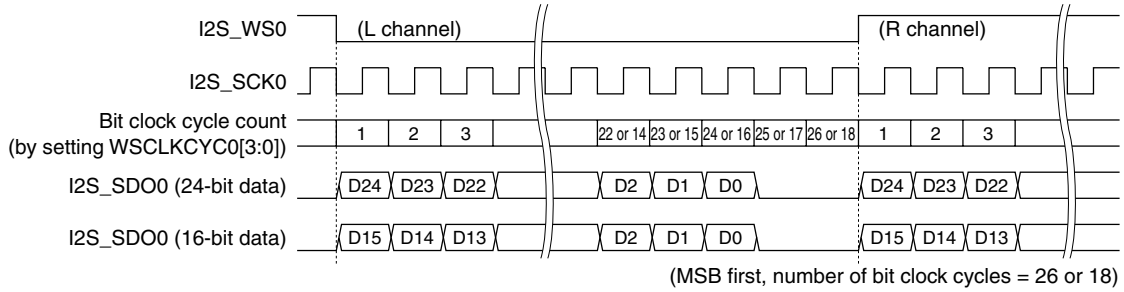


Figure VI.5.8.6 Data Output Timing 2 (Left Justified Mode)

When DTTMG0[1:0] is set to 0x2, right justified mode is selected. In this mode, output data will be right justified to the I2S_WS signal edge.

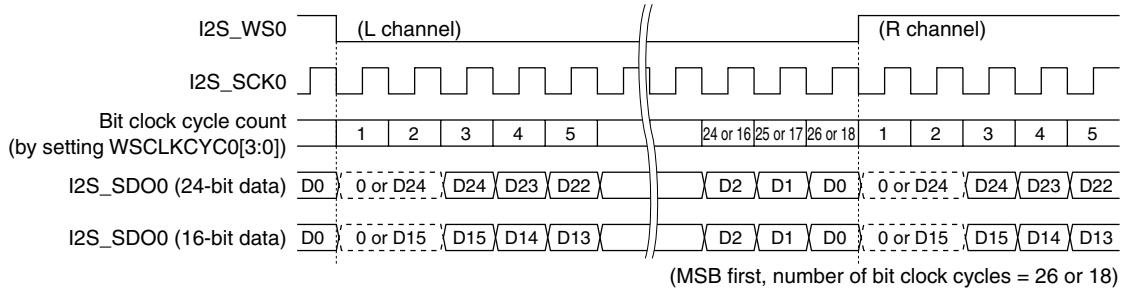


Figure VI.5.8.7 Data Output Timing 3 (Right Justified Mode)

Note: When using right justified mode, the number of bit clock cycles (sample clock period) must be equal to or greater than [Data bit size + 2].

D[1:0] CHMD[1:0]: I²S CH.0 Output Channel Mode Select Bits

Selects the I²S CH.0 output channel mode.

Table VI.5.8.3 Selecting Output Channel Mode

CHMD[1:0]	Output channel mode	L channel	R channel
0x3	Mute	0	0
0x2	Mono (L)	Data output	0
0x1	Mono (R)	0	Data output
0x0	Stereo	Data output	Data output

(Default: 0x0)

The output channel mode can be switched even if data is being output. In this case, the mode changes after the current word output has finished.

When mute mode is selected, the I2S_SDO0 pin is fixed at 0. However, the FIFO and shift register run the same as stereo mode and three clock signals are output normally. Also in mono mode, the I2S_SDO0 pin is fixed at 0 during the output period for the unselected channel.

The FIFO data is read out normally, therefore an interrupt occurs.

0x5302: I²S CH.1 Control Register (I2S_CTL_IN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S CH.1 Control Register (I2S_CTL_IN)	0x5302 (16 bits)	D15–6	–	reserved	–	–	–	0 when being read.
		D5	WCLKMD1	I²S CH.1 input word clock mode select	1 L: high R: low / 0 L: low R: high	0	R/W	
		D4	BCLKPOL1	I²S CH.1 input bit clock polarity select	1 Negative / 0 Positive	0	R/W	
		D3–2	DTTMG1 [1:0]	I²S CH.1 input data timing select	DTTMG1[1:0]: 0x3 reserved; 0x2 Right justified; 0x1 Left justified; 0x0 I²S	0x0	R/W	
		D1	I2SBYPASS	I²S bypass mode select	1 Bypass / 0 Normal	0	R/W	
		D0	I2SEN1	I²S CH.1 enable	1 Enable / 0 Disable	0	R/W	

Note: All the data transfer conditions must be set using this register before setting I2SSTART1 (D8/ I2S_START register) to start data input to the I²S CH.1.

D[15:6] Reserved

D5 WCLKMD1: I²S CH.1 Input Word Clock Mode Select Bit

Selects the I2S_WS input signal level for indicating a channel.
1 (R/W): High = L channel, Low = R channel
0 (R/W): High = R channel, Low = L channel (default)

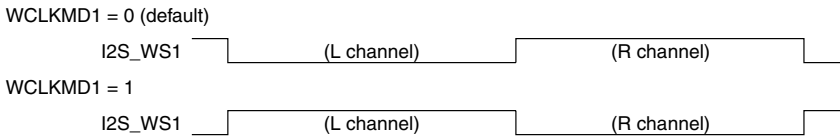


Figure VI.5.8.8 Selecting Word Clock Mode

D4 BCLKPOL1: I²S CH.1 Input Bit Clock Polarity Select Bit

Selects the bit clock polarity.
1 (R/W): Negative
0 (R/W): Positive (default)

When BCLKPOL1 is 0, the I2S_SDI1 input changes at the falling edge of the I2S_SCK clock (bit clock) and the I²S CH.1 samples the data bit at the rising edge of I2S_SCK.
When BCLKPOL1 is set to 1, the I2S_SDI1 input changes at the rising edge of I2S_SCK and the I²S CH.1 samples the data bit at the falling edge of I2S_SCK.

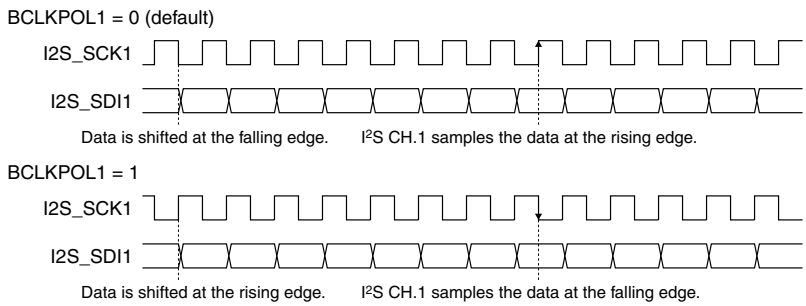


Figure VI.5.8.9 Selecting Bit Clock Polarity

D[3:2] DTTMG1[1:0]: I²S CH.1 Input Data Timing Select Bits

Selects the data bit input timing.

Table VI.5.8.4 Data Input Timing

DTTMG1[1:0]	Data input timing mode
0x3	Reserved
0x2	Right justified mode
0x1	Left justified mode
0x0	I²S mode

(Default: 0x0)

When DTTMG1[1:0] is set to 0x0 (default), I²S mode is selected. In this mode, the first bit of each data is input after one I2S_SCK clock delay from the I2S_WS signal edge.

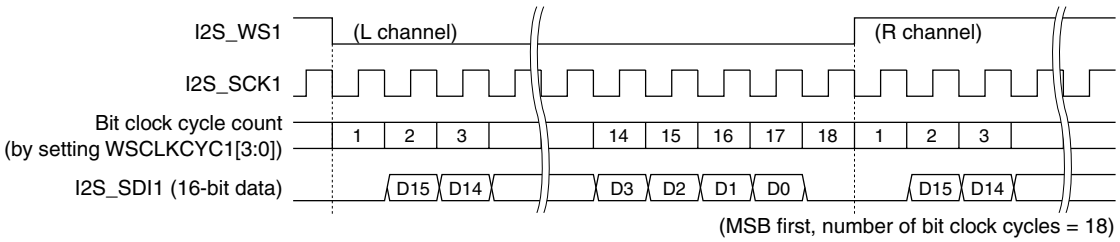


Figure VI.5.8.10 Data Input Timing 1 (I²S Mode)

When DTTMG1[1:0] is set to 0x1, left justified mode is selected. In this mode, each data input will start at the I2S_WS signal edge.

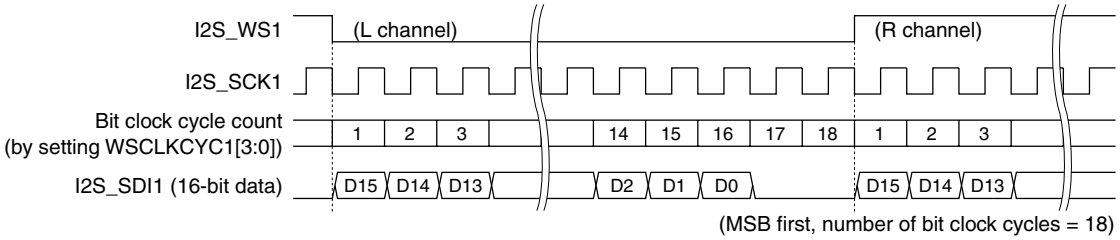


Figure VI.5.8.11 Data Input Timing 2 (Left Justified Mode)

When DTTMG1[1:0] is set to 0x2, right justified mode is selected. In this mode, input data is right justified to the I2S_WS signal edge.

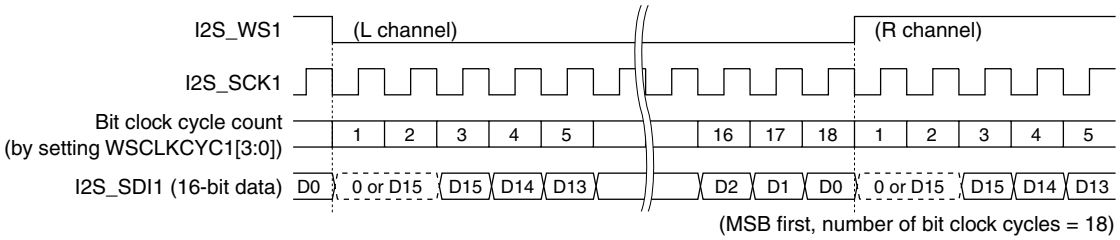


Figure VI.5.8.12 Data Input Timing 3 (Right Justified Mode)

D1 I2SBYPASS: I²S Bypass Mode Select Bit

Selects I²S bypass mode.
1 (R/W): Bypass mode
0 (R/W): Normal mode (default)

Set I2SBYPASS to 1 to set the I²S module to bypass mode. In this mode, the signals input from the I2S_MCLK1, I2S_SCK1, I2S_WS1, and I2S_SDI1 pins can be directly output from the I2S_MCLK0, I2S_SCK0, I2S_WS0, and I2S_SDO0 pins, respectively. When I2SBYPASS is 0 (default), I²S CH.0 and I²S CH.1 can be used independently.

D0 I2SEN1: I²S CH.1 Enable Bit

Enables/disables operation of the I²S CH.1.
1 (R/W): Enable (on)
0 (R/W): Disable (off) (default)

When I2SEN1 is set to 1, the I²S CH.1 starts operating and data transfer is enabled.
When I2SEN1 is set to 0, the I²S CH.1 goes off.

0x5304: I²S MCLK Divide Ratio Register (I2S_DV_MCLK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S MCLK Divide Ratio Register (I2S_DV_MCLK)	0x5304 (16 bits)	D15	MCLKSEL	I2S_MCLK source clock select	1 I2S_MCLK0 input clock 0 System clock	0	R/W	
		D14–6	–	reserved	–	–	–	0 when being read.
		D5–0	MCLKDIV [5:0]	I2S_MCLK divide ratio select	MCLKDIV[5:0] / I2S_MCLK 0x3f PCLK•1/64 0x3e PCLK•1/63 0x3d PCLK•1/62 : : 0x2 PCLK•1/3 0x1 PCLK•1/2 0x0 PCLK•1/1	0x0	R/W	

D15 MCLKSEL: I2S_MCLK Source Clock Select Bit

Selects the source clock for the master clock (I2S_MCLK0) of I²S CH.0.
1 (R/W): I2S_MCLK0 pin input clock
0 (R/W): System clock (PCLK) (default)

When MCLKSEL is set to 0 (default), the I²S module generates the master clock (I2S_MCLK) from PCLK using a frequency divider. When MCLKSEL is set to 1, the I²S CH.0 uses the external clock input from the I2S_MCLK0 pin as the master clock.

D[14:6] Reserved

D[5:0] MCLKDIV[5:0]: I2S_MCLK Divide Ratio Select Bits

Configures the I²S master clock (I2S_MCLK) to be output from the I2S_MCLK0 pin.
The I²S module generates the I2S_MCLK by dividing the operating clock (PCLK generated by the CMU). Specify the divide ratio using MCLKDIV[5:0].

Table VI.5.8.5 Setting I2S_MCLK (Master Clock)

MCLKDIV[5:0]	I2S_MCLK
0x3f	PCLK•1/64
0x3e	PCLK•1/63
0x3d	PCLK•1/62
:	:
0x2	PCLK•1/3
0x1	PCLK•1/2
0x0	PCLK•1/1

(Default: 0x0)

0x5306: I²S Audio Clock Divide Ratio Register (I2S_DV_AUDIO_CLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
I²S Audio Clock Divide Ratio Register (I2S_DV_AUDIO_CLK)	**0x5306** (16 bits)	D15–12	**WSCLKCYC1 [3:0]**	I²S CH.1 WS clock cycle setup	WSCLKCYCx[3:0]	Clock period	0x0	R/W	
					0xf	31 clocks			
					0xe	30 clocks			
					0xd	29 clocks			
					0xc	28 clocks			
					0xb	27 clocks			
					0xa	26 clocks			
					0x9	25 clocks			
					0x8	24 clocks			
		D11–8	**WSCLKCYC0 [3:0]**	I²S CH.0 WS clock cycle setup	0x7	23 clocks	0x0	R/W	
					0x6	22 clocks			
					0x5	21 clocks			
					0x4	20 clocks			
					0x3	19 clocks			
					0x2	18 clocks			
					0x1	17 clocks			
					0x0	16 clocks			
		D7–0	**BCLKDIV [7:0]**	I²S CH.0 bit clock divide ratio select	BCLKDIV[7:0]	Bit clock	0x0	R/W	SRC_CLK: PCLK or I2S_MCLK0 input clock
					0xff	SRC_CLK•1/512			
					0xfe	SRC_CLK•1/510			
					0xfd	SRC_CLK•1/508			
					:	:			
					0x2	SRC_CLK•1/6			
					0x1	SRC_CLK•1/4			
					0x0	SRC_CLK•1/2			

D[15:12] WSCLKCYC1[3:0]: I²S CH.1 WS Clock Cycle Setup Bits

The I²S CH.1 inputs the sample clock from the I2S_WS1 pin. The clock period must be specified with the number of bit clock cycles using WSCLKCYC1[3:0]. See the description of WSCLKCYC0[3:0].

D[11:8] WSCLKCYC0[3:0]: I²S CH.0 WS Clock Cycle Setup Bits

Specifies the sample clock (I2S_WS signal) period.

The I²S CH.0 generates the sample clock to be output from the I2S_WS0 pin by counting the bit clock configured with BCLKDIV[7:0] (D[7:0]). Specify the half cycle (a high or low level period) of the I2S_WS clock with the number of bit clock cycles using WSCLKCYC0[3:0].

Table VI.5.8.6 Setting the Sample Clock

WSCLKCYC*x*[3:0]	**Sample clock period (number of bit clock cycles)**
0xf	31 clocks
0xe	30 clocks
0xd	29 clocks
0xc	28 clocks
0xb	27 clocks
0xa	26 clocks
0x9	25 clocks
0x8	24 clocks
0x7	23 clocks
0x6	22 clocks
0x5	21 clocks
0x4	20 clocks
0x3	19 clocks
0x2	18 clocks
0x1	17 clocks
0x0	16 clocks

(Default: 0x0)

The sampling clock frequency is calculated as below.

$$fs = \frac{f_{I2S\_SCK}}{n \times 2} \text{ [Hz]}$$

fs: Sampling clock frequency [Hz]
f_{I2S_SCK}: Bit clock frequency [Hz] (CH.0: See Table VI.5.8.7. CH.1: I2S_SCK1 input clock frequency)
n: Number of bit clocks selected by WSCLKCYC*x*[3:0] (See Table VI.5.8.6.)

Note: The value to be set to the WSCLKCYC*x*[3:0] is not the number of audio data bits, but the number of bit clock cycles that is used to adjust the sample clock period. It must be equal to or greater than the number of audio data bits (24 bits or 16 bits).

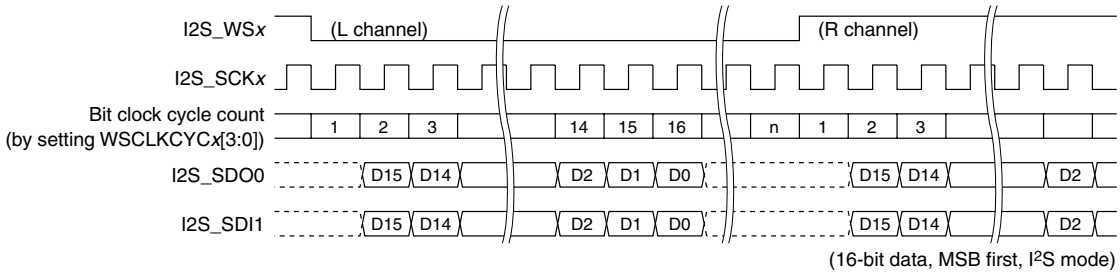


Figure VI.5.8.13 Sample Clock Period

D[7:0] BCLKDIV[7:0]: I²S CH.0 Bit Clock Divide Ratio Select Bits

Configures the bit clock to be output.
The I²S module generates the bit clock to be output from the I2S_SCK0 pin of the I²S CH.0 by dividing the source clock selected for the I2S_MCLK output. Specify the divide ratio using BCLKDIV[7:0].

Table VI.5.8.7 Setting Output Bit Clock

BCLKDIV[7:0]	Bit clock (I2S_SCK0)
0xff	SRC_CLK•1/512
0xfe	SRC_CLK•1/510
0xfd	SRC_CLK•1/508
:	:
0x2	SRC_CLK•1/6
0x1	SRC_CLK•1/4
0x0	SRC_CLK•1/2

(SRC_CLK = PCLK or I2S_MCLK0 input clock, default: 0x0)

The I²S CH.0 bit clock frequency is calculated as below.

$$f_{I2S\_SCK0} = \frac{f_{SRC\_CLK}}{(BCLKDIV + 1) \times 2} \text{ [Hz]}$$

f_{I2S_SCK0}: I²S CH.0 bit clock frequency [Hz]
f_{SRC_CLK}: PCLK or I2S_MCLK0 input clock frequency [Hz]
BCLKDIV: BCLKDIV[7:0] set value (0x0–0xff)

I²S CH.1 uses the bit clock input from the I2S_SCK1 pin, therefore the above setting is not applied to CH.1.

0x5308: I²S Start/Stop Register (I2S_START)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Start/Stop Register (I2S_START)	0x5308 (16 bits)	D15–9	–	reserved	–	–	–	0 when being read.
		D8	I2SSTART1	I²S CH.1 start/stop control	1 Start 0 Stop	0	R/W	
		D7	I2SBUSY0	I²S CH.0 busy flag	1 Busy 0 Idle	0	R	
		D6–1	–	reserved	–	–	–	0 when being read.
		D0	I2SSTART0	I²S CH.0 start/stop control	1 Start (run) 0 Stop	0	R/W	

D[15:9] Reserved

D8 I2SSTART1: I²S CH.1 Start/Stop Control Bit

Starts/stops data receiving of I²S CH.1.
1 (R/W): Start
0 (R/W): Stop (default)

Writing 1 to I2SSTART1 starts serial data reception through the I2S_SDI1 pin.
Writing 0 to I2SSTART1 stops receive operation immediately. After that, the data sent from the external I²S device will be ignored. This bit is read as 1 when the I²S CH.1 is receiving data or is read as 0 when the I²S CH.1 is not receiving data.

D7 I2SBUSY0: I²S CH.0 Busy Flag Bit

Indicates the data output status of the I²S module.
1 (R): Busy
0 (R): Idle (default)

I2SBUSY0 is set to 1 when the I²S CH.0 starts data output and stays 1 while data is being output. This flag is cleared to 0 upon completion of the output operation.

D[6:1] Reserved

D0 I2SSTART0: I²S CH.0 Start/Stop Control Bit

Starts/stops data output of the I²S CH.0.
1 (R/W): Start
0 (R/W): Stop (default)

Writing 1 to I2SSTART0 starts serial data transmission through the I2S_SDO0 pin.
Writing 0 to I2SSTART0 stops transmission after all the data currently stored in the FIFO have been output. After I2SSTART0 is set to 0, new transmit data cannot be written to the FIFO.

Note: Be sure to avoid altering the I2S_DV_MCLK (0x5304) and I2S_DV_AUDIO_CLK (0x5306) registers when I2SSTART0 is 1.

0x530a: I²S FIFO Status Register (I2S_FIFO_STAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S FIFO Status Register (I2S_FIFO_STAT)	0x530a (16 bits)	D15–10	–	reserved	–	–	–	0 when being read.
		D9	I2SFIFOFF1	I²S CH.1 FIFO full flag	1 Full 0 Not full	0	R	
		D8	I2SFIFOEF1	I²S CH.1 FIFO empty flag	1 Empty 0 Not empty	1	R	
		D7–5	–	reserved	–	–	–	0 when being read.
		D4–2	FIFOSTAT0 [2:0]	I²S CH.0 FIFO state machine	FIFOSTAT0[2:0] State 0x7–0x6 reserved 0x5 FLUSH 0x4 EMPTY 0x3 LACK 0x2 FULL 0x1 INIT 0x0 STOP	0x0	R	
		D1	I2SFIFOFF0	I²S CH.0 FIFO full flag	1 Full 0 Not full	0	R	
		D0	I2SFIFOEF0	I²S CH.0 FIFO empty flag	1 Empty 0 Not empty	1	R	

D[15:10] Reserved

D9 I2SFIFOFF1: I²S CH.1 FIFO Full Flag Bit

Indicates whether the receive FIFO is full or not.

1 (R): Full

0 (R): Not full (default)

I2SFIFOFF1 is set to 1 when the FIFO becomes full of the received data (16 bits × 2 channels (L & R) × 4). In this case, it is necessary to read out the received data from the FIFO, otherwise, the newest data in the FIFO will be overwritten with a new data received.

I2SFIFOFF1 is reset to 0 by reading the stored data.

D8 I2SFIFOEF1: I²S CH.1 FIFO Empty Flag Bit

Indicates whether the receive FIFO is empty or not.

1 (R): Empty (default)

0 (R): Not empty

I2SFIFOEF1 is reset to 0 when a received data is written to the FIFO and is set to 1 when all the stored data are read out.

D[7:5] Reserved

D[4:2] FIFOSTAT0[2:0]: I²S CH.0 FIFO State Machine Bits

Indicates the transmit FIFO status.

Table VI.5.8.8 Monitoring the FIFO State Machine

FIFOSTAT0[2:0]	State
0x7–0x6	Reserved
0x5	FLUSH: FIFO is flushing the remained audio data before it stops.
0x4	EMPTY: FIFO is empty.
0x3	LACK: FIFO is not full and not empty.
0x2	FULL: FIFO is full.
0x1	INIT: Initialize all four entries of FIFO.
0x0	STOP: FIFO is idle.

(Default: 0x0)

D1 I2SFIFOFF0: I²S CH.0 FIFO Full Flag Bit

Indicates whether the transmit FIFO is full or not.

1 (R): Full

0 (R): Not full (default)

I2SFIFOFF0 is set to 1 when the FIFO becomes full of the written data (16 or 24 bits × 2 channels (L & R) × 4) to indicate that no more data can be written.

I2SFIFOFF0 is reset to 0 when the stored data is read out to transmit.

D0 I2SFIFOEF0: I²S CH.0 FIFO Empty Flag Bit

Indicates whether the transmit FIFO is empty or not.

1 (R): Empty (default)

0 (R): Not empty

I2SFIFOEF0 is reset to 0 when a transmit data is written to the FIFO and is set to 1 when all the stored data have been transmitted.

0x530c: I²S Interrupt Mode Select Register (I2S_INT_MOD)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
I²S Interrupt Mode Select Register (I2S_INT_MOD)	0x530c (16 bits)	D15–6	–	reserved	–				–	–	0 when being read.
		D5–4	I2SINTMD1 [1:0]	I²S CH.1 interrupt mode select	I2SINTMD1[1:0]	State			0x0	R/W	
					0x3	reserved					
					0x2	One data					
					0x1	Whole full					
					0x0	Half full					
		D3–2	I2SINTMD0 [1:0]	I²S CH.0 interrupt mode select	I2SINTMD0[1:0]	State			0x0	R/W	
					0x3	reserved					
					0x2	One empty					
					0x1	Whole empty					
					0x0	Half empty					
		D1	I2SINTEN1	I²S CH.1 interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	I2SINTEN0	I²S CH.0 interrupt enable	1	Enable	0	Disable	0	R/W	

D[15:6] Reserved

D[5:4] I2SINTMD1[1:0]: I²S CH.1 Interrupt Mode Select Bits

Selects the interrupt mode for I²S CH.1.

Table VI.5.8.9 Selecting I²S CH.1 Interrupt Mode

I2SINTMD1[1:0]	Interrupt mode
0x3	Reserved
0x2	One data interrupt mode
0x1	Whole full interrupt mode
0x0	Half full interrupt mode

(Default: 0x0)

Whole full interrupt mode

While audio data is being input in this mode, the I²S CH.1 generates an interrupt after four received stereo data have been written to the FIFO. In other words, the FIFO is full when an interrupt occurs. Therefore, the application program needs to read four stereo data (16 bits × 2 channels (L & R) × 4) from the FIFO at once after an interrupt occurs.

Half full interrupt mode (default)

In this mode, the I²S CH.1 generates an interrupt after two received stereo data have been written to the FIFO. In this case, the FIFO may be full or it may contain two or three received data (the FIFO status can be checked using the status bits). The application program needs to read two stereo data (16 bits × 2 channels (L & R) × 2) from the FIFO at once after an interrupt occurs.

One data interrupt mode

In this mode, the I²S CH.1 generates an interrupt after one received stereo data has been written to the FIFO. In this case, the FIFO may be full or it may contain one to three data (the FIFO status can be checked using the status bits). The application program needs to read one stereo data (16 bits × 2 channels (L & R) × 1) from the FIFO at once after an interrupt occurs.

D[3:2] I2SINTMD0[1:0]: I²S CH.0 Interrupt Mode Select Bits

Selects the interrupt mode for I²S CH.0.

Table VI.5.8.10 Selecting I²S CH.0 Interrupt Mode

I2SINTMD0[1:0]	Interrupt mode
0x3	Reserved
0x2	One empty interrupt mode
0x1	Whole empty interrupt mode
0x0	Half empty interrupt mode

(Default: 0x0)

Whole empty interrupt mode

While audio data is being output in this mode, the I²S CH.0 generates an interrupt after all data (four stereo data) has been read out from the FIFO to transmit. In other words, the FIFO is empty when an interrupt occurs. Therefore, the application program needs to fill the FIFO with four stereo data (24 or 16 bits × 2 channels (L & R) × 4) at once after an interrupt occurs.

Half empty interrupt mode (default)

In this mode, the I²S CH.0 generates an interrupt after two stereo data has been read out from the FIFO to transmit. In this case, the FIFO may be empty or it may still contain one or two data remained (the FIFO status can be checked using the status bits). The application program needs to fill the FIFO with two stereo data (24 or 16 bits × 2 channels (L & R) × 2) at once after an interrupt occurs.

One empty interrupt mode

In this mode, the I²S CH.0 generates an interrupt after one stereo data has been read out from the FIFO to transmit. In this case, the FIFO may be empty or it may still contain one to three data remained (the FIFO status can be checked using the status bits). The application program needs to fill the FIFO with one stereo data (24 or 16 bits × 2 channels (L & R) × 1) at once after an interrupt occurs.

D1 I2SINTEN1: I²S CH.1 Interrupt Enable Bit

Enables/disables I²S CH.1 interrupt caused by receive FIFO full.
1 (R/W): Enable
0 (R/W): Disable (default)

When I2SINTEN1 is set to 1, I²S CH.1 (FIFO full) interrupt requests to the ITC are enabled. A FIFO full interrupt request occurs according to the interrupt mode set with I2SINTMD1[1:0] (D[5:4]).
When I2SINTEN1 is set to 0, I²S CH.1 interrupts will not be generated.

D0 I2SINTEN0: I²S CH.0 Interrupt Enable Bit

Enables/disables I²S CH.0 interrupt caused by transmit FIFO empty.
1 (R/W): Enable
0 (R/W): Disable (default)

When I2SINTEN0 is set to 1, I²S CH.0 (FIFO empty) interrupt requests to the ITC are enabled. A FIFO empty interrupt request occurs according to the interrupt mode set with I2SINTMD0[1:0] (D[3:2]).
When I2SINTEN0 is set to 0, I²S CH.0 interrupts will not be generated.

0x5310: I²S CH.0 FIFO Register (I2S_FIFO_OUT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S CH.0 FIFO Register (I2S_FIFO_OUT)	**0x5310** (24 bits)	D23–0	**I2SFIFO0 [23:0]**	I²S CH.0 FIFO (output data)	0 to 0xffffff (24 bits)	0x0	W	for 24-bit data 0 when being read.
	0x5310 (16 bits)	D15–0	**I2SFIFO0 [15:0]**		0 to 0xffff (16 bits)			for 16-bit data 0 when being read.

Note: This register must be accessed using a 24-bit access instruction (`ld.a [%rb],%rs`) for writing 24-bit audio data or a 16-bit access instruction (`ld [%rb],%rs`) for writing 16-bit audio data.

D[15:0] I2SFIFO0[15:0]: I²S CH.0 FIFO (Output Data) Bits

Write output data to the FIFO through this address.

Up to four stereo data (24 or 16 bits × 2 channels (L & R) × 4) can be written to the FIFO regardless of the data size. Before starting audio data output, fill the FIFO with the first four stereo data.

When the data size is 16 bits, use a 16-bit memory write (`ld [%rb],%rs`) instruction for writing data. Note that 8-bit and 24-bit memory write instructions cannot be used when the data size is 16 bits.

When the data size is 24 bits, use a 24-bit memory write (`ld.a [%rb],%rs`) instruction for writing data. In this case, two memory write access cycles are generated for writing one channel data. Note that 8-bit and 16-bit memory write instructions cannot be used when the data size is 24 bits.

First write L-channel data, then R channel data. Both channel data must be written as a pair even if "mono" is selected as the output channel mode.

0x5314: I²S CH.1 FIFO Register (I2S_FIFO_IN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S CH.1 FIFO Register (I2S_FIFO_IN)	**0x5314** (16 bits)	D15–0	**I2SFIFO1 [15:0]**	I²S CH.1 FIFO (input data)	0 to 0xffff	0x0	R	

Note: This register must be accessed using a 16-bit access instruction (`ld %rd,[%rb]`) for reading 16-bit audio data.

D[15:0] I2SFIFO1[15:0]: I²S CH.1 FIFO (Input Data) Bits

Read receive data from the FIFO through this address.

Up to four stereo data (16 bits × 2 channels (L & R) × 4) can be stored in the FIFO.

Use a 16-bit memory read (`ld %rd,[%rb]`) instruction for reading data from the FIFO. Note that 8-bit and 24-bit memory read instructions cannot be used.

Both channel data should be read as a pair. L-channel data is read first, then R-channel data.

VI.5.9 Precautions

- Always make sure the I²S module is not started (I2SSTART0 (D0/I2S_START register)/I2SSTART1 (D8/I2S_START register) = 0) before the I²S settings are made. A change of settings during operation may cause a malfunction.
  - ∗ **I2SSTART0**: I²S CH.0 Start/Stop Control Bit in the I²S Start/Stop (I2S_START) Register (D0/0x5308)
  - ∗ **I2SSTART1**: I²S CH.1 Start/Stop Control Bit in the I²S Start/Stop (I2S_START) Register (D8/0x5308)
- When using right justified mode, the number of bit clock cycles (sample clock period) must be equal to or greater than [Data bit size + 2].
- The I2S_FIFO_OUT (0x5310) register must be accessed using a 24-bit access instruction (`ld.a [%rb],%rs`) for writing 24-bit audio data or a 16-bit access instruction (`ld [%rb],%rs`) for writing 16-bit audio data.
- The I2S_FIFO_IN (0x5314) register must be accessed using a 16-bit access instruction (`ld %rd,[%rb]`) for reading 16-bit audio data.

VI.5.10 Setting the I²S Clocks

This section explains how to configure the I2S_MCLK, I2S_WS, and I2S_SCK clocks.

When using the internal system clock as the source clock to generate I2S_MCLK

The following shows how to determine the clock setting values from the sampling rate. The example below assumes that the system clock frequency is 48 MHz and the sampling rate of audio data is 44.1 kHz.
The sample clock (I2S_WS) is in sync with the master clock (I2S_MCLK), so the following equation is formulated:

$$\frac{f_{I2S\_MCLK}}{f_{I2S\_WS}} = \text{Integer}$$

where f_{I2S_MCLK} is the output master clock (I2S_MCLK) frequency and f_{I2S_WS} is the sample clock (I2S_WS) frequency.

$$f_{I2S\_MCLK} = \frac{48\ \text{MHz}}{\text{MCLKDIV}[5:0] + 1} \qquad \text{(eq1)}$$

$$f_{I2S\_WS} = \frac{48\ \text{MHz}}{(\text{BCLKDIV}[7:0] + 1) \times 2 \times (\text{WSCLKCYC0}[3:0] + 16) \times 2} \qquad \text{(eq2)}$$

$$\frac{(\text{BCLKDIV}[7:0] + 1) \times 2 \times (\text{WSCLKCYC0}[3:0] + 16) \times 2}{\text{MCLKDIV}[5:0] + 1} = \text{Integer} \qquad \text{(eq3)}$$

* **MCLKDIV[5:0]**: I2S_MCLK Divide Ratio Select Bits in the I²S MCLK Divide Ratio (I2S_DV_MCLK) Register (D[5:0]/0x5304)
* **BCLKDIV[7:0]**: I²S CH.0 Bit Clock Divide Ratio Select Bits in the I²S Audio Clock Divide Ratio (I2S_DV_AUDIO_CLK) Register (D[7:0]/0x5306)
* **WSCLKCYC0[3:0]**: I²S CH.0 WS Clock Cycle Setup Bits in the I²S Audio Clock Divide Ratio (I2S_DV_AUDIO_CLK) Register (D[11:8]/0x5306)

Table VI.5.10.1 Setting I2S_MCLK (Master Clock)

MCLKDIV[5:0]	I2S_MCLK
0x3f	PCLK•1/64
0x3e	PCLK•1/63
0x3d	PCLK•1/62
:	:
0x2	PCLK•1/3
0x1	PCLK•1/2
0x0	PCLK•1/1

Table VI.5.10.2 Setting the Bit Clock

BCLKDIV[7:0]	Bit clock (I2S_SCK0)
0xff	SRC_CLK•1/512
0xfe	SRC_CLK•1/510
0xfd	SRC_CLK•1/508
:	:
0x2	SRC_CLK•1/6
0x1	SRC_CLK•1/4
0x0	SRC_CLK•1/2

Table VI.5.10.3 Setting the Sample Clock Period

WSCLKCYC*x*[3:0]	Sample clock period (number of bit clock cycles)
0xf	31 clocks
0xe	30 clocks
0xd	29 clocks
0xc	28 clocks
0xb	27 clocks
0xa	26 clocks
0x9	25 clocks
0x8	24 clocks
0x7	23 clocks
0x6	22 clocks
0x5	21 clocks
0x4	20 clocks
0x3	19 clocks
0x2	18 clocks
0x1	17 clocks
0x0	16 clocks

The table below is made from Equitation 2 (eq2) using Excel.
The value filled in cell A1 is the system clock frequency (48 MHz). The other values (32–62) in the first horizontal line (B1–Q1) are number of bit clock cycles ((WSCLKCYC0[3:0] + 16) × 2) equivalent to a sample clock cycle. The first vertical line (A2–A257) shows all the settings of bit clock divide ratio ((BCLKDIV[7:0] + 1) × 2).

Table VI.5.10.4 List of Sample Clock Frequencies

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	48	32	**34**	36	38	40	42	44	46	48	50	52	54	56	58	60	62
2	2	750.00	705.88	666.67	631.58	600.00	571.43	545.45	521.74	500.00	480.00	461.54	444.44	428.57	413.79	400.00	387.10
3	4	375.00	352.94	333.33	315.79	300.00	285.71	272.73	260.87	250.00	240.00	230.77	222.22	214.29	206.90	200.00	193.55
4	6	250.00	235.29	222.22	210.53	200.00	190.48	181.82	173.91	166.67	160.00	153.85	148.15	142.86	137.93	133.33	129.03
5	8	187.50	176.47	166.67	157.89	150.00	142.86	136.36	130.43	125.00	120.00	115.38	111.11	107.14	103.45	100.00	96.77
6	10	150.00	141.18	133.33	126.32	120.00	114.29	109.09	104.35	100.00	96.00	92.31	88.89	85.71	82.76	80.00	77.42
7	12	125.00	117.65	111.11	105.26	100.00	95.24	90.91	86.96	83.33	80.00	76.92	74.07	71.43	68.97	66.67	64.52
8	14	107.14	100.84	95.24	90.23	85.71	81.63	77.92	74.53	71.43	68.57	65.93	63.49	61.22	59.11	57.14	55.30
9	16	93.75	88.24	83.33	78.95	75.00	71.43	68.18	65.22	62.50	60.00	57.69	55.56	53.57	51.72	50.00	48.39
10	18	83.33	78.43	74.07	70.18	66.67	63.49	60.61	57.97	55.56	53.33	51.28	49.38	47.62	45.98	44.44	43.01
11	20	75.00	70.59	66.67	63.16	60.00	57.14	54.55	52.17	50.00	48.00	46.15	44.44	42.86	41.38	40.00	38.71
12	22	68.18	64.17	60.61	57.42	54.55	51.95	49.59	47.43	45.45	43.64	41.96	40.40	38.96	37.62	36.36	35.19
13	24	62.50	58.82	55.56	52.63	50.00	47.62	45.45	43.48	41.67	40.00	38.46	37.04	35.71	34.48	33.33	32.26
14	26	57.69	54.30	51.28	48.58	46.15	43.96	41.96	40.13	38.46	36.92	35.50	34.19	32.97	31.83	30.77	29.78
15	28	53.57	50.42	47.62	45.11	42.86	40.82	38.96	37.27	35.71	34.29	32.97	31.75	30.61	29.56	28.57	27.65
16	30	50.00	47.06	44.44	42.11	40.00	38.10	36.36	34.78	33.33	32.00	30.77	29.63	28.57	27.59	26.67	25.81
17	**32**	46.88	**44.12**	41.67	39.47	37.50	35.71	34.09	32.61	31.25	30.00	28.85	27.78	26.79	25.86	25.00	24.19
18	34	44.12	41.52	39.22	37.15	35.29	33.61	32.09	30.69	29.41	28.24	27.15	26.14	25.21	24.34	23.53	22.77
19	36	41.67	39.22	37.04	35.09	33.33	31.75	30.30	28.99	27.78	26.67	25.64	24.69	23.81	22.99	22.22	21.51
20	38	39.47	37.15	35.09	33.24	31.58	30.08	28.71	27.46	26.32	25.26	24.29	23.39	22.56	21.78	21.05	20.37
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
256	510	2.94	2.77	2.61	2.48	2.35	2.24	2.14	2.05	1.96	1.88	1.81	1.74	1.68	1.62	1.57	1.52
257	512	2.93	2.76	2.60	2.47	2.34	2.23	2.13	2.04	1.95	1.88	1.80	1.74	1.67	1.62	1.56	1.51

↑ ↑

B2 = A1 ∗ 1000/(B1 ∗ A2)
:
B257 = A1 ∗ 1000/(B1 ∗ A257)

L2 = A1 ∗ 1000/(L1 ∗ A2)
:
L257 = A1 ∗ 1000/(L1 ∗ A257)

Find “44.1 (kHz)” or an approximate value from the table. You may choose “44.12” in cell C17. Cells A17 and C1 show 32 and 34 respectively.
So you may get the BCLKDIV[7:0] and WSCLKCYC0[3:0] values as follows:
BCLKDIV[7:0] = (32/2) - 1 = 15 (0xf)
WSCLKCYC0[3:0] = (34/2) - 16 = 1

Substituting these values in Equation 3 (eq3) yields the MCLKDIV[5:0] values.

$$\frac{32 \times 34}{\text{MCLKDIV[5:0]} + 1} = \text{Integer}$$

MCLKDIV[5:0] = 0, 1, 3, 7, 15, 16, 31, 33, 63

Table VI.5.10.5 Master Clock Frequency

MCLKDIV[5:0]	f_{I2S_MCLK}	
0	48 MHz	(1088 fs)
1	24 MHz	(544 fs)
3	12 MHz	(272 fs)
7	6 MHz	(136 fs)
15	3 MHz	(68 fs)
16	2.824 MHz	(64 fs)
31	1.5 MHz	(34 fs)
33	1.412 MHz	(32 fs)
63	0.75 MHz	(17 fs)

When using the external master clock (I2S_MCLK0 input clock)

The following shows how to determine the clock setting values from the sampling rate and the input master clock frequency. The example below assumes that the sampling rate of audio data is 44.1 kHz (= fs) and the master clock frequency is 320 fs.

Master clock frequency = 320 × 44.1 kHz = 14.112 MHz

Find "320 fs" from the table shown below. Either one of the following two settings can be chosen:

1) BCLKDIV[7:0] = 4, WSCLKCYC0[3:0] = 0 (16 cycles)
2) BCLKDIV[7:0] = 3, WSCLKCYC0[3:0] = 4 (20 cycles)

The sample clock (I2S_WS) will be 44.1 kHz.

TableVI.5.10.6 External Master Clock Frequency

BCLK DIV [7:0]	External master clock frequency (fs: sample clock frequency)															
	Number of bit clock cycles (WSCLKCYC0[3:0] + 16)															
	16 cycles	17 cycles	18 cycles	19 cycles	20 cycles	21 cycles	22 cycles	23 cycles	24 cycles	25 cycles	26 cycles	27 cycles	28 cycles	29 cycles	30 cycles	31 cycles
0	64 fs	68 fs	72 fs	76 fs	80 fs	84 fs	88 fs	92 fs	96 fs	100 fs	104 fs	108 fs	112 fs	116 fs	120 fs	124 fs
1	128 fs	136 fs	144 fs	152 fs	160 fs	168 fs	176 fs	184 fs	192 fs	200 fs	208 fs	216 fs	224 fs	232 fs	240 fs	248 fs
2	192 fs	204 fs	216 fs	228 fs	240 fs	252 fs	264 fs	276 fs	288 fs	300 fs	312 fs	324 fs	336 fs	348 fs	360 fs	372 fs
3	256 fs	272 fs	288 fs	304 fs	320 fs	336 fs	352 fs	368 fs	384 fs	400 fs	416 fs	432 fs	448 fs	464 fs	480 fs	496 fs
4	320 fs	340 fs	360 fs	380 fs	400 fs	420 fs	440 fs	460 fs	480 fs	500 fs	520 fs	540 fs	560 fs	580 fs	600 fs	620 fs
5	384 fs	408 fs	432 fs	456 fs	480 fs	504 fs	528 fs	552 fs	576 fs	600 fs	624 fs	648 fs	672 fs	696 fs	720 fs	744 fs
6	448 fs	476 fs	504 fs	532 fs	560 fs	588 fs	616 fs	644 fs	672 fs	700 fs	728 fs	756 fs	784 fs	812 fs	840 fs	868 fs
7	512 fs	544 fs	576 fs	608 fs	640 fs	672 fs	704 fs	736 fs	768 fs	800 fs	832 fs	864 fs	896 fs	928 fs	960 fs	992 fs
8	576 fs	612 fs	648 fs	684 fs	720 fs	756 fs	792 fs	828 fs	864 fs	900 fs	936 fs	972 fs	1008 fs	1044 fs	1080 fs	1116 fs
9	640 fs	680 fs	720 fs	760 fs	800 fs	840 fs	880 fs	920 fs	960 fs	1000 fs	1040 fs	1080 fs	1120 fs	1160 fs	1200 fs	1240 fs
10	704 fs	748 fs	792 fs	836 fs	880 fs	924 fs	968 fs	1012 fs	1056 fs	1100 fs	1144 fs	1188 fs	1232 fs	1276 fs	1320 fs	1364 fs
11	768 fs	816 fs	864 fs	912 fs	960 fs	1008 fs	1056 fs	1104 fs	1152 fs	1200 fs	1248 fs	1296 fs	1344 fs	1392 fs	1440 fs	1488 fs
12	832 fs	884 fs	936 fs	988 fs	1040 fs	1092 fs	1144 fs	1196 fs	1248 fs	1300 fs	1352 fs	1404 fs	1456 fs	1508 fs	1560 fs	1612 fs
13	896 fs	952 fs	1008 fs	1064 fs	1120 fs	1176 fs	1232 fs	1288 fs	1344 fs	1400 fs	1456 fs	1512 fs	1568 fs	1624 fs	1680 fs	1736 fs
14	960 fs	1020 fs	1080 fs	1140 fs	1200 fs	1260 fs	1320 fs	1380 fs	1440 fs	1500 fs	1560 fs	1620 fs	1680 fs	1740 fs	1800 fs	1860 fs
15	1024 fs	1088 fs	1152 fs	1216 fs	1280 fs	1344 fs	1408 fs	1472 fs	1536 fs	1600 fs	1664 fs	1728 fs	1792 fs	1856 fs	1920 fs	1984 fs

VI.6 Remote Controller (REMC)

VI.6.1 Outline of the REMC

The S1C17501 is equipped with a remote controller (REMC) module for generating/receiving infrared remote control signals. The REMC module consists of a carrier generator for generating a carrier signal, a 16-bit envelope counter for counting the transmit/receive data length, a modulator for generating transmit data with a designated carrier length, and an edge detector for detecting rising and falling edges from the input signal.
Also the REMC module supports three types of interrupts: (1) counter underflow interrupt that will occur when the envelope counter has reaches 0 indicating that a specified length of data has been sent during transmission or the received data length is too long during reception, (2) input rising edge interrupt that will occur when a rising edge of the input signal (received data) has been detected during reception, and (3) falling edge interrupt that will occur when a falling edge of the input signal has been detected.
Figure VI.6.1.1 shows the structure of the REMC module.

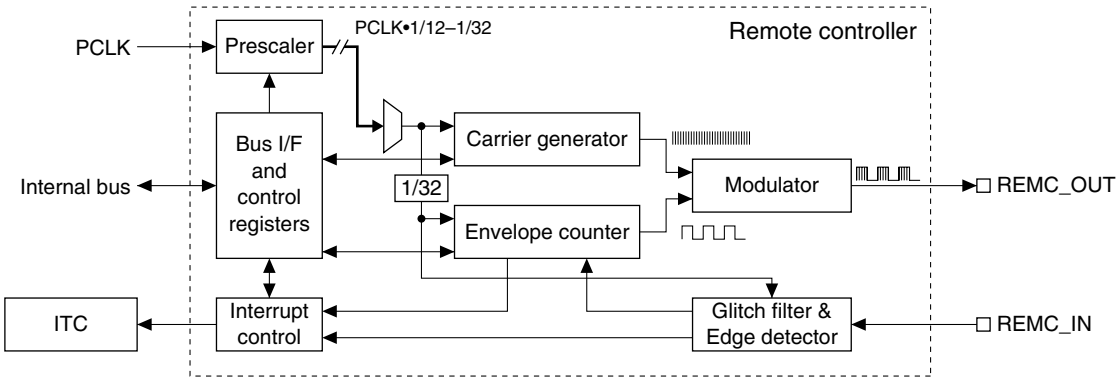


Figure VI.6.1.1 Structure of REMC Module

VI.6.2 REMC I/O Pins

Table VI.6.2.1 lists the REMC input/output pins.

Table VI.6.2.1 List of REMC Pins

Pin name	I/O	Size	Function
REMC_IN	I	1	Remote controller receive data input pin This pin inputs receive data.
REMC_OUT	O	1	Remote controller transmit data output pin This pin outputs the modulated Remote control transmit data.

The REMC input/output pins (REMC_IN, REMC_OUT) are shared with the I/O ports and they are initialized as general-purpose I/O port pins by default. Before using these pins for the REMC, the pin functions must be switched using the Port Function Select Register.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.6.3 Prescaler and Carrier Generator

The REMC module contains a carrier generator with a prescaler that generates a transmit carrier signal according to the H carrier length, and L carrier length set with software.

Setting the prescaler

The REMC module also contains a prescaler that divides the PCLK clock input from the CMU to generate the count clock for the carrier generator. It outputs four clocks, PCLK divided by 12 to PCLK divided by 32. Use REMPSDIV[1:0] (D[1:0]/REMC_PSC register) to select one them.

* **REMPSDIV[1:0]**: REMC Prescaler Division Ratio Select Bits in the REMC Prescaler Control (REMC_PSC) Register (D[1:0]/0x5400)

Table VI.6.3.1 Selecting a Clock for Carrier Generator

REMPSDIV[1:0]	Output clock
0x3	PCLK•1/32
0x2	PCLK•1/24
0x1	PCLK•1/16
0x0	PCLK•1/12

(Default: 0x0)

Most infrared remote carrier frequencies fall within the range from 30 kHz to 56 kHz and their duty ratio is 1/2, 1/3, or 1/4. To insure the accuracy of the carrier frequency, the prescaler output clock frequency should be set to 1 MHz (min.) to 8 MHz (max.).
Examples:

1. When the PCLK frequency is 48 MHz, PCLK•1/24 (2 MHz) is recommended.
2. When the PCLK frequency is 60 MHz, PCLK•1/32 (1.875 MHz) is recommended.

The prescaler is disabled (turned off) by default. Set REMPSON (D2/REMC_PSC register) to 1 to turn the prescaler on before the REMC module can be used.

* **REMPSON**: REMC Prescaler Control Bit in the REMC Prescaler Control (REMC_PSC) Register (D2/0x5400)

Also the prescaler clock is supplied to the envelope counter and the edge detector after it is divided by 32.

If the REMC module is not used, the prescaler should be turned off (REMPSON = 0) to reduce current consumption.

Setting the carrier signal

The high period and low period widths of the carrier signal can be set using CLDH[7:0] (D[15:8]/REMC_CARL register) and CLDL[7:0] (D[7:0]/REMC_CARL register). The carrier high and low widths should be set as the number of clock (selected as above) cycles +1.
Carrier high width [s] = (CLDH[7:0] + 1) / f_{PSOUT} [Hz]
Carrier low width [s] = (CLDL[7:0] + 1) / f_{PSOUT} [Hz]

* **CLDH[7:0]**: REMC Carrier High Width Setup Bits in the REMC Carrier Load (REMC_CARL) Register (D[15:8]/0x540c)
* **CLDL[7:0]**: REMC Carrier Low Width Setup Bits in the REMC Carrier Load (REMC_CARL) Register (D[7:0]/0x540c)

The table below shows examples of carrier settings when the prescaler clock frequency is 2 MHz.

Table VI.6.3.2 Carrier Setting Examples

Prescaler clock frequency	2 MHz		
Carrier frequency	30 kHz	38 kHz	56 kHz
Carrier duty	1/2	1/3	1/4
Ideal period	33.333 µs	26.316 µs	17.857 µs
CLDH[7:0] setting	32	17	8
CLDL[7:0] setting	32	34	26
Period error margin	1%	0.7%	0.8%

The carrier signal is generated as shown in Figure VI.6.3.1.

Example: REMPSDIV[1:0] = 0x2 (PCLK•1/24, 2 MHz), CLDH[7:0] = 17, CLDL[7:0] = 34
(carrier 38 kHz, 1/3 duty)

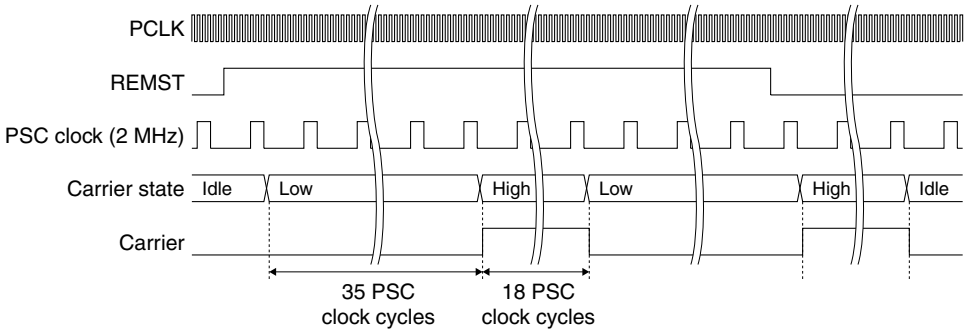


Figure VI.6.3.1 Carrier Signal Generation

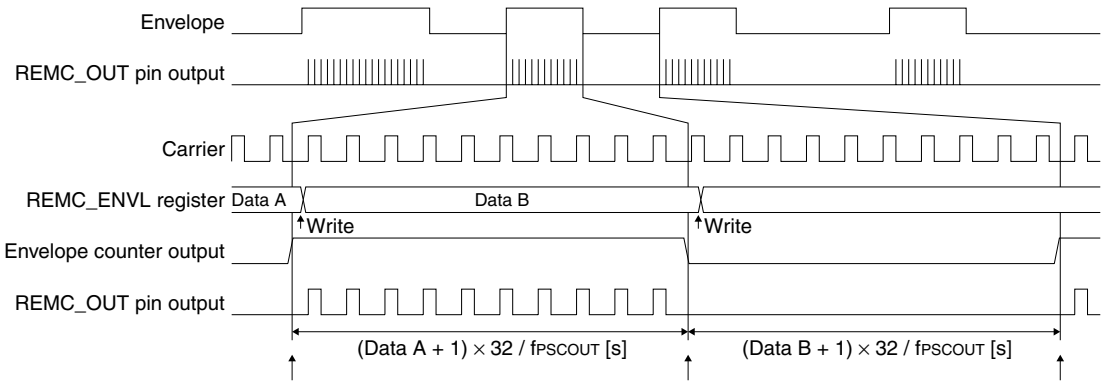
VI.6.4 Controlling Data Transmission/Reception

Before starting data transfer, set up the conditions by the procedure below.
(1) Configure the carrier signal. See Section VI.6.3.
(2) Set up the interrupt conditions. See Section VI.6.5.

Note: Make sure that the REMC module is idle (REMST/REMC_CTL register = 0) before setting the conditions above.

∗ **REMST**: REMC Start/Stop Control Bit in the REMC Control (REMC_CTL) Register (D0/0x5408)

Data transmit control



An underflow occurs in the envelope counter. The envelope counter outputs an underflow signal for interrupt and inverts the envelope output signal. At the same time, it loads the REMC_ENVL register value to the counter and continues counting from the loaded new data.

Figure VI.6.4.1 Data Transmission

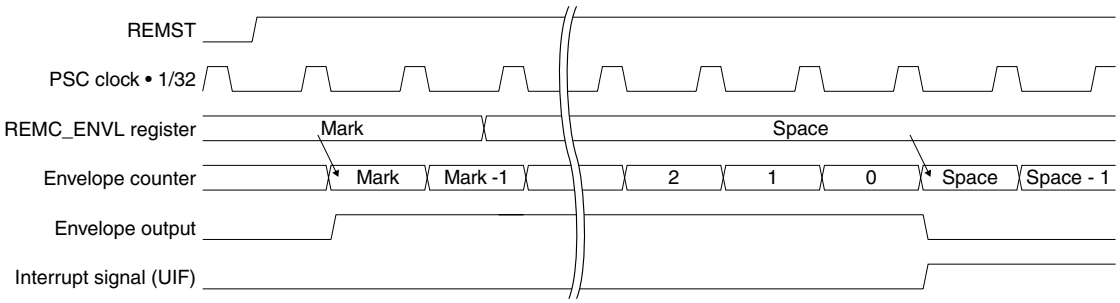


Figure VI.6.4.2 Underflow Interrupt Generation Timing

(1) Setting data transmit mode

Write 0 to MODE (D7/REMC_CFG register) to set the REMC in data transmit mode.

∗ **MODE**: REMC Mode Select Bit in the REMC Configuration (REMC_CFG) Register (D7/0x5404)

(2) Setting the REMC_ENVL register

Write the value equivalent to the first Mark width (high period) of the transmit data to REMC_ENVL register (0x540e).

Use the following equation to determine the value to be set to the envelope counter.

$$\text{REMC\_ENVL} = \frac{\text{Mark/Space width [second]} \times \text{Prescaler output clock frequency [Hz]}}{32} - 1$$

(3) Starting transmission

Set REMST (D0/REMC_CTL register) to 1 to start data transmit operation. The carrier generator starts outputting the carrier signal. The envelope counter turns the envelope output signal to high at the next count clock (PSC clock • 1/32) and loads the value set in the REMC_ENVL register into the counter. The envelope counter starts counting down from the loaded value.

(4) Setting the next envelope period

After starting transmission, set the Space width (low period) that follows the Mark period being currently output to REMC_ENVL register. Then, wait occurrence of an underflow interrupt.

(5) Underflow interrupt

When the envelope counter underflows, it inverts the envelope output signal and loads the REMC_ENVL register value to the counter. The counting operation continues.

At the same time, the REMC module sends an interrupt request to the interrupt controller (ITC) if the interrupt is enabled.

Use this interrupt to set the next period count data to the REMC_ENVL register.

(6) Terminating data transmission

After the last data transfer has finished (after an underflow interrupt occurs), write 0 to REMST to terminate data transmission. Note that the envelope counter stops immediately after REMST is set to 0.

Data receive control

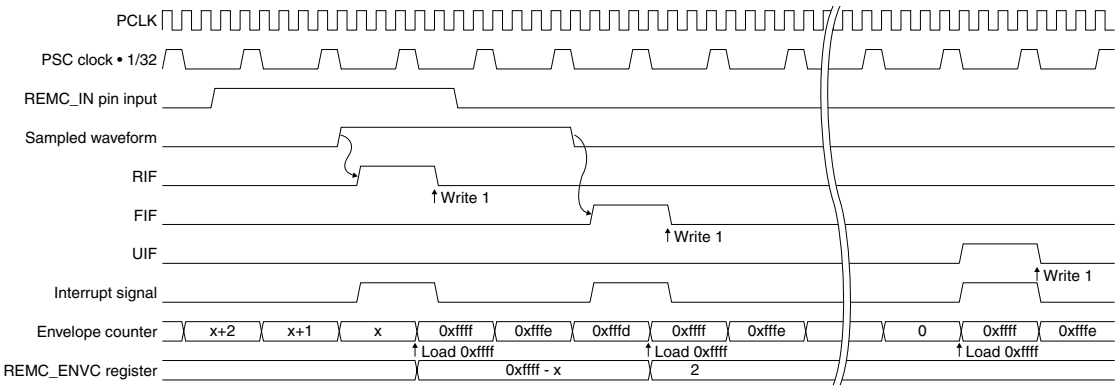


Figure VI.6.4.3 Data Reception

(1) Setting data receive mode

Write 1 to MODE (D7/REMC_CFG register) to set the REMC in data receive mode.

(2) Setting the REMC_ENVL register

Write 0xffff to REMC_ENVL register (0x540e). Note that the correct envelope pulse width cannot be obtained if a value other than 0xffff is set to the REMC_ENVL register.

(3) Start data reception

Set REMST (D0/REMC_CTL register) to 1 to start the data receive operation (input signal edge detection).

(4) Rising edge and falling edge detection and interrupt

The edge detector samples the signal input to the REMC_IN pin with the envelope count clock (PSC clock • 1/32) to detect input transition (rising edge or falling edge of the signal). When a signal edge is detected, a cause of rising edge or falling edge interrupt occurs and the REMC module sends an interrupt request to the ITC if the interrupt is enabled. The rising edge and falling edge interrupts can be enabled individually. A rising edge interrupt notifies the application program that a Space period ends and a Mark period starts. A falling edge interrupt notifies the application program that a Mark period ends and a Space period starts.

Note that a signal transition is regarded as noise by the glitch filter if the signal level after the input changes is not sampled for two or more sampling clock cycles. In this case no rising edge or falling edge interrupt occurs.

When an input signal edge is detected, the envelope counter data bits are inverted and loaded to the REMC_ENVC register (0x5410). This value represents the width of the envelope pulse that has been received. The envelope counter loads 0xffff and starts counting for the next envelope pulse.

If the envelope counter reaches 0 without an interrupt generated after the counter is set to 0xffff, either no receive data remains or a receive error has occurred. An underflow interrupt occurs even in data reception, use it for a terminate/error processing.

Note: After REMST is set to 1, the envelope counter loads 0xffff written in the REMC_ENVL register when the first input signal edge is detected. At this time, a rising edge interrupt occurs if the interrupt is enabled. This interrupt should be ignored.

(5) Obtain envelope pulse width

By using the interrupt above, read the counted value from the REMC_ENVC register. It represents the width of the envelope pulse received previously.

The pulse width can be determined by the following equation:

$$\text{Envelope pulse width} = \frac{(\text{REMC\_ENVC} + 1) \times 32}{f_{PSOUT}\ [\text{Hz}]}\ [\text{s}]$$

where REMC_ENVC is the REMC_ENVC register value and f_{PSOUT} is the prescaler output clock frequency.

The REMC_ENVC register value must be read out until the next rising or falling edge interrupt occurs. Otherwise, the REMC_ENVC register will be overwritten with the next count data.

(6) Terminating data reception

After the last data transfer has finished, write 0 to REMST to terminate data reception.

VI.6.5 REMC Interrupt

The REMC module can generate the following three types of interrupts:
• Underflow interrupt
• Rising edge interrupt
• Falling edge interrupt

The REMC module has one interrupt signal to be output to the interrupt controller (ITC) and it is shared with the three causes of interrupt. To determine the cause of interrupt that has occurred, read the interrupt flags in the REMC module.

Underflow interrupt

This interrupt request occurs when the envelope counter underflows during count-down, and it sets the interrupt flag UIF (D4/REMC_CFG register) in the REMC module to 1.
During data transmission, this interrupt notifies the application program that a envelope pulse with the period length specified has completed. During data reception, this interrupt notifies the application program that a data reception has completed or a receive error has occurred.

∗ **UIF**: Underflow Interrupt Flag Bit in the REMC Configuration (REMC_CFG) Register (D4/0x5404)

Set UIE (D0/REMC_CFG register) to 1 when using this interrupt. If UIE is set to 0 (default), UIF will not be set to 1 and an interrupt request by this cause will not be sent to the ITC.

∗ **UIE**: Underflow Interrupt Enable Bit in the REMC Configuration (REMC_CFG) Register (D0/0x5404)

If UIF is set to 1, the REMC module outputs the interrupt request signal to the ITC. The interrupt request signal sets the REMC interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The REMC interrupt handler routine should read the UIF flag to check if the interrupt has occurred due to a envelope counter underflow or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) UIF in the REMC module as well as the REMC interrupt flag in the ITC, to clear the cause of interrupt.

Rising edge interrupt

This interrupt request occurs when the signal input to the REMC_IN pin goes high from low status, and it sets the interrupt flag RIF (D6/REMC_CFG register) in the REMC module to 1.
The envelope counter value that has been counted from the time the previous falling edge detected is loaded to the REMC_ENVC register (0x5410). The application program can read the REMC_ENVC register to obtain the received Space pulse width after this interrupt has occurred.
At the same time this interrupt occurs, the envelope counter loads 0xffff and starts counting to measure the length between this interrupt and the next falling edge interrupt (Mark pulse width).

∗ **RIF**: Rising Edge Interrupt Flag Bit in the REMC Configuration (REMC_CFG) Register (D6/0x5404)

Set RIE (D2/REMC_CFG register) to 1 when using this interrupt. If RIE is set to 0 (default), RIF will not be set to 1 and an interrupt request by this cause will not be sent to the ITC.

∗ **RIE**: Rising Edge Interrupt Enable Bit in the REMC Configuration (REMC_CFG) Register (D2/0x5404)

If RIF is set to 1, the REMC module outputs the interrupt request signal to the ITC. The interrupt request signal sets the REMC interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The REMC interrupt handler routine should read the RIF flag to check if the interrupt has occurred due to detection of a rising edge of the input signal or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) RIF in the REMC module as well as the REMC interrupt flag in the ITC, to clear the cause of interrupt.

Falling edge interrupt

This interrupt request occurs when the signal input to the REMC_IN pin goes low from high status, and it sets the interrupt flag FIF (D5/REMC_CFG register) in the REMC module to 1.
The envelope counter value that has been counted from the time the previous rising edge detected is loaded to the REMC_ENVC register (0x5410). The application program can read the REMC_ENVC register to obtain the received Mark pulse width after this interrupt has occurred.
At the same time this interrupt occurs, the envelope counter loads 0xffff and starts counting to measure the length between this interrupt and the next rising edge interrupt (Space pulse width).

∗ **FIF**: Falling Edge Interrupt Flag Bit in the REMC Configuration (REMC_CFG) Register (D5/0x5404)

Set the FIE bit (D1/REMC_CFG register) to 1 when using this interrupt. If FIE is set to 0 (default), FIF will not be set to 1 and an interrupt request by this cause will not be sent to the ITC.

∗ **FIE**: Falling Edge Interrupt Enable Bit in the REMC Configuration (REMC_CFG) Register (D1/0x5404)

If FIF is set to 1, the REMC module outputs the interrupt request signal to the ITC. The interrupt request signal sets the REMC interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
The REMC interrupt handler routine should read the FIF flag to check if the interrupt has occurred due to detection of a falling edge of the input signal or another cause.
Furthermore, the interrupt handler routine must reset (write 1 to) FIF in the REMC module as well as the REMC interrupt flag in the ITC, to clear the cause of interrupt.

ITC registers for REMC interrupt

When a cause of interrupt that has been enabled occurs according to the interrupt condition settings shown above, the REMC module asserts the interrupt signal sent to the ITC. To generate a REMC interrupt, set the interrupt level and enable the interrupt using the ITC registers.
The following shows the control bits for the REMC interrupt in the ITC.

Interrupt flag in the ITC

∗ **AIFT14**: REMC Interrupt Flag Bit in the Additional Interrupt Flag (ITC_AIFLG) Register (D14/0x42e0)

Interrupt enable bit in the ITC

∗ **AIEN14**: REMC Interrupt Enable Bit in the Additional Interrupt Enable (ITC_AEN) Register (D14/0x42e2)

Interrupt level setup bits in the ITC

∗ **AILV14[2:0]**: REMC Interrupt Level Bits in the Additional Interrupt Level Setup (ITC_AILV7) Register 7 (D[2:0]/0x42f4)

The interrupt signal sent from the REMC module sets AIFT14 to 1. If AIEN14 has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the REMC interrupt, set AIEN14 to 0. AIFT14 is always set to 1 by the interrupt signal sent from the REMC module, regardless of how AIEN14 is set (even when set to 0).
AILV14[2:0] sets the interrupt level (0 to 7) of the REMC interrupt.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.
- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The REMC interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vector

The following shows the vector number and vector address for the REMC interrupt:

Vector number: 30 (0x1e)
Vector address: TTBR + 0x78

VI.6.6 Details of Control Registers

Table VI.6.6.1 List of REMC Registers

Address	Register name		Function
0x5400	REMC_PSC	REMC Prescaler Control Register	Sets up the REMC prescaler.
0x5404	REMC_CFG	REMC Configuration Register	Sets the REMC modes and controls the REMC interrupt.
0x5408	REMC_CTL	REMC Control Register	Starts/stops transmission.
0x540c	REMC_CARL	REMC Carrier Load Register	Configures the carrier signal.
0x540e	REMC_ENVL	REMC Envelope Load Register	Configures the envelope pulse width.
0x5410	REMC_ENVC	REMC Envelope Capture Register	Input envelope pulse width

The following describes each REMC register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x5400: REMC Prescaler Control Register (REMC_PSC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Prescaler Control Register (REMC_PSC)	0x5400 (16 bits)	D15–3	–	reserved	–	–	–	0 when being read.
		D2	REMPSON	REMC prescaler control	1 On / 0 Off	0	R/W	
		D1–0	REMPSDIV [1:0]	REMC prescaler division ratio select (Prescaler output clock)	REMPSDIV[1:0] / Clock: 0x3 PCLK•1/32; 0x2 PCLK•1/24; 0x1 PCLK•1/16; 0x0 PCLK•1/12	0x0	R/W	

D[15:3] Reserved

D2 REMPSON: REMC Prescaler Control Bit

Turns the prescaler in the REMC module on and off.

1 (R/W): On

0 (R/W): Off (default)

Set REMPSON to 1 to turn the prescaler on before the REMC module can be used.

D[1:0] REMPSDIV[1:0]: REMC Prescaler Division Ratio Select Bits

These bits select the count clock for the carrier generator.

Table VI.6.6.2 Selecting a Clock for Carrier Generator

REMPSDIV[1:0]	Output clock
0x3	PCLK•1/32
0x2	PCLK•1/24
0x1	PCLK•1/16
0x0	PCLK•1/12

(Default: 0x0)

Most infrared remote carrier frequencies fall within the range from 30 kHz to 56 kHz and their duty ratio is 1/2, 1/3, or 1/4. To insure the accuracy of the carrier frequency, the prescaler output clock frequency should be set to 1 MHz (min.) to 8 MHz (max.).

Examples:

1. When the PCLK frequency is 48 MHz, PCLK•1/24 (2 MHz) is recommended.
2. When the PCLK frequency is 60 MHz, PCLK•1/32 (1.875 MHz) is recommended.

Also the prescaler clock is supplied to the envelope counter and the edge detector after it is divided by 32.

0x5404: REMC Configuration Register (REMC_CFG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
REMC Configuration Register (REMC_CFG)	0x5404 (16 bits)	D15–8	–	reserved	–				–	–	0 when being read.
		D7	MODE	REMC mode select	1	Receive	0	Transmit	0	R/W	
		D6	RIF	Rising edge interrupt flag	1	Cause of interrupt occurred	0	Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D5	FIF	Falling edge interrupt flag					0	R/W	
		D4	UIF	Underflow interrupt flag					0	R/W	
		D3	–	reserved	–				–	–	0 when being read.
		D2	RIE	Rising edge interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	FIE	Falling edge interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	UIE	Underflow interrupt enable	1	Enable	0	Disable	0	R/W	

D[15:8] Reserved

D7 MODE: REMC Mode Select Bit

Selects the data transmit/receive direction.

1 (R/W): Receive mode

0 (R/W): Transmit mode (default)

Note: Make sure REMST (D0/REMC_CTL register) is 0 (REMC is idle) before changing the mode.

D6 RIF: Rising Edge Interrupt Flag Bit

This is the interrupt flag to indicate the rising edge interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred

0 (R): No cause of interrupt has occurred (default)

1 (W): Flag is reset

0 (W): Has no effect

RIF is set to 1 when a rising edge is detected in the input signal. This flag is effective only when the REMC is in receive mode and RIE (D2) is set to 1. When RIE is 0, RIF will not be set to 1 even if a rising edge is detected.

When this flag is set to 1, the REMC interrupt request signal is output to the ITC. The interrupt request signal sets the REMC interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.

RIF is reset by writing 1.

D5 FIF: Falling Edge Interrupt Flag Bit

This is the interrupt flag to indicate the falling edge interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred

0 (R): No cause of interrupt has occurred (default)

1 (W): Flag is reset

0 (W): Has no effect

FIF is set to 1 when a falling edge is detected in the input signal. This flag is effective only when the REMC is in receive mode and FIE (D1) is set to 1. When FIE is 0, FIF will not be set to 1 even if a falling edge is detected.

When this flag is set to 1, the REMC interrupt request signal is output to the ITC. The interrupt request signal sets the REMC interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.

FIF is reset by writing 1.

D4 UIF: Underflow Interrupt Flag Bit

This is the interrupt flag to indicate the underflow interrupt cause occurrence status.

1 (R): Cause of interrupt has occurred

0 (R): No cause of interrupt has occurred (default)

1 (W): Flag is reset

0 (W): Has no effect

UIF is set to 1 when the envelope counter underflows during data transmission or reception. This flag is effective only when the UIE (D0) is set to 1. When UIE is 0, UIF will not be set to 1 even if an envelope counter underflow occurs.

When this flag is set to 1, the REMC interrupt request signal is output to the ITC. The interrupt request signal sets the REMC interrupt flag in the ITC to 1 and an interrupt occurs if other interrupt conditions meet the ITC and S1C17 Core settings.
UIF is reset by writing 1.

D3 Reserved

D2 RIE: Rising Edge Interrupt Enable Bit

Enables or disables the interrupt by detecting a rising edge of the input signal.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting RIE to 1 enables the rising edge interrupt; setting to 0 disables the interrupt.
In addition, it is necessary to set the REMC interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

D1 FIE: Falling Edge Interrupt Enable Bit

Enables or disables the interrupt by detecting a falling edge of the input signal.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting FIE to 1 enables the falling edge interrupt; setting to 0 disables the interrupt.
In addition, it is necessary to set the REMC interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

D0 UIE: Underflow Interrupt Enable Bit

Enables or disables the interrupt by a envelope counter underflow.
1 (R/W): Enable interrupt
0 (R/W): Disable interrupt (default)

Setting UIE to 1 enables the underflow interrupt; setting to 0 disables the interrupt.
In addition, it is necessary to set the REMC interrupt enable bits in the ITC to interrupt enabled to actually generate an interrupt.

Note: To avoid occurrence of unnecessary interrupts, be sure to reset the interrupt flag before the REMC interrupt is enabled using the interrupt enable bit.

0x5408: REMC Control Register (REMC_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Control Register (REMC_CTL)	0x5408 (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	REMST	REMC start/stop control	1 Start 0 Stop	0	R/W	

D[15:1] Reserved

D0 REMST: REMC Start/Stop Control Bit

Starts/stops the REMC module.

1 (R): REMC is transmitting/receiving data
0 (R): REMC is idle (default)
1 (W): Start REMC operation
0 (W): Stop REMC operation

When REMST is set to 1, the REMC module starts data transmission or data reception according to the MODE (D7/REMC_CFG register) setting. When REMST is set to 0, the REMC module stops operating.

0x540c: REMC Carrier Load Register (REMC_CARL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Carrier Load Register (REMC_CARL)	0x540c (16 bits)	D15–8	CLDH[7:0]	REMC carrier high width setup	0x0 to 0xff	0x0	R/W	
		D7–0	CLDL[7:0]	REMC carrier low width setup	0x0 to 0xff	0x0	R/W	

Note: This register is effective only in transmit mode and is not used in receive mode.

D[15:8] CLDH[7:0]: REMC Carrier High Width Setup Bits

Sets the high period width of the carrier signal. (Default: 0x0)
The carrier high width should be set as the number of the prescaler clock cycles +1.
Carrier high width [s] = (CLDH[7:0] + 1) / fPSOUT [Hz] (fPSOUT: Prescaler clock frequency)

D[7:0] CLDL[7:0]: REMC Carrier Low Width Setup Bits

Sets the low period width of the carrier signal. (Default: 0x0)
The carrier low widths should be set as the number of prescaler clock cycles +1.
Carrier low width [s] = (CLDL[7:0] + 1) / fPSOUT [Hz] (fPSOUT: Prescaler clock frequency)

The table below shows examples of carrier settings when the prescaler clock frequency is 2 MHz.

Table VI.6.6.3 Carrier Setting Examples

Prescaler clock frequency	2 MHz		
Carrier frequency	30 kHz	38 kHz	56 kHz
Carrier duty	1/2	1/3	1/4
Ideal period	33.333 µs	26.316 µs	17.857 µs
CLDH[7:0] setting	32	17	8
CLDL[7:0] setting	32	34	26
Period error margin	1%	0.7%	0.8%

The carrier signal is generated as shown in Figure VI.6.6.1.

Example: REMPSDIV[1:0] = 0x2 (PCLK•1/24, 2 MHz), CLDH[7:0] = 17, CLDL[7:0] = 34
(carrier 38 kHz, 1/3 duty)

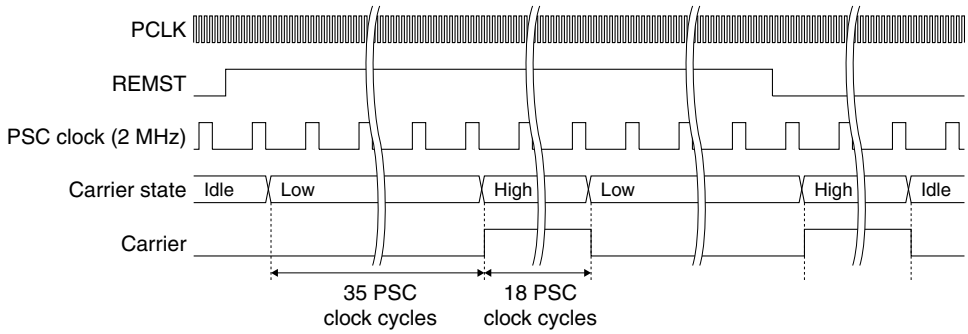


Figure VI.6.6.1 Carrier Signal Generation

0x540e: REMC Envelope Load Register (REMC_ENVL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Envelope Load Register (REMC_ENVL)	**0x540e** (16 bits)	D15–0	**ELD[15:0]**	Envelope counter preset data	0x0 to 0xffff	0x0	R/W	

D[15:0] ELD[15:0]: Envelope Counter Preset Data Bits

Sets the initial value to be set to the envelope counter. (Default: 0x0)
The counter stops when it reaches 0 and generates a cause of underflow interrupt.

In data transmit mode

During data transmission, set the envelope pulse width to be transmitted.
Use the following equation to determine the value to be set to this register.

$$\text{REMC\_ENVL} = \frac{\text{Mark/Space width [second]} \times \text{Prescaler output clock frequency [Hz]}}{32} - 1$$

When data transmission is started by writing 1 to REMST (D0/REMC_CTL register), the envelope counter starts counting down from the value set in this register and generates a cause of underflow interrupt when the counter underflows. The next envelope pulse width can be set using this interrupt.
The envelope counter loads the value set in this register when it underflows and continues counting down. It stops when REMST is set to 0.

In data receive mode

During data reception, set 0xffff to this register. It is used as the initial value for counting input envelope pulse width using the envelope counter. When a rising or falling edge of the input signal is detected, an interrupt occurs and the envelope counter loads 0xffff set in this register and starts counting down from the set value. Before loading 0xffff, the count value in the envelope counter is inverted and sent to the REMC_ENVC register (0x5410). This value represents the width of the envelope pulse previously received.

0x5410: REMC Envelope Capture Register (REMC_ENVC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Envelope Capture Register (REMC_ENVC)	**0x5410** (16 bits)	D15–0	**ECP[15:0]**	Receive envelope pulse width	0x0 to 0xffff	0x0	R	

D[15:0] ECP[15:0]: Receive Envelope Pulse Width Bits

The envelope pulse width measured by the envelope counter during reception is loaded in this register. (Default: 0x0)

When an input signal edge is detected, the envelope counter data bits are inverted and loaded to this register. This value represents the width of the envelope pulse that has been received.

After a rising or falling edge interrupt has occurred, read this register. The pulse width can be determined by the following equation:

$$\text{Envelope pulse width} = \frac{(\text{REMC\_ENVC} + 1) \times 32}{f_{PSOUT}\ [\text{Hz}]}\ [\text{s}]$$

where REMC_ENVC is the REMC_ENVC register value and f_{PSOUT} is the prescaler output clock frequency.

The REMC_ENVC register value must be read out until the next rising or falling edge interrupt occurs. Otherwise, the REMC_ENVC register will be overwritten with the next count data.

VI.6.7 Precautions

- Before the REMC module can start transmission/reception, the prescaler must be run.
- Make sure that the REMC module is idle (REMST/REMC_CTL register = 0) before setting the prescaler and carrier generator conditions.

  * **REMST**: REMC Start/Stop Control Bit in the REMC Control (REMC_CTL) Register (D0/0x5408)

VI.7 Card Interface (CARD)

VI.7.1 Outline of the Card Interface

The card interface (CARD) module generates the #SMRD and #SMWR signals for NAND Flash and SmartMedia cards.
The device can be located only in the #CE2 area (0xd00000–0xeffffff, 2MB). The data and address signals of the device can be connected directly to the external bus of the SRAMC.
The CARD module supports 8- and 16-bit NAND Flash devices and SmartMedia cards.
Use general-purpose input/output ports to control the signals specific to NAND Flash or SmartMedia card.

VI.7.2 Card Interface Pins

Table VI.7.2.1 lists the card interface output pins.

Table VI.7.2.1 List of Card Interface Pins

Pin name	I/O	Size	Function
#SMRD	O	1	SmartMedia read signal output pin This pin outputs the read signal for NAND Flash and SmartMedia card.
#SMWR	O	1	SmartMedia write signal output pin This pin outputs the write signal for NAND Flash and SmartMedia card.

The card interface output pins (#SMRD, #SMWR) are shared with the I/O ports and they are initialized as general-purpose I/O port pins by default. Before using these pins for the card interface, the pin functions must be switched using the Port Function Select Registers.
For details on switching pin function, Section I.3.3, “Switching Over the Multiplexed Pin Functions.”

VI.7.3 Card Interface Control Signals

Figure VI.7.3.1 shows the logic used to generate SmartMedia interface signals. Figure VI.7.3.2 shows an example of connecting the S1C17501 and a SmartMedia card (NAND Flash).

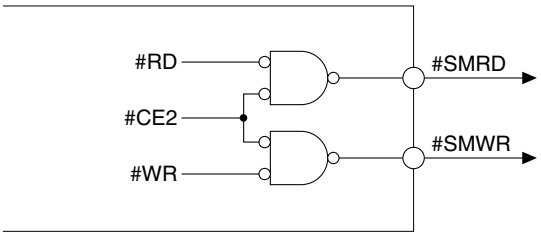


Figure VI.7.3.1 SmartMedia Interface Signal Generation Circuit

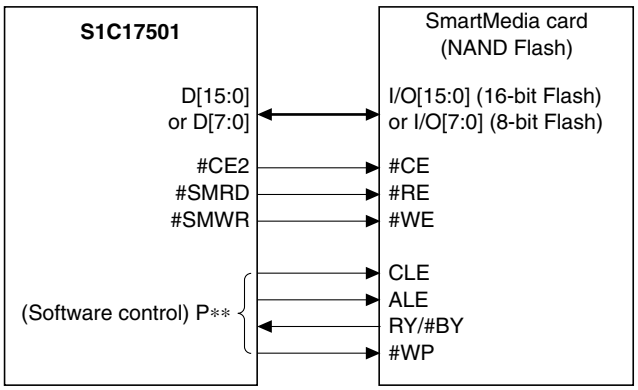


Figure VI.7.3.2 Example of Connecting a SmartMedia Card

VII S1C17501 I/O PORTS

VII.1 General-Purpose I/O Ports (GPIO)

VII.1.1 Structure of I/O Port

The S1C17501 contains 8 input ports (P0[7:0]) and 91 I/O ports (P1[6:0], P2[7:0], P3[7:0], P4[5:0], P5[7:0], P6[7:0], P7[7:0], P8[6:0], P9[7:0], PA[6:0], PB[7:0], and PC[7:0]) that can be directed for input or output through the use of a program. Although each pin is used for input/output from/to the internal peripheral circuits, some pins can be used as general-purpose input/output ports unless they are used for the peripheral circuits.
Also the I/O ports support 8 systems of port input interrupts.
Figure VII.1.1.1 shows the structure of a typical I/O port.

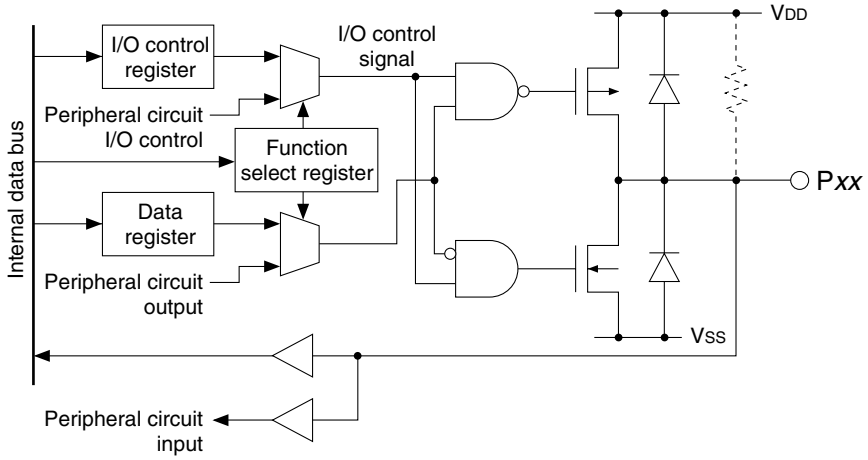


Figure VII.1.1.1 Structure of I/O Port

Notes: • The P36, P45, and PA[6:0] ports provide a 100 kΩ internal pull-up resistor. Other ports have no pull-up/down resistor.

• The P0[7:0] are input only ports and the interface voltage level is AV_{DD}, not V_{DD}.

VII.1.2 Selecting the I/O Pin Functions

The I/O ports concurrently serve as the input/output pins for peripheral circuits or bus signals. Whether they are used as I/O ports or for peripheral circuits/bus signals can be selected bit-for-bit using the Port Function Select Registers. All pins not used for peripheral circuits/bus signals can be used as general-purpose I/O ports.

Each I/O port pin (P*xx*) is initialized for a default function at initial reset.
For details of pin functions and how to switch over, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

The subsequent sections explain the port functions assuming that the pin has been set as a general-purpose I/O port.

Note: The P3[7:5] pins are configured as the debug interface pins by default. Other P*xx* pins are all configured as general-purpose I/O ports.

VII.1.3 I/O Control Register and I/O Modes

The I/O ports are directed for input or output modes by writing data to IOC*x* corresponding to each port bit.

- **IOC1[6:0]**: P1[6:0] I/O Control Bits in the P1 I/O Control (P1_IOC) Register (D[6:0]/0x4403)
- **IOC2[7:0]**: P2[7:0] I/O Control Bits in the P2 I/O Control (P2_IOC) Register (D[7:0]/0x4405)
- **IOC3[7:0]**: P3[7:0] I/O Control Bits in the P3 I/O Control (P3_IOC) Register (D[7:0]/0x4407)
- **IOC4[5:0]**: P4[5:0] I/O Control Bits in the P4 I/O Control (P4_IOC) Register (D[5:0]/0x4409)
- **IOC5[7:0]**: P5[7:0] I/O Control Bits in the P5 I/O Control (P5_IOC) Register (D[7:0]/0x440b)
- **IOC6[7:0]**: P6[7:0] I/O Control Bits in the P6 I/O Control (P6_IOC) Register (D[7:0]/0x440d)
- **IOC7[7:0]**: P7[7:0] I/O Control Bits in the P7 I/O Control (P7_IOC) Register (D[7:0]/0x440f)
- **IOC8[6:0]**: P8[6:0] I/O Control Bits in the P8 I/O Control (P8_IOC) Register (D[6:0]/0x4411)
- **IOC9[7:0]**: P9[7:0] I/O Control Bits in the P9 I/O Control (P9_IOC) Register (D[7:0]/0x4413)
- **IOCA[6:0]**: PA[6:0] I/O Control Bits in the PA I/O Control (PA_IOC) Register (D[6:0]/0x4415)
- **IOCB[7:0]**: PB[7:0] I/O Control Bits in the PB I/O Control (PB_IOC) Register (D[7:0]/0x4417)
- **IOCC[7:0]**: PC[7:0] I/O Control Bits in the PC I/O Control (PC_IOC) Register (D[7:0]/0x4419)

To set an I/O port for input, write 0 to the I/O control bit. I/O ports set for input mode are placed in the high-impedance state, and thus function as input ports. In the input mode, the state of the input pin is read directly, so the data is 1 when the pin state is high (V_{DD} level) or 0 when the pin state is low (V_{SS} level).

To set an I/O port for output, write 1 to the I/O control bit. I/O port set for output function as output ports. When the port output data is 1, the port outputs a high level (V_{DD} level); when the data is 0, the port outputs a low level (V_{SS} level).

VII.1.4 I/O Data Register

The registers listed below are used to read data from the I/O-port pins or to set output data.

- **P0D[7:0]**: P0[7:0] Port Input Data Bits in the P0 Port Input Data (P0_DAT) Register (D[7:0]/0x4400)
- **P1D[6:0]**: P1[6:0] Port Input/Output Data Bits in the P1 Port Input/Output Data (P1_DAT) Register (D[6:0]/0x4402)
- **P2D[7:0]**: P2[7:0] Port Input/Output Data Bits in the P2 Port Input/Output Data (P2_DAT) Register (D[7:0]/0x4404)
- **P3D[7:0]**: P3[7:0] Port Input/Output Data Bits in the P3 Port Input/Output Data (P3_DAT) Register (D[7:0]/0x4406)
- **P4D[5:0]**: P4[5:0] Port Input/Output Data Bits in the P4 Port Input/Output Data (P4_DAT) Register (D[5:0]/0x4408)
- **P5D[7:0]**: P5[7:0] Port Input/Output Data Bits in the P5 Port Input/Output Data (P5_DAT) Register (D[7:0]/0x440a)
- **P6D[7:0]**: P6[7:0] Port Input/Output Data Bits in the P6 Port Input/Output Data (P6_DAT) Register (D[7:0]/0x440c)
- **P7D[7:0]**: P7[7:0] Port Input/Output Data Bits in the P7 Port Input/Output Data (P7_DAT) Register (D[7:0]/0x440e)
- **P8D[6:0]**: P8[6:0] Port Input/Output Data Bits in the P8 Port Input/Output Data (P8_DAT) Register (D[6:0]/0x4410)
- **P9D[7:0]**: P9[7:0] Port Input/Output Data Bits in the P9 Port Input/Output Data (P9_DAT) Register (D[7:0]/0x4412)
- **PAD[6:0]**: PA[6:0] Port Input/Output Data Bits in the PA Port Input/Output Data (PA_DAT) Register (D[6:0]/0x4414)
- **PBD[7:0]**: PB[7:0] Port Input/Output Data Bits in the PB Port Input/Output Data (PB_DAT) Register (D[7:0]/0x4416)
- **PCD[7:0]**: PC[7:0] Port Input/Output Data Bits in the PC Port Input/Output Data (PC_DAT) Register (D[7:0]/0x4418)

When an I/O port is set for output, the data written to the register is directly output to the I/O port pin. If the data written to the register is 1, the port pin is set high (V_{DD} level); if the data is 0, the port pin is set low (V_{SS} level). Even in the input mode, data can be written to the data register without affecting the pin state.

When the register is read, the voltage level on the port pin is read out regardless of whether an I/O port is set for input or output mode. If the pin voltage is high (V_{DD} level), 1 is read out as input data; if the pin voltage is low (V_{SS} level), 0 is read out as input data.

VII.1.5 Port Input Interrupt

The GPIO module has eight interrupt systems (port input interrupts 0 to 7) and a port can be selected for generating each cause of interrupt.
The interrupt condition can also be selected from between input signal edge (rising edge or falling edge) and input signal level (high level or low level) in the interrupt controller (ITC).
Figure VII.1.5.1 shows the configuration of the port input interrupt circuit.

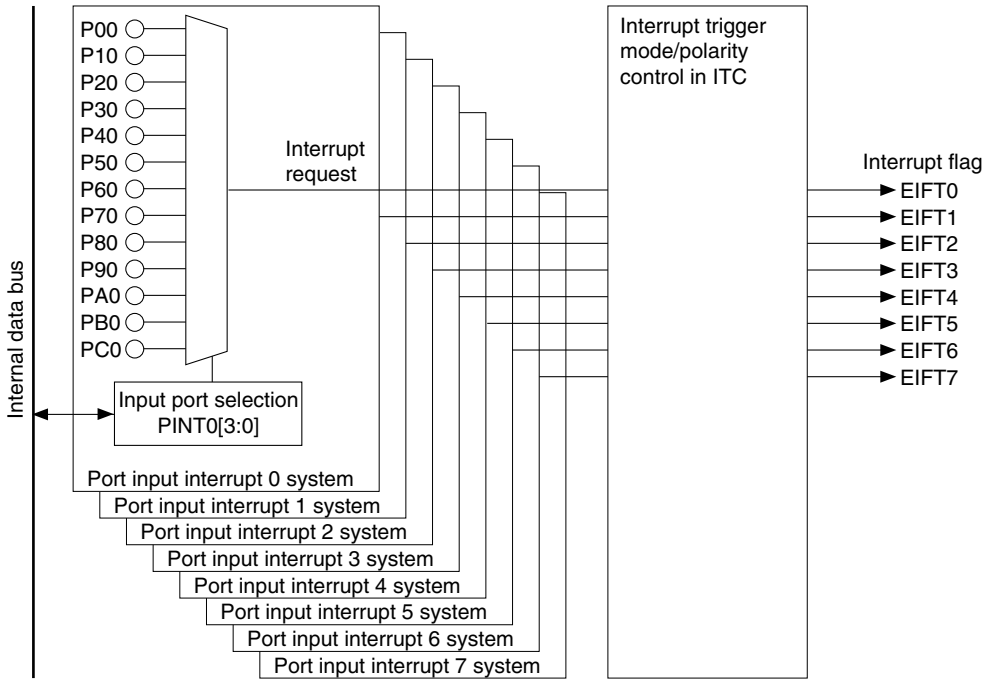


Figure VII.1.5.1 Configuration of Port Input Interrupt Circuit

VII.1.5.1 Selecting Input Pins

Each port input interrupt system allows selection of an I/O port pin from the eight predefined pins.
The following lists the control bits and registers to select a port for each interrupt system.

- **PINT0[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 0 Select (PINTSEL0) Register (D[7:4]/0x4440)
- **PINT1[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 1 Select (PINTSEL1) Register (D[7:4]/0x4441)
- **PINT2[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 2 Select (PINTSEL2) Register (D[7:4]/0x4442)
- **PINT3[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 3 Select (PINTSEL3) Register (D[7:4]/0x4443)
- **PINT4[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 4 Select (PINTSEL4) Register (D[7:4]/0x4444)
- **PINT5[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 5 Select (PINTSEL5) Register (D[7:4]/0x4445)
- **PINT6[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 6 Select (PINTSEL6) Register (D[7:4]/0x4446)
- **PINT7[3:0]**: Interrupt Port Select Bits in the Port Input Interrupt 7 Select (PINTSEL7) Register (D[7:4]/0x4447)

Table VII.1.5.1.1 shows the selectable pins for each interrupt system.

Table VII.1.5.1.1 Selecting Pins for Port Input Interrupts

Port input interrupt system	PINT*x*[3:0] settings												
	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc
0	P00	P10	P20	P30	P40	P50	P60	P70	P80	P90	PA0	PB0	PC0
1	P01	P11	P21	P31	P41	P51	P61	P71	P81	P91	PA1	PB1	PC1
2	P02	P12	P22	P32	P42	P52	P62	P72	P82	P92	PA2	PB2	PC2
3	P03	P13	P23	P33	P43	P53	P63	P73	P83	P93	PA3	PB3	PC3
4	P04	P14	P24	P34	P44	P54	P64	P74	P84	P94	PA4	PB4	PC4
5	P05	P15	P25	P35	P45	P55	P65	P75	P85	P95	PA5	PB5	PC5
6	P06	P16	P26	P36	–	P56	P66	P76	P86	P96	PA6	PB6	PC6
7	P07	–	P27	P37	–	P57	P67	P77	–	P97	–	PB7	PC7

VII.1.5.2 Control Registers of the Interrupt Controller

Selecting the trigger mode and polarity

The interrupt controller (ITC) provides two trigger modes for the port interrupts, the pulse trigger mode and the level trigger mode, to accept either a pulse signal or a level signal as interrupt requests.
The trigger mode can be selected using the EITG*x* bit in the ITC_ELV*x* registers (0x4306 to 0x430c). When EITG*x* is set to 1, level trigger mode is selected; when EITG*x* is set to 0 (default), pulse trigger mode is selected.

The ITC allows these interrupt sources to select the polarity of the interrupt request signal to be sent to the ITC. The signal polarity can be selected using the EITP*x* bit in the ITC_ELV*x* registers (0x4306 to 0x430c). When EITP*x* is set to 1, positive pulse/rising edge (in pulse trigger mode) or active high (in level mode) is selected; when EIPT*x* is set to 0 (default), negative pulse/falling edge (in pulse trigger mode) or active low (in level mode) is selected.

Table VII.1.5.2.1 Trigger Mode/Polarity Select Bits

Interrupt system	Trigger mode select bit	Trigger polarity select bit	Register address
Port input interrupt 0	EITG0 (D4/ITC_ELV0 register)	EITP0 (D5/ITC_ELV0 register)	0x4306
Port input interrupt 1	EITG1 (D12/ITC_ELV0 register)	EITP1 (D13/ITC_ELV0 register)	0x4306
Port input interrupt 2	EITG2 (D4/ITC_ELV1 register)	EITP2 (D5/ITC_ELV1 register)	0x4308
Port input interrupt 3	EITG3 (D12/ITC_ELV1 register)	EITP3 (D13/ITC_ELV1 register)	0x4308
Port input interrupt 4	EITG4 (D4/ITC_ELV2 register)	EITP4 (D5/ITC_ELV2 register)	0x430a
Port input interrupt 5	EITG5 (D12/ITC_ELV2 register)	EITP5 (D13/ITC_ELV2 register)	0x430a
Port input interrupt 6	EITG6 (D4/ITC_ELV3 register)	EITP6 (D5/ITC_ELV3 register)	0x430c
Port input interrupt 7	EITG7 (D12/ITC_ELV3 register)	EITP7 (D13/ITC_ELV3 register)	0x430c

With these registers, the port input interrupt condition is determined as shown in Table VII.1.5.2.2.

Table VII.1.5.2.2 Port Input Interrupt Condition

EITG*x*	EITP*x*	Port input interrupt condition
1	1	High level
1	0	Low level
0	1	Rising edge
0	0	Falling edge

ITC registers to control interrupt generation

Table VII.1.5.2.3 shows the interrupt control registers of the ITC that are provided for each port input interrupt system.

Table VII.1.5.2.3 Control Registers of Interrupt Controller

Interrupt system	Interrupt flag	Interrupt enable bit	Interrupt level setup bits
Port input interrupt 0	EIFT0 (D0/ITC_IFLG)	EIEN0 (D0/ITC_EN)	EILV0[2:0] (D[2:0]/ITC_ELV0)
Port input interrupt 1	EIFT1 (D1/ITC_IFLG)	EIEN1 (D1/ITC_EN)	EILV1[2:0] (D[10:8]/ITC_ELV0)
Port input interrupt 2	EIFT2 (D2/ITC_IFLG)	EIEN2 (D2/ITC_EN)	EILV2[2:0] (D[2:0]/ITC_ELV1)
Port input interrupt 3	EIFT3 (D3/ITC_IFLG)	EIEN3 (D3/ITC_EN)	EILV3[2:0] (D[10:8]/ITC_ELV1)
Port input interrupt 4	EIFT4 (D4/ITC_IFLG)	EIEN4 (D4/ITC_EN)	EILV4[2:0] (D[2:0]/ITC_ELV2)
Port input interrupt 5	EIFT5 (D5/ITC_IFLG)	EIEN5 (D5/ITC_EN)	EILV5[2:0] (D[10:8]/ITC_ELV2)
Port input interrupt 6	EIFT6 (D6/ITC_IFLG)	EIEN6 (D6/ITC_EN)	EILV6[2:0] (D[2:0]/ITC_ELV3)
Port input interrupt 7	EIFT7 (D7/ITC_IFLG)	EIEN7 (D7/ITC_EN)	EILV7[2:0] (D[10:8]/ITC_ELV3)

ITC_IFLG register (0x4300)
ITC_EN register (0x4302)
ITC_ELV0, ITC_ELV1, ITC_ELV2, ITC_ELV3 registers (0x4306, 0x4308, 0x430a, 0x430c)

When the interrupt generation condition described above is met, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the port input interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the designated port input, regardless of how the interrupt enable bit is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the port input interrupt. If the same interrupt level is set, port input interrupt 0 has highest priority and port input interrupt 7 has lowest priority.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The port input interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, "Interrupt Controller (ITC)."

Interrupt vectors

The following shows the vector numbers and vector addresses for the port input interrupts:

Table VII.1.5.2.4 Port Input Interrupt Vectors

Interrupt system	Vector number	Vector address
Port input interrupt 0	4 (0x4)	TTBR + 0x10
Port input interrupt 1	5 (0x5)	TTBR + 0x14
Port input interrupt 2	6 (0x6)	TTBR + 0x18
Port input interrupt 3	7 (0x7)	TTBR + 0x1c
Port input interrupt 4	12 (0xc) or 16 (0x10)	TTBR + 0x30 or TTBR + 0x40
Port input interrupt 5	13 (0xd) or 17 (0x11)	TTBR + 0x34 or TTBR + 0x44
Port input interrupt 6	14 (0xe) or 18 (0x12)	TTBR + 0x38 or TTBR + 0x48
Port input interrupt 7	15 (0xf) or 19 (0x13)	TTBR + 0x3c or TTBR + 0x4c

Note: The interrupt vector numbers 12, 14–16, and 18–19 for port input interrupts 4, 6, and 7 are shared with another cause of interrupt. To use the vector number for the port input interrupt, set the interrupt enable bit for the port input interrupt to 1 and clear one for another to 0.

Table VII.1.5.2.5 Interrupt Vectors 12 and 16 (UART CH.0, CLG_T16U0, and Port 4 Interrupts)

Interrupt enable bit			Interrupt vector 12	Interrupt vector 16
IIEN4 (UART CH.0)	IIEN0 (CLG_T16U0)	EIEN4 (Port 4)		
1	1	1	CLG_T16U0 interrupt	UART CH.0 interrupt
1	1	0	CLG_T16U0 interrupt	UART CH.0 interrupt
1	0	1	Port interrupt 4	UART CH.0 interrupt
1	0	0	–	UART CH.0 interrupt
0	1	1	CLG_T16U0 interrupt	Port interrupt 4
0	1	0	CLG_T16U0 interrupt	–
0	0	1	Port interrupt 4	Port interrupt 4
0	0	0	–	–

Table VII.1.5.2.6 Interrupt Vectors 14 and 18 (SPI CH.0, CLG_T8S, and Port 6 Interrupts)

Interrupt enable bit			Interrupt vector 14	Interrupt vector 18
IIEN6 (SPI CH.0)	IIEN2 (CLG_T8S)	EIEN6 (Port 6)		
1	1	1	CLG_T8S interrupt	SPI CH.0 interrupt
1	1	0	CLG_T8S interrupt	SPI CH.0 interrupt
1	0	1	Port interrupt 6	SPI CH.0 interrupt
1	0	0	–	SPI CH.0 interrupt
0	1	1	CLG_T8S interrupt	Port interrupt 6
0	1	0	CLG_T8S interrupt	–
0	0	1	Port interrupt 6	Port interrupt 6
0	0	0	–	–

Table VII.1.5.2.7 Interrupt Vectors 15 and 19 (I^2C, CLG_T8I, and Port 7 Interrupts)

Interrupt enable bit			Interrupt vector 15	Interrupt vector 19
IIEN7 (I^2C)	IIEN3 (CLG_T8I)	EIEN7 (Port 7)		
1	1	1	CLG_T8I interrupt	I^2C interrupt
1	1	0	CLG_T8I interrupt	I^2C interrupt
1	0	1	Port interrupt 7	I^2C interrupt
1	0	0	–	I^2C interrupt
0	1	1	CLG_T8I interrupt	Port interrupt 7
0	1	0	CLG_T8I interrupt	–
0	0	1	Port interrupt 7	Port interrupt 7
0	0	0	–	–

VII.1.6 Details of Control Registers

Table VII.1.6.1 List of I/O Port Registers

Address	Register name		Function
0x4400	P0_DAT	P0 Port Input Data Register	P0 port input data
0x4402	P1_DAT	P1 Port Input/Output Data Register	P1 port input/output data
0x4403	P1_IOC	P1 Port I/O Control Register	Selects the P1 port I/O direction.
0x4404	P2_DAT	P2 Port Input/Output Data Register	P2 port input/output data
0x4405	P2_IOC	P2 Port I/O Control Register	Selects the P2 port I/O direction.
0x4406	P3_DAT	P3 Port Input/Output Data Register	P3 port input/output data
0x4407	P3_IOC	P3 Port I/O Control Register	Selects the P3 port I/O direction.
0x4408	P4_DAT	P4 Port Input/Output Data Register	P4 port input/output data
0x4409	P4_IOC	P4 Port I/O Control Register	Selects the P4 port I/O direction.
0x440a	P5_DAT	P5 Port Input/Output Data Register	P5 port input/output data
0x440b	P5_IOC	P5 Port I/O Control Register	Selects the P5 port I/O direction.
0x440c	P6_DAT	P6 Port Input/Output Data Register	P6 port input/output data
0x440d	P6_IOC	P6 Port I/O Control Register	Selects the P6 port I/O direction.
0x440e	P7_DAT	P7 Port Input/Output Data Register	P7 port input/output data
0x440f	P7_IOC	P7 Port I/O Control Register	Selects the P7 port I/O direction.
0x4410	P8_DAT	P8 Port Input/Output Data Register	P8 port input/output data
0x4411	P8_IOC	P8 Port I/O Control Register	Selects the P8 port I/O direction.
0x4412	P9_DAT	P9 Port Input/Output Data Register	P9 port input/output data
0x4413	P9_IOC	P9 Port I/O Control Register	Selects the P9 port I/O direction.
0x4414	PA_DAT	PA Port Input/Output Data Register	PA port input/output data
0x4415	PA_IOC	PA Port I/O Control Register	Selects the PA port I/O direction.
0x4416	PB_DAT	PB Port Input/Output Data Register	PB port input/output data
0x4417	PB_IOC	PB Port I/O Control Register	Selects the PB port I/O direction.
0x4418	PC_DAT	PC Port Input/Output Data Register	PC port input/output data
0x4419	PC_IOC	PC Port I/O Control Register	Selects the PC port I/O direction.
0x4420	P0_03_CFP	P00–P03 Port Function Select Register	Selects the P00–P03 port functions.
0x4421	P0_47_CFP	P04–P07 Port Function Select Register	Selects the P04–P07 port functions.
0x4422	P1_03_CFP	P10–P13 Port Function Select Register	Selects the P10–P13 port functions.
0x4423	P1_46_CFP	P14–P16 Port Function Select Register	Selects the P14–P16 port functions.
0x4424	P2_03_CFP	P20–P23 Port Function Select Register	Selects the P20–P23 port functions.
0x4425	P2_47_CFP	P24–P27 Port Function Select Register	Selects the P24–P27 port functions.
0x4426	P3_03_CFP	P30–P33 Port Function Select Register	Selects the P30–P33 port functions.
0x4427	P3_47_CFP	P34–P37 Port Function Select Register	Selects the P34–P37 port functions.
0x4428	P4_03_CFP	P40–P43 Port Function Select Register	Selects the P40–P43 port functions.
0x4429	P4_45_CFP	P44–P45 Port Function Select Register	Selects the P44–P45 port functions.
0x442a	P5_03_CFP	P50–P53 Port Function Select Register	Selects the P50–P53 port functions.
0x442b	P5_47_CFP	P54–P57 Port Function Select Register	Selects the P54–P57 port functions.
0x442c	P6_03_CFP	P60–P63 Port Function Select Register	Selects the P60–P63 port functions.
0x442d	P6_47_CFP	P64–P67 Port Function Select Register	Selects the P64–P67 port functions.
0x442e	P7_03_CFP	P70–P73 Port Function Select Register	Selects the P70–P73 port functions.
0x442f	P7_47_CFP	P74–P77 Port Function Select Register	Selects the P74–P77 port functions.
0x4430	P8_03_CFP	P80–P83 Port Function Select Register	Selects the P80–P83 port functions.
0x4431	P8_46_CFP	P84–P86 Port Function Select Register	Selects the P84–P86 port functions.
0x4432	P9_03_CFP	P90–P93 Port Function Select Register	Selects the P90–P93 port functions.
0x4433	P9_47_CFP	P94–P97 Port Function Select Register	Selects the P94–P97 port functions.
0x4434	PA_03_CFP	PA0–PA3 Port Function Select Register	Selects the PA0–PA3 port functions.
0x4435	PA_46_CFP	PA4–PA6 Port Function Select Register	Selects the PA4–PA6 port functions.
0x4436	PB_03_CFP	PB0–PB3 Port Function Select Register	Selects the PB0–PB3 port functions.
0x4437	PB_47_CFP	PB4–PB7 Port Function Select Register	Selects the PB4–PB7 port functions.
0x4438	PC_03_CFP	PC0–PC3 Port Function Select Register	Selects the PC0–PC3 port functions.
0x4439	PC_47_CFP	PC4–PC7 Port Function Select Register	Selects the PC4–PC7 port functions.
0x4440	PINTSEL0	Port Input Interrupt 0 Select Register	Selects a P*x*0 port for input interrupt.
0x4441	PINTSEL1	Port Input Interrupt 1 Select Register	Selects a P*x*1 port for input interrupt.
0x4442	PINTSEL2	Port Input Interrupt 2 Select Register	Selects a P*x*2 port for input interrupt.
0x4443	PINTSEL3	Port Input Interrupt 3 Select Register	Selects a P*x*3 port for input interrupt.
0x4444	PINTSEL4	Port Input Interrupt 4 Select Register	Selects a P*x*4 port for input interrupt.
0x4445	PINTSEL5	Port Input Interrupt 5 Select Register	Selects a P*x*5 port for input interrupt.
0x4446	PINTSEL6	Port Input Interrupt 6 Select Register	Selects a P*x*6 port for input interrupt.
0x4447	PINTSEL7	Port Input Interrupt 7 Select Register	Selects a P*x*7 port for input interrupt.

The following describes each I/O port register. These are all 8-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x4400–0x4418: P*x* Port Input/Output Data Registers (P*x*_DAT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
P*x* Port Input /Output Data Register (P*x*_DAT)	**0x4400 | 0x4418** (8 bits)	D7–0	**P*x*D[7:0]**	P*x*[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.

Note: The letter '*x*' in bit names, etc., denotes a port number from 0 to 9, A, B and C.

0x4400 P0 Port Input Data Register (P0_DAT) * Input only
0x4402 P1 Port Input/Output Data Register (P1_DAT)
0x4404 P2 Port Input/Output Data Register (P2_DAT)
0x4406 P3 Port Input/Output Data Register (P3_DAT)
0x4408 P4 Port Input/Output Data Register (P4_DAT)
0x440a P5 Port Input/Output Data Register (P5_DAT)
0x440c P6 Port Input/Output Data Register (P6_DAT)
0x440e P7 Port Input/Output Data Register (P7_DAT)
0x4410 P8 Port Input/Output Data Register (P8_DAT)
0x4412 P9 Port Input/Output Data Register (P9_DAT)
0x4414 PA Port Input/Output Data Register (PA_DAT)
0x4416 PB Port Input/Output Data Register (PB_DAT)
0x4418 PC Port Input/Output Data Register (PC_DAT)

These registers are used to read data from I/O-port pins or to set output data. (Default: external pin status)
1 (R/W): High level
0 (R/W): Low level

When an I/O port is set for output, the data written to the register is directly output to the I/O port pin. If the data written to the port is 1, the port pin is set high (V_{DD} level); if the data is 0, the port pin is set low (V_{SS} level).
Even in input mode, data can be written to the port data register.

When the register is read, the voltage level on the port pin is read out regardless of whether an I/O port is set for input or output mode. If the pin voltage is high (V_{DD} level), 1 is read out as input data; if the pin voltage is low (V_{SS} level), 0 is read out as input data.

Note: If noise may affect the input data, read it twice and verify that the read two data are the same.

0x4403–0x4419: P*x* I/O Control Registers (P*x*_IOC)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
P*x* I/O Control Register (P*x*_IOC)	**0x4403 | 0x4419** (8 bits)	D7–0	**IOC*x*[7:0]**	P*x*[7:0] I/O control	1	Output	0	Input	0	R/W	

Note: The letter '*x*' in bit names, etc., denotes a port number from 1 to 9, A, B and C.

0x4403 P1 I/O Control Register (P1_IOC)
0x4405 P2 I/O Control Register (P2_IOC)
0x4407 P3 I/O Control Register (P3_IOC)
0x4409 P4 I/O Control Register (P4_IOC)
0x440b P5 I/O Control Register (P5_IOC)
0x440d P6 I/O Control Register (P6_IOC)
0x440f P7 I/O Control Register (P7_IOC)
0x4411 P8 I/O Control Register (P8_IOC)
0x4413 P9 I/O Control Register (P9_IOC)
0x4415 PA I/O Control Register (PA_IOC)
0x4417 PB I/O Control Register (PB_IOC)
0x4419 PC I/O Control Register (PC_IOC)

Directs the I/O port for input or output and indicates the I/O control signal value of the port.
1 (R/W): Output mode
0 (R/W): Input mode (default)

Each I/O control register bit corresponds to each I/O port. When IOC*x* is set to 1, the corresponding I/O port is directed for output; if it is set to 0, the I/O port is directed for input.
When the pin is used for a peripheral function, the input/output direction depends on the peripheral function.

When the register is read, the I/O control signal value for the port pin is read out. When I/O port function is selected using the Port Function Select Register, the value written to the I/O Control Register is read out as is. When peripheral function is selected, the read value depends on the peripheral circuit status and may not indicate the value written to IOC*x*.

0x4420–0x4439: Port Function Select Registers (P*x*_*xx*_CFP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P*x*0–P*x*3 Port Function Select Register (P*x*_03_CFP) or P*x*4–P*x*7 Port Function Select Register (P*x*_47_CFP)	**0x4420 | 0x4439** (8 bits)	D7–6	**CFP*x*3[1:0] or CFP*x*7[1:0]**	P*x*3/P*x*7 port function select	CFP*x*3/7[1:0]	Function	0x0	R/W	
					0x3	Pin function 3			
					0x2	Pin function 2			
					0x1	Pin function 1			
					0x0	Pin function 0			
		D5–4	**CFP*x*2[1:0] or CFP*x*6[1:0]**	P*x*2/P*x*6 port function select	CFP*x*2/6[1:0]	Function	0x0	R/W	
					0x3	Pin function 3			
					0x2	Pin function 2			
					0x1	Pin function 1			
					0x0	Pin function 0			
		D3–2	**CFP*x*1[1:0] or CFP*x*5[1:0]**	P*x*1/P*x*5 port function select	CFP*x*1/5[1:0]	Function	0x0	R/W	
					0x3	Pin function 3			
					0x2	Pin function 2			
					0x1	Pin function 1			
					0x0	Pin function 0			
		D1–0	**CFP*x*0[1:0] or CFP*x*4[1:0]**	P*x*0/P*x*4 port function select	CFP*x*0/4[1:0]	Function	0x0	R/W	
					0x3	Pin function 3			
					0x2	Pin function 2			
					0x1	Pin function 1			
					0x0	Pin function 0			

Note: The letter '*x*' in bit names, etc., denotes a port number from 0 to 9, A, B and C.

0x4420 P00–P03 Port Function Select Register (P0_03_CFP)
0x4421 P04–P07 Port Function Select Register (P0_47_CFP)
0x4422 P10–P13 Port Function Select Register (P1_03_CFP)
0x4423 P14–P16 Port Function Select Register (P1_46_CFP)
0x4424 P20–P23 Port Function Select Register (P2_03_CFP)
0x4425 P24–P27 Port Function Select Register (P2_47_CFP)
0x4426 P30–P33 Port Function Select Register (P3_03_CFP)
0x4427 P34–P37 Port Function Select Register (P3_47_CFP)
0x4428 P40–P43 Port Function Select Register (P4_03_CFP)
0x4429 P44–P45 Port Function Select Register (P4_45_CFP)
0x442a P50–P53 Port Function Select Register (P5_03_CFP)
0x442b P54–P57 Port Function Select Register (P5_47_CFP)
0x442c P60–P63 Port Function Select Register (P6_03_CFP)
0x442d P64–P67 Port Function Select Register (P6_47_CFP)
0x442e P70–P73 Port Function Select Register (P7_03_CFP)
0x442f P74–P77 Port Function Select Register (P7_47_CFP)
0x4430 P80–P83 Port Function Select Register (P8_03_CFP)
0x4431 P84–P86 Port Function Select Register (P8_46_CFP)
0x4432 P90–P93 Port Function Select Register (P9_03_CFP)
0x4433 P94–P97 Port Function Select Register (P9_47_CFP)
0x4434 PA0–PA3 Port Function Select Register (PA_03_CFP)
0x4435 PA4–PA6 Port Function Select Register (PA_46_CFP)
0x4436 PB0–PB3 Port Function Select Register (PB_03_CFP)
0x4437 PB4–PB7 Port Function Select Register (PB_47_CFP)
0x4438 PC0–PC3 Port Function Select Register (PC_03_CFP)
0x4439 PC4–PC7 Port Function Select Register (PC_47_CFP)

These bits select the function of each I/O port pin. (Default: 0x0 = Pin function 0)
The I/O ports concurrently serve as the input/output pins for peripheral circuits or bus signals. Whether they are used as I/O ports or for peripheral circuits/bus signals can be selected bit-for-bit using these registers. All pins not used for peripheral circuits/bus signals can be used as general-purpose I/O ports.
For details of pin functions, see Section I.3.3, "Switching Over the Multiplexed Pin Functions."

0x4440–0x4447: Port Input Interrupt *x* Select Registers (PINTSEL*x*)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Port Input Interrupt *x* Select Register (PINTSEL*x*)	**0x4440** | **0x4447** (8 bits)	D7–4	**PINT*x*[3:0]**	Interrupt port select	PINT*x*[3:0]	Function	0xf	R/W	
					0xc	PC*x*			
					0xb	PB*x*			
					0xa	PA*x*			
					0x9	P9*x*			
					0x8	P8*x*			
					0x7	P7*x*			
					0x6	P6*x*			
					0x5	P5*x*			
					0x4	P4*x*			
					0x3	P3*x*			
					0x2	P2*x*			
					0x1	P1*x*			
					0x0	P0*x*			
		D3–0	–	reserved	–		X	R	

Note: The letter '*x*' in bit names, etc., denotes a port input interrupt number from 0 to 7.

0x4440 Port Input Interrupt 0 Select Register (PINTSEL0)
0x4441 Port Input Interrupt 1 Select Register (PINTSEL1)
0x4442 Port Input Interrupt 2 Select Register (PINTSEL2)
0x4443 Port Input Interrupt 3 Select Register (PINTSEL3)
0x4444 Port Input Interrupt 4 Select Register (PINTSEL4)
0x4445 Port Input Interrupt 5 Select Register (PINTSEL5)
0x4446 Port Input Interrupt 6 Select Register (PINTSEL6)
0x4447 Port Input Interrupt 7 Select Register (PINTSEL7)

PINT*x*[3:0]: Interrupt Port Select Bits

Selects an I/O port used to generate the port input interrupt.

Table VII.1.6.2 Selecting Pins for Port Input Interrupts

Port input interrupt system	PINT*x*[3:0] settings												
	0x0	**0x1**	**0x2**	**0x3**	**0x4**	**0x5**	**0x6**	**0x7**	**0x8**	**0x9**	**0xa**	**0xb**	**0xc**
0	P00	P10	P20	P30	P40	P50	P60	P70	P80	P90	PA0	PB0	PC0
1	P01	P11	P21	P31	P41	P51	P61	P71	P81	P91	PA1	PB1	PC1
2	P02	P12	P22	P32	P42	P52	P62	P72	P82	P92	PA2	PB2	PC2
3	P03	P13	P23	P33	P43	P53	P63	P73	P83	P93	PA3	PB3	PC3
4	P04	P14	P24	P34	P44	P54	P64	P74	P84	P94	PA4	PB4	PC4
5	P05	P15	P25	P35	P45	P55	P65	P75	P85	P95	PA5	PB5	PC5
6	P06	P16	P26	P36	–	P56	P66	P76	P86	P96	PA6	PB6	PC6
7	P07	–	P27	P37	–	P57	P67	P77	–	P97	–	PB7	PC7

(Default: 0xf)

VII.1.7 Precautions

- The interrupt vector numbers 12, 14–16, and 18–19 for port input interrupts 4, 6, and 7 are shared with another cause of interrupt. To use the vector number for the port input interrupt, set the interrupt enable bit for the port input interrupt to 1 and clear one for another to 0.
- When using a port input interrupt as the trigger to restart from SLEEP mode, the interrupt trigger mode must be set to level interrupt.
  If it is set to active high level interrupt, the CPU restarts when the input signal goes to a high level.
  If it is set to active low level interrupt, the CPU restarts when the input signal goes to a low level.
  Therefore, design the system assuming that the CPU can restart normally due to the signal level at the interrupt port, not an edge interrupt, when restarting the CPU from SLEEP mode using a port input interrupt.
- When pulse trigger interrupt mode is selected, the input pulse width must be longer than 1 cycle of the system clock to be certain an interrupt will be generated.
- If noise may affect the input data, read it twice and verify that the read two data are the same.
- When using the port input interrupt (level interrupt), the input signal must be set to the active level for more than the time shown below.

  (1) When the clock is stopped during SLEEP mode
  OSC3 oscillation start time + OSC3 oscillation stabilization wait time (set by the user) + 10 system clock cycle time

  (2) When the clock is not stopped during SLEEP mode, or in HALT mode
  10 system clock cycle time

  (3) During normal operation
  10 system clock cycle time

VIII S1C17501 ANALOG MODULE

VIII.1 A/D Converter (ADC)

VIII.1.1 Features and Structure of A/D Converter

The S1C17501 contains an A/D converter with the following features:

- Conversion method: Successive comparison
- Resolution: 10 bits
- Input channels: 8 channels
- A/D converter input clock: Maximum of 2 MHz, minimum of 16 kHz
- Conversion time: 9 clocks (sampling time) + 11 clocks (conversion time) = 20 clocks
  Minimum of 10 µs (when a 2-MHz input clock is selected)
  Maximum of 1250 µs (when a 16-kHz input clock is selected)
- Conversion range: Between Vss and AVDD
- Two conversion modes can be selected:
  Normal mode: Conversion is completed in one operation.
  Continuous mode: Conversion is continuous and terminated through software control.
  Continuous conversion of multiple channels can be performed in each mode.
- Four types of A/D-conversion start triggers can be selected:
  Triggered by the external pin (#ADTRG)
  Triggered by the period-match of the 16-bit multi-function timer (MFT)
  Triggered by the underflow of the 8-bit programmable timer (PT8) CH.0
  Triggered by the software
- A/D conversion results can be read out from the 10-bit data register or the conversion result buffer* for each channel.
- An interrupt is generated upon completion of A/D conversion or when the conversion result is out of the specified range (upper and lower-limit values can be specified)*.
- The upper-limit and lower-limit out-of-range signals of CH.0 are sent to the MFT for controlling IGBT.

* These functions can be used in the advanced mode. The A/D converter of the S1C17501 has two operating modes, standard mode of which functions are compatible with the legacy analog block for the existing models and an advanced mode allowing use of the extended functions.

Figure VIII.1.1.1 shows the structure of the A/D converter.

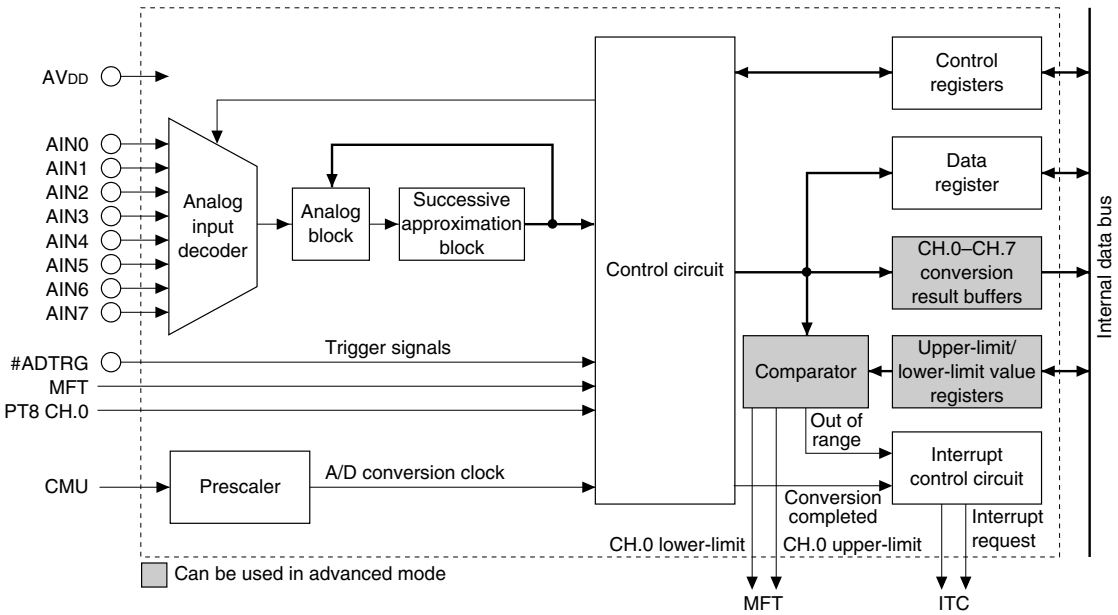


Figure VIII.1.1.1 Structure of A/D Converter

VIII.1.2 Input Pins of A/D Converter

Table VIII.1.2.1 shows the pins used by the A/D converter.

Table VIII.1.2.1 Input Pins of A/D Converter

Pin name	I/O	Function
AV_{DD}	–	Analog power-supply pin AV_{DD} is the power-supply pin for the analog circuit.
AIN0–AIN7	I	Analog signal input pins AIN0 (CH.0)–AIN7 (CH.7) The analog input voltage AV_{IN} can be input in the range of $V_{SS} \leq AV_{IN} \leq AV_{DD}$.
#ADTRG	I	External trigger input pin This pin is used to input a trigger signal to start A/D conversion from an external source.

The A/D converter input pins (AIN[7:0], #ADTRG) are shared with the input ports and they are initialized as general-purpose input port pins by default. Before using these pins for the A/D converter, the pin functions must be switched using the Port Function Select registers.
For details on switching pin function, Section I.3.3, "Switching Over the Multiplexed Pin Functions."

Notes: • When the A/D converter is enabled, a current flows between AV_{DD} and V_{SS}, and power is consumed, even when A/D operations are not performed. Therefore, when the A/D converter is not used, it must be disabled (default 0 setting of ADE (D2/AD_CTL register)).

∗ **ADE**: A/D Enable Bit in the A/D Control/Status (AD_CTL) Register (D2/0x5544)

• Take measures against noise when designing the #ADTRG signal path on the printed circuit board.

• Be aware that the interface voltage level is AV_{DD} even if the AIN*x* pin is used as an input port (P0*x*) pin.

VIII.1.3 A/D Converter Operating Clock

The A/D converter use the ADC_CLK clock (= PCLK) generated by the CMU as the operating clock. The conversion clock is generated in the A/D converter module.

Controlling the supply of the operating clock

ADC_CLK is supplied to the A/D converter with default settings. It can be turned off using ADC_CLK_EN (D3/CMU_GATEDCLK2 register) to reduce the amount of power consumed on the chip if the A/D converter is not used.

∗ **ADC_CLK_EN**: ADC Module Clock Control Bit in the Gated Clock Control 2 (CMU_GATEDCLK2) Register (D3/0x4908)

Setting ADC_CLK_EN to 0 (1 by default) turns off the clock supply to the A/D converter. When the clock supply is turned off, the A/D converter control registers cannot be accessed.
For details on how to set and control the clock, refer to Section II.2, "Clock Management Unit (CMU)."

Note: The Gated Clock Control 2 Register (0x4908) is write-protected. Write protection of this and other CMU registers at addresses 0x4900 to 0x4909 to be rewritten must be removed by writing 0x96 to the CMU Write Protect Register (0x4920). Since unnecessary rewrites to addresses 0x4900 to 0x4909 could cause the system to operate erratically, make sure the data set in the CMU Write Protect Register (0x4920) is other than 0x96, unless rewriting said registers.

Clock state in standby mode

The clock supply to the A/D converter stops depending on type of standby mode.
HALT mode: The operating clock is supplied the same way as in normal mode.
SLEEP mode: The operating clock supply stops.

Therefore, the A/D converter also stops operating in SLEEP mode.

VIII.1.4 Setting A/D Converter

When the A/D converter is used, the following settings must be made before an A/D conversion can be performed:
1. Setting analog input pins ... See Sections VIII.1.2 and I.3.3.
2. Setting the operating mode (standard mode/advanced mode)
3. Setting the input clock
4. Selecting the analog-conversion start and end channels
5. Setting the A/D conversion mode
6. Selecting a trigger
7. Setting the sampling time
8. Setting the upper-limit and lower-limit values (advanced mode)
9. Setting the interrupt mode (advanced mode)
10. Setting interrupt ... See Section VIII.1.6.

Note: Before making these settings, make sure the A/D converter is disabled (ADE (D2/AD_CTL register) = 0). Changing the settings while the A/D converter is enabled could cause a malfunction.

∗ **ADE**: A/D Enable Bit in the A/D Control/Status (AD_CTL) Register (D2/0x5544)

Setting the operating mode (standard mode / advanced mode)

The A/D converter of the S1C17501 has two operating modes, standard mode of which functions are compatible with the legacy analog block for the existing models and an advanced mode allowing use of the extended functions. Table VIII.1.4.1 shows differences between the standard mode and the advanced mode.

Table VIII.1.4.1 Differences Between Standard Mode and Advanced Mode

Function	Standard mode	Advanced mode
Reading conversion results	The conversion results are read from the A/D conversion result register common to all channels. When converting for multiple channels, the A/D conversion result register must be read before conversion for the next channel has completed.	The conversion results can be read from the conversion result buffer provided for each channel. Thus the conversion result for the current channel will not be lost even when the conversion for the next channel is completed during a multiple channel conversion.
Conversion-complete flag, overwrite error flag	One bit is assigned for the flag and is commonly used in all channels.	Different flags are provided for each channel.
Comparison with upper/lower-limit values	Not supported.	An upper-limit value and a lower-limit value can be set and conversion results of the specified channel can be checked whether they are within the specified range or not.
Interrupts	Conversion-complete interrupt only can be generated. The interrupts cannot be masked in channel units.	Conversion-complete interrupts and out-of-range interrupts can be generated. Conversion complete interrupts for the specified channels can be masked.

To configure the A/D converter in the advanced mode, set ADCADV (D8/AD_ADVMODE register) to 1. The control bits for the extended functions can be accessed after this setting. At initial reset, ADCADV is set to 0 and the A/D converter enters the standard mode.

∗ **ADCADV**: Standard/Advanced Mode Selection Bit in the A/D Converter Mode Select/Internal Status (AD_ADVMODE) Register (D8/0x555e)

The following descriptions unless otherwise specified are common contents for both modes.
The extended functions in the advanced mode are explained assuming that ADCADV has been set to 1.

Setting the input clock

The A/D converter contains a prescaler and the A/D conversion clock can be selected from among the eight types shown in Table VIII.1.4.2 below. Use PSAD[2:0] (D[2:0]/AD_CLKCTL register) for this selection.

* **PSAD[2:0]**: A/D Converter Clock Division Ratio Selection Bits in the A/D Clock Control (AD_CLKCTL) Register (D[2:0]/0x5520)

Table VIII.1.4.2 Input Clock Selection

PSAD[2:0]	A/D clock
0x7	PCLK•1/256
0x6	PCLK•1/128
0x5	PCLK•1/64
0x4	PCLK•1/32
0x3	PCLK•1/16
0x2	PCLK•1/8
0x1	PCLK•1/4
0x0	PCLK•1/2

(Default: 0x0)

The selected clock is output from the prescaler by writing 1 to PSONAD (D3/AD_CLKCTL register).

* **PSONAD**: A/D Converter Clock Control Bit in the A/D Clock Control (AD_CLKCTL) Register (D3/0x5520)

Notes:
- The recommended input clock frequency is a maximum of 2 MHz and a minimum of 16 kHz.
- Do not start an A/D conversion when the clock output from the prescaler is turned off, and do not turn off the prescaler's clock output when an A/D conversion is underway. This could cause the A/D converter to operate erratically.

Selecting analog-conversion start and end channels

Select the channel in which the A/D conversion is to be performed from among the pins (channels) that have been set for analog input. To enable A/D conversions in multiple channels to be performed successively through one convert operation, specify the conversion start and conversion end channels using CS[2:0] (D[10:8]/AD_TRIG_CH register) and CE[2:0] (D[13:11]/AD_TRIG_CH register) respectively.

* **CS[2:0]**: A/D Converter Start Channel Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[10:8]/0x5542)
* **CE[2:0]**: A/D Converter End Channel Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[13:11]/0x5542)

Table VIII.1.4.3 Relationship between CS/CE and Input Channel

CS[2:0]/CE[2:0]	Channel selected
0x7	AIN7
0x6	AIN6
0x5	AIN5
0x4	AIN4
0x3	AIN3
0x2	AIN2
0x1	AIN1
0x0	AIN0

(Default: 0x0)

Example: Operation of one A/D conversion

CS[2:0] = 0, CE[2:0] = 0: Converted only in AIN0

CS[2:0] = 0, CE[2:0] = 3: Converted in the following order: AIN0→AIN1→AIN2→AIN3

CS[2:0] = 6, CE[2:0] = 1: Converted in the following order: AIN6→AIN7→AIN0→AIN1

Setting the A/D conversion mode

The A/D converter can operate in one of the following two modes. This operation mode is selected using MS (D5/AD_TRIG_CH register).

* **MS**: A/D Conversion Mode Selection Bit in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D5/0x5542)

1. Normal mode (MS = 0)

All inputs in the range of channels set using CS[2:0] (D[10:8]/AD_TRIG_CH register) and CE[2:0] (D[13:11]/AD_TRIG_CH register) are A/D converted once and then stopped.

2. Continuous mode (MS = 1)

A/D conversions in the range of channels set using CS[2:0] and CE[2:0] are executed successively until stopped by the software.

At initial reset, the normal mode is selected.

Selecting a trigger

Use TS[1:0] (D[4:3]/AD_TRIG_CH register) to select a trigger to start A/D conversion from among the four types shown in Table VIII.1.4.4.

* **TS[1:0]**: A/D Conversion Trigger Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[4:3]/0x5542)

Table VIII.1.4.4 Trigger Selection

TS[1:0]	Trigger source
0x3	External trigger (#ADTRG pin)
0x2	8-bit programmable timer (PT8) CH.0
0x1	16-bit multi-function timer (MFT)
0x0	Software trigger

(Default: 0x0)

1. External (#ADTRG) trigger

The signal input to the #ADTRG pin is used as a trigger. When this trigger is used, the #ADTRG pin must be set in advance using the port function select register. A/D conversion is started when a low level of the #ADTRG signal is detected.

2. 8-bit programmable timer (PT8) CH.0

The underflow signal of the PT8 CH.0 is used as a trigger. Since the cycle can be programmed using the timer, this trigger is effective when cyclic A/D conversions are required.
For details on how to set the timer, refer to the explanation of the PT8 in this manual.

3. 16-bit multi-function timer (MFT)

The period-match signal of the 16-bit multi-function timer (MFT) is used as a trigger. Since the cycle can be programmed using the timer, this trigger is effective when cyclic A/D conversions are required.
For details on how to set the timer, refer to the explanation of the MFT in this manual.

4. Software trigger

Writing 1 to ADST (D1/AD_CTL register) in the software serves as a trigger to start A/D conversion.

* **ADST**: A/D Conversion Control/Status Bit in the A/D Control/Status (AD_CTL) Register (D1/0x5544)

Setting the sampling time

The A/D converter contains ST[1:0] (D[9:8]/AD_CTL register) that allows the analog-signal input sampling time to be set in four steps (3, 5, 7, or 9 times the conversion clock period).
However, this register should be used as set by default (ST[1:0] = 11; 9 clock periods).

* **ST[1:0]**: Input Signal Sampling Time Setup Bits in the A/D Control/Status (AD_CTL) Register (D[9:8]/0x5544)

Setting the upper-limit and lower-limit values (advanced mode)

The advanced mode allows a range check of the conversion results by setting the upper-limit and lower-limit values. When the conversion result is out of the range, the A/D converter can output the interrupt signal to the interrupt controller (ITC). Furthermore, the A/D converter CH.0 outputs the upper-limit out-of-range or lower-limit out-of-range signal to the MFT for controlling IGBT (see Section V.2 for more information).

1. Selecting the channel

Select the channel to compare the A/D conversion results and the upper-limit and lower-limit value using ADCMP[2:0] (D[14:12]/AD_CTL register).

* **ADCMP[2:0]**: A/D Upper/Lower-Limit Comparison Channel Selection Bits in the A/D Control/Status (AD_CTL) Register (D[14:12]/0x5544)

Table VIII.1.4.5 Selecting the Channel for Checking Conversion Results

ADCMP[2:0]	Channel selected
0x7	AIN7
0x6	AIN6
0x5	AIN5
0x4	AIN4
0x3	AIN3
0x2	AIN2
0x1	AIN1
0x0	AIN0

(Default: 0x0)

ADCMP[2:0] selects the channel to generate the out-of-range interrupt. The A/D converter CH.0 can output the out-of-range signal to the MFT even if another channel is selected by ADCMP[2:0].

2. Setting upper-limit and lower-limit values

Set the upper-limit value to ADUPR[9:0] (D[9:0]/AD_UPPER register) and the lower-limit value to ADLWR[9:0] (D[9:0]/AD_LOWER register).

* **ADUPR[9:0]**: A/D Conversion Upper Limit Value Bits in the A/D Upper Limit Value (AD_UPPER) Register (D[9:0]/0x5558)
* **ADLWR[9:0]**: A/D Conversion Lower Limit Value Bits in the A/D Lower Limit Value (AD_LOWER) Register (D[9:0]/0x555a)

When the conversion result exceeds the upper-limit value set or is lower than the lower-limit value, it is determined as out of range. If the conversion result is the same value as the upper-limit or lower-limit value, it is determined as within the range.

3. Enabling comparison with the upper-limit and lower-limit values

Set ADCMPE (D15/AD_CTL register) to 1 to enable the range check function.

* **ADCMPE**: Upper/Lower-Limit Comparison Enable Bit in the A/D Control/Status (AD_CTL) Register (D15/0x5544)

The A/D converter CH.0 can output the out-of-range signal to the MFT when the ADCMPE is set.

Setting the interrupt mode (advanced mode)

The interrupt functions are extended in the advanced mode, so the following configuration is necessary.

1. Enabling/disabling the conversion-complete interrupt

The conversion-complete interrupt can be enabled/disabled using CNVINTEN (D4/AD_CTL register). Set CNVINTEN to 1 when using the conversion-complete interrupt, or to 0 when it is not used. At initial reset, CNVINTEN is set to 1, so the conversion-complete interrupt function is enabled.

* **CNVINTEN**: Conversion-Complete Interrupt Enable Bit in the A/D Control/Status (AD_CTL) Register (D4/0x5544)

2. Enabling/disabling the out-of-range interrupt

The out-of-range interrupt can be enabled/disabled using CMPINTEN (D5/AD_CTL register). Set CMPINTEN to 1 when using the out-of range interrupt, or to 0 when it is not used. At initial reset, CMPINTEN is set to 0, so the out-of-range interrupt function is disabled.

* **CMPINTEN**: Out-of-Range Interrupt Enable Bit in the A/D Control/Status (AD_CTL) Register (D5/0x5544)

3. Setting the interrupt signal mode

The S1C17501 A/D converter has two interrupt request outputs for the interrupt sources above and each interrupt can be handled individually. In the initial setting, the out-of-range interrupt signal is ORed with the conversion-complete interrupt signal to send to the ITC. So, the conversion-complete interrupt flag in the ITC is set when an A/D conversion has completed or when the conversion results are out of range. This signal mode can be canceled using INTMODE (D6/AD_CTL register). To handle each interrupt individually, set INTMODE to 1. In this setting, the out-of-range interrupt signal is not ORed with the conversion-complete interrupt signal.

* **INTMODE**: Interrupt Signal Mode Bit in the A/D Control/Status (AD_CTL) Register (D6/0x5544)

4. Masking conversion-complete interrupt for the specified channels

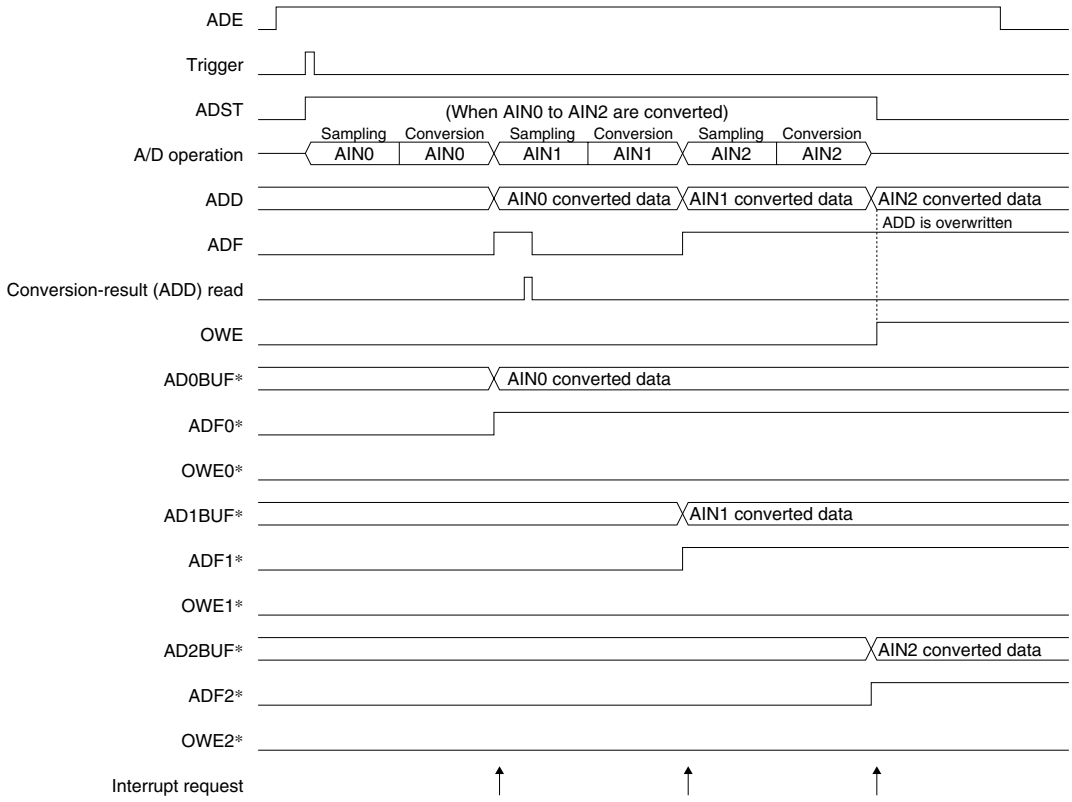
The A/D conversion-complete interrupt mask register is used to mask the conversion-complete interrupts of the specified channels. When INTMASK*x* (D*x*/AD_INTMASK register) for channel '*x*' in the register is set to 0, channel '*x*' does not generate conversion-complete interrupts. For instance, by masking the conversion-complete interrupt of the channel used for range checking, it is possible to generate out-of range interrupts only.

* **INTMASK*x***: CH.*x* Conversion-Complete Interrupt Mask Bit in the A/D Conversion Complete Interrupt Mask (AD_INTMASK) Register (D*x*/0x555c)

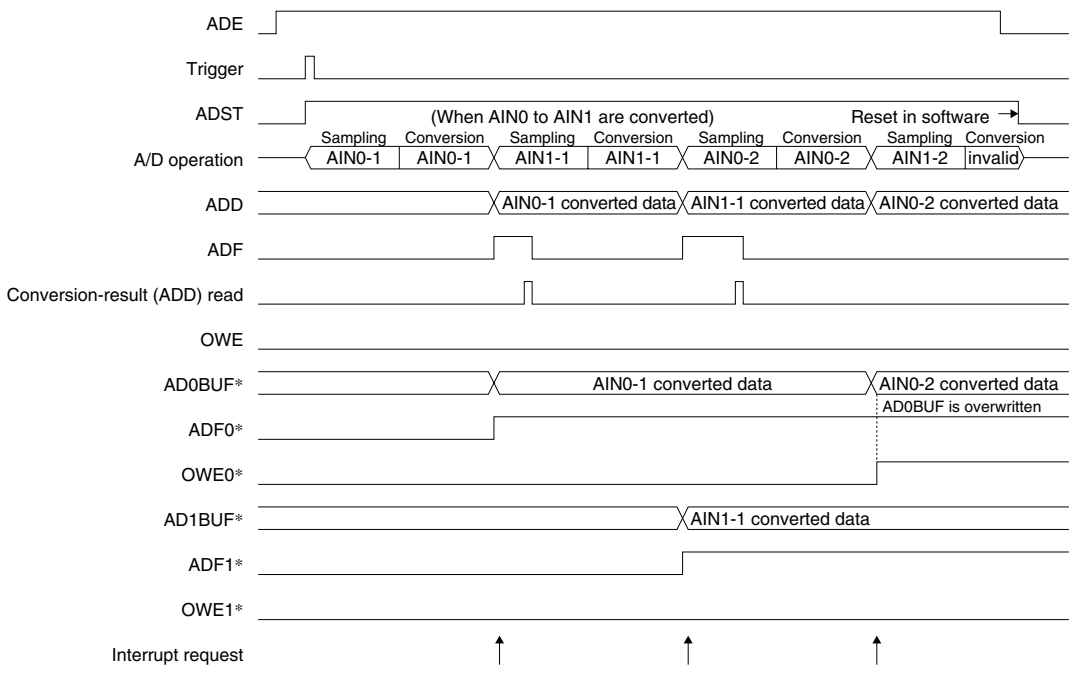
At initial reset, INTMASK*x* are all set to 1 to enable conversion-complete interrupts.

VIII.1.5 Control and Operation of A/D Conversion

Figure VIII.1.5.1 shows the operation of the A/D converter.



(1) Normal mode



(2) Continuous mode

∗ Extended functions that can be used when ADCADV = 1

Figure VIII.1.5.1 Operation of A/D Converter

Starting up the A/D converter circuit

After the settings specified in the preceding section have been made, write 1 to ADE (D2/AD_CTL register) to enable the A/D converter. The A/D converter is thereby ready to accept a trigger to start A/D conversion. To set the A/D converter again, or if it is not used, set ADE to 0.

∗ **ADE**: A/D Enable Bit in the A/D Control/Status (AD_CTL) Register (D2/0x5544)

Starting A/D conversion

When a trigger is input while ADE = 1, A/D conversion is started. If a software trigger has been selected, A/D conversion is started by writing 1 to ADST (D1/AD_CTL register).

∗ **ADST**: A/D Conversion Control/Status Bit in the A/D Control/Status (AD_CTL) Register (D1/0x5544)

Only the trigger selected using TS[1:0] (D[4:3]/AD_TRIG_CH register) are valid; no other trigger is accepted.

∗ **TS[1:0]**: A/D Conversion Trigger Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[4:3]/0x5542)

When a trigger is input, the A/D converter samples and A/D-converts the analog input signal, beginning with the conversion start channel selected by CS[2:0] (D[10:8]/AD_TRIG_CH register).

∗ **CS[2:0]**: A/D Converter Start Channel Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[10:8]/0x5542)

ADST used for the software trigger is set to 1 during A/D conversion, even when it is started by some other trigger, so it can be used as an A/D-conversion status bit.
The channel in which conversion is underway can be identified by reading CH[2:0] (D[2:0]/AD_TRIG_CH register).

∗ **CH[2:0]**: A/D Conversion Channel Status Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[2:0]/0x5542)

Reading out A/D conversion results

- **Standard mode**

Upon completion of the A/D conversion in the start channel, the A/D converter stores the conversion result into the 10-bit data registers ADD[9:0] (D[9:0]/AD_DAT register) and sets the conversion-complete flag ADF (D3/AD_CTL register). If multiple channels are specified using CS[2:0] (D[10:8]/AD_TRIG_CH register) and CE[2:0] (D[13:11]/AD_TRIG_CH register), A/D conversions in the subsequent channels are performed in succession.

∗ **ADD[9:0]**: A/D Converted Data Bits in the A/D Conversion Result (AD_DAT) Register (D[9:0]/0x5540)
∗ **ADF**: Conversion-Complete Flag Bit in the A/D Control/Status (AD_CTL) Register (D3/0x5544)
∗ **CE[2:0]**: A/D Converter End Channel Selection Bits in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D[13:11]/0x5542)

The results of A/D conversion are stored in ADD[9:0] each time conversion in one channel is completed. Since an interrupt can be generated simultaneously, this interrupt is normally used to read out the converted data. In addition, be sure to reset the interrupt flag (by writing 0) to prepare the A/D converter for the next operation.

If multiple A/D conversion channels are specified, the conversion results in one channel must be read out prior to completion of conversion in the next channel. If the A/D conversion currently under way is completed before the previous conversion results are read out, ADD[9:0] is overwritten with the new conversion results.
If ADD[9:0] is updated when the conversion-complete flag ADF (D3/AD_CTL register) = 1 (before the converted data is read out), the overwrite-error flag OWE (D0/AD_CTL register) is set to 1. The conversion-complete flag ADF is reset to 0 when the converted data is read out. If ADD[9:0] is updated when ADF = 0, OWE remains at 0, indicating that the operation has been completed normally. When reading out data, also read OWE to make sure the data is valid. Once OWE is set, it remains set until it is reset to 0 in the software. Note also that if OWE is set, ADF also is set. In this case, read out the converted data and reset ADF.

∗ **OWE**: Overwrite Error Flag Bit in the A/D Control/Status (AD_CTL) Register (D0/0x5544)

- **Advanced mode**

  Upon completion of the A/D conversion in the start channel (CH.*x*), the A/D converter stores the conversion result to the 10-bit CH.*x* conversion result buffer AD*x*BUF[9:0] (D[9:0]/AD_CH*x*_BUF register) and sets the CH.*x* conversion-complete flag ADF*x* (D*x*/AD_CH_STAT register). If multiple channels are specified using CS[2:0] (D[10:8]/AD_TRIG_CH register) and CE[2:0] (D[13:11]/AD_TRIG_CH register), A/D conversions in the subsequent channels are performed in succession.

  * **AD*x*BUF[9:0]**: A/D CH.*x* Converted Data Bits in the A/D CH.*x* Conversion Result Buffer (AD_CH*x*_BUF) Register (D[9:0]/0x5548 + 2•*x*)
  * **ADF*x***: CH.*x* Conversion-Complete Flag Bit in the A/D Channel Status Flag (AD_CH_STAT) Register (D*x*/0x5546)

  The results of A/D conversion are stored in the A/D conversion result buffer for each channel each time conversion in one channel is completed. Since an interrupt can be generated simultaneously, this interrupt is normally used to read out the converted data. In addition, be sure to reset the interrupt flag (by writing 0) to prepare the A/D converter for the next operation.
  In the advanced mode, each channel has a conversion result buffer, so it is not necessary to read the conversion results prior to completion of conversion in the next channel. However, if the next A/D conversion in the same channel is completed before the previous conversion results are read out, the conversion result buffer is overwritten with the new conversion results. If AD*x*BUF[9:0] is updated when the conversion-complete flag ADF*x* = 1 (before the converted data is read out), the overwrite-error flag OWE*x* (D*x* + 8/AD_CH_STAT register) is set to 1. ADF*x* is reset to 0 when the converted data is read out. If AD*x*BUF[9:0] is updated when ADF*x* = 0, OWE*x* remains at 0, indicating that the operation has been completed normally. When reading out data, also read OWE*x* to make sure the data is valid. Once OWE*x* is set, it remains set until it is reset to 0 by writing 0 in the software. Note also that if OWE*x* is set, ADF*x* is also set. In this case, read out the converted data and reset ADF*x*.

  * **OWE*x***: CH.*x* Overwrite Error Flag Bit in the A/D Channel Status Flag (AD_CH_STAT) Register (D*x* + 8/0x5546)

  ADD[9:0] (D[9:0]/AD_DAT register), ADF (D3/AD_CTL register) and OWE (D0/AD_CTL register) used in the standard mode are also effective in the advanced mode as well. The functions and actions of the register/bits are the same as those of the standard mode. OWE is set during conversion in multiple-channels, but it is not necessary to reset it.

Range check (comparison with upper-limit/lower-limit values in advanced mode)

When the range check function is enabled (ADCMPE (D15/AD_CTL register) = 1) and an A/D conversion in the channel specified using ADCMP[2:0] (D[14:12]/AD_CTL register) has completed, the conversion results are compared with the contents of ADUPR[9:0] (D[9:0]/AD_UPPER register) and ADLWR[9:0] (D[9:0]/AD_LOWER register).

* **ADCMPE**: Upper/Lower-Limit Comparison Enable Bit in the A/D Control/Status (AD_CTL) Register (D15/0x5544)
* **ADCMP[2:0]**: Upper/Lower-Limit Comparison Channel Selection Bits in the A/D Control/Status (AD_CTL) Register (D[14:12]/0x5544)
* **ADUPR[9:0]**: A/D Upper Limit Value Setup Bits in the A/D Upper Limit Value (AD_UPPER) Register (D[9:0]/0x5558)
* **ADLWR[9:0]**: A/D Lower Limit Value Setup Bits in the A/D Lower Limit Value (AD_LOWER) Register (D[9:0]/0x555a)

If the conversion results exceed the upper-limit value, the upper-limit comparison status bit ADUPRST (D11/AD_CTL register) is set to 1. If the results are less than the lower-limit value, the lower-limit comparison status bit ADLWRST (D10/AD_CTL register) is set to 1. When the out-of range interrupt is enabled, an interrupt occurs if one of the status bits has been set.

* **ADUPRST**: Upper-Limit Comparison Status Bit in the A/D Control/Status (AD_CTL) Register (D11/0x5544)
* **ADLWRST**: Lower-Limit Comparison Status Bit in the A/D Control/Status (AD_CTL) Register (D10/0x5544)

When the conversion results are the same as the upper-limit or lower-limit values, it is assumed within the range and an interrupt is not generated.

When the A/D converter CH.0 has finished an A/D conversion, the range check is always performed regardless of how the ADCMP[2:0] is set. If the conversion results exceed the upper-limit value, the upper-limit out-of-range signal is output to the MFT. If the results are less than the lower-limit value, the lower-limit out-of-range signal is output to the MFT.

Terminating A/D conversion

- **For normal mode (MS = 1)**

  In the normal mode, A/D conversion is performed successively from the conversion start channel specified using CS[2:0] (D[10:8]/AD_TRIG_CH register) to the conversion end channel specified using CE[2:0] (D[13:11]/AD_TRIG_CH register), and is completed after these conversions are executed in one operation. ADST (D1/AD_CTL register) is reset to 0 upon completion of the conversion.

  * **MS**: A/D Conversion Mode Selection Bit in the A/D Trigger/Channel Select (AD_TRIG_CH) Register (D5/0x5542)

- **For continuous mode (MS = 0)**

  In the continuous mode, A/D conversion from the conversion-start to the conversion-end channels is executed repeatedly, without being stopped in the hardware. To terminate conversion, therefore, ADST (D1/AD_CTL register) must be reset to 0 in the software. However, the A/D conversion being executed will be completed normally or forcibly stopped depending on the timing of writing 0 to ADST. When the A/D conversion has completed normally, ADF (D3/AD_CTL register) is set to 1 and the conversion results can be obtained. If it is forcibly stopped, ADF maintains its previous status, therefore, conversion results cannot be obtained.

- **Forced termination**

  A/D conversion is immediately terminated by writing 0 to ADST. The results of the conversion then under-way cannot be obtained.

VIII.1.6 A/D Converter Interrupt

The A/D converter can generate the following two types of interrupts:
• Conversion-complete interrupt
• Out-of-range interrupt

Conversion-complete interrupt

When A/D conversion in one channel has completed, the A/D converter outputs the conversion-complete interrupt signal to the interrupt controller (ITC) to request an interrupt.
When using this interrupt in advanced mode, set CNVINTEN (D4/AD_CTL register) to 1. If CNVINTEN is set to 0, an interrupt request by this cause will not be sent to the ITC. In standard mode, it is not necessary to set CNVINTEN.

∗ **CNVINTEN**: Conversion-Complete Interrupt Enable Bit in the A/D Control/Status (AD_CTL) Register (D4/0x5544)

If other interrupt conditions are satisfied, an interrupt is generated.

In advanced mode, the specified channels can be configured to disable interrupt generation using INTMASK*x* (D*x*/AD_INTMASK register). When INTMASK*x* is set to 0, the A/D converter CH.*x* does not generate conversion-complete interrupts.

∗ **INTMASK*x***: CH.*x* Conversion-Complete Interrupt Mask Bit in the A/D Conversion Complete Interrupt Mask (AD_INTMASK) Register (D*x*/0x555c)

Out-of-range interrupt (advanced mode)

When the range check function has been enabled and the conversion results are out of the range from the lower- to upper limits that have been set with software, the A/D converter outputs the out-of-range interrupt signal to the ITC to request an interrupt.
When using this interrupt, set the A/D converter to advanced mode and set CMPINTEN (D5/AD_CTL register) to 1. If CMPINTEN is set to 0, an interrupt request by this cause will not be sent to the ITC.

∗ **CMPINTEN**: Out-of-Range Interrupt Enable Bit in the A/D Control/Status (AD_CTL) Register (D5/0x5544)

By default, the A/D converter uses the conversion-complete interrupt signal line to send an out-of-range interrupt request to the ITC.
So, the conversion-complete interrupt flag in the ITC is set when an A/D conversion has completed or when the conversion results are out of range. To generate the out-of-range interrupt independently of the conversion complete interrupt, set INTMODE (D6/AD_CTL register) to 1.

∗ **INTMODE**: Interrupt Signal Mode Bit in the A/D Control/Status (AD_CTL) Register (D6/0x5544)

In standard mode, this interrupt cannot be used.

ITC registers for A/D converter interrupts

Table VIII.1.6.1 shows the control registers of the ITC provided for each cause of A/D converter interrupt.

Table VIII.1.6.1 ITC Registers

Cause of interrupt	Interrupt flag	Interrupt enable bit	Interrupt level setup bits
End of conversion	AIFT3 (D3/ITC_AIFLG)	AIEN3 (D3/ITC_AEN)	AILV3[2:0] (D[10:8]/ITC_AILV1)
Out of range	AIFT2 (D2/ITC_AIFLG)	AIEN2 (D2/ITC_AEN)	AILV2[2:0] (D[2:0]/ITC_AILV1)

ITC_AIFLG register (0x42e0)
ITC_AEN register (0x42e2)
ITC_AILV1 register (0x42e8)

When the A/D converter outputs an interrupt signal, the corresponding interrupt flag is set to 1.
If the interrupt enable bit corresponding to that interrupt flag has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the A/D converter interrupt, set the interrupt enable bit to 0.
The interrupt flag is always set to 1 by the interrupt signal, regardless of how the interrupt enable bit is set (even when set to 0).
The interrupt level setup bits set the interrupt level (0 to 7) of the A/D converter interrupt. If the same interrupt level is set, the out-of-range interrupt has higher priority than the conversion-complete interrupt.
An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The A/D converter interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vectors

The following shows the vector numbers and vector addresses for the A/D converter interrupts:

Table VIII.1.6.2 A/D Converter Interrupt Vectors

Cause of interrupt	Vector number	Vector address
Out of range	10 (0xa)	TTBR + 0x28
End of conversion	11 (0xb)	TTBR + 0x2c

VIII.1.7 Details of Control Registers

Table VIII.1.7.1 List of A/D Converter Registers

Address	Register name		Function
0x5520	AD_CLKCTL	A/D Clock Control Register	Controls A/D converter clock.
0x5540	AD_DAT	A/D Conversion Result Register	A/D converted data
0x5542	AD_TRIG_CH	A/D Trigger/Channel Select Register	Sets start/end channels and conversion mode.
0x5544	AD_CTL	A/D Control/Status Register	Controls A/D converter and indicates conversion status.
0x5546	AD_CH_STAT	A/D Channel Status Flag Register	Indicates overwrite error and conversion complete status.
0x5548	AD_CH0_BUF	A/D CH.0 Conversion Result Buffer Register	A/D CH.0 converted data
0x554a	AD_CH1_BUF	A/D CH.1 Conversion Result Buffer Register	A/D CH.1 converted data
0x554c	AD_CH2_BUF	A/D CH.2 Conversion Result Buffer Register	A/D CH.2 converted data
0x554e	AD_CH3_BUF	A/D CH.3 Conversion Result Buffer Register	A/D CH.3 converted data
0x5550	AD_CH4_BUF	A/D CH.4 Conversion Result Buffer Register	A/D CH.4 converted data
0x5552	AD_CH5_BUF	A/D CH.5 Conversion Result Buffer Register	A/D CH.5 converted data
0x5554	AD_CH6_BUF	A/D CH.6 Conversion Result Buffer Register	A/D CH.6 converted data
0x5556	AD_CH7_BUF	A/D CH.7 Conversion Result Buffer Register	A/D CH.7 converted data
0x5558	AD_UPPER	A/D Upper Limit Value Register	Specifies A/D conversion upper limit value.
0x555a	AD_LOWER	A/D Lower Limit Value Register	Specifies A/D conversion lower limit value.
0x555c	AD_INTMASK	A/D Conversion Complete Interrupt Mask Register	Masks A/D conversion complete interrupt.
0x555e	AD_ADVMODE	A/D Converter Mode Select/Internal Status Register	Selects A/D operating mode and indicates internal status and internal counter value.

The following describes each A/D converter register. These are all 16-bit registers.

Note: When setting the registers, be sure to write a 0, and not a 1, for all “reserved bits.”

0x5520: A/D Clock Control Register (AD_CLKCTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Clock Control Register (AD_CLKCTL)	0x5520 (16 bits)	D15–4	–	reserved	–	–	–	0 when being read.
		D3	PSONAD	A/D converter clock control	1 On / 0 Off	0	R/W	
		D2–0	PSAD[2:0]	A/D converter clock division ratio selection	PSAD[2:0] / A/D clock: 0x7 PCLK•1/256; 0x6 PCLK•1/128; 0x5 PCLK•1/64; 0x4 PCLK•1/32; 0x3 PCLK•1/16; 0x2 PCLK•1/8; 0x1 PCLK•1/4; 0x0 PCLK•1/2	0x0	R/W	

D[15:4] Reserved

D3 PSONAD: A/D Converter Clock Control Bit

Controls the A/D conversion clock supply to the A/D converter.

1 (R/W): On

0 (R/W): Off (default)

D[2:0] PSAD[2:0]: A/D Converter Clock Division Ratio Selection Bits

Selects a division ratio to generate the A/D converter clock.

Table VIII.1.7.2 Selecting Division Ratio

PSAD[2:0]	A/D clock
0x7	PCLK•1/256
0x6	PCLK•1/128
0x5	PCLK•1/64
0x4	PCLK•1/32
0x3	PCLK•1/16
0x2	PCLK•1/8
0x1	PCLK•1/4
0x0	PCLK•1/2

(Default: 0x0)

Notes: • The recommended A/D clock frequency is a maximum of 2 MHz and a minimum of 16 kHz.

• Do not start an A/D conversion when the clock output from the prescaler is turned off, and do not turn off the prescaler's clock output when an A/D conversion is underway. This could cause the A/D converter to operate erratically.

0x5540: A/D Conversion Result Register (AD_DAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Conversion Result Register (AD_DAT)	**0x5540** (16 bits)	D15–10	–	reserved	–	–	–	0 when being read.
		D9–0	**ADD[9:0]**	A/D converted data ADD9 = MSB ADD0 = LSB	0x0 to 0x3ff	0x0	R	

D[15:10] Reserved

D[9:0] ADD[9:0]: A/D Converted Data Bits

Stores the results of A/D conversion. (Default: 0x0)
The LSB is stored in ADD0, and the MSB is stored in ADD9.
This is a read-only register, so writing to this register is ignored.

0x5542: A/D Trigger/Channel Select Register (AD_TRIG_CH)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Trigger/ Channel Select Register (AD_TRIG_CH)	0x5542 (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13–11	CE[2:0]	A/D converter end channel selection	0 to 7	0x0	R/W	
		D10–8	CS[2:0]	A/D converter start channel selection	0 to 7	0x0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5	MS	A/D conversion mode selection	1 Continuous 0 Normal	0	R/W	
		D4–3	TS[1:0]	A/D conversion trigger selection	TS[1:0] Trigger: 0x3 #ADTRG pin; 0x2 PT8 CH.0; 0x1 MFT; 0x0 Software	0x0	R/W	
		D2–0	CH[2:0]	A/D conversion channel status	0 to 7	0x0	R	

D[15:14] Reserved

D[13:11] CE[2:0]: A/D Converter End Channel Selection Bits

Sets the conversion end channel by selecting a channel number from 0 to 7. (Default: 0x0 = AIN0)
Analog inputs can be A/D-converted successively from the channel set using CS[2:0] (D[10:8]) to the channel set using these bits in one operation. If only one channel is to be A/D converted, set the same channel number in both CS[2:0] and CE[2:0].

D[10:8] CS[2:0]: A/D Converter Start Channel Selection Bits

Sets the conversion start channel by selecting a channel number from 0 to 7. (Default: 0x0 = AIN0)
Analog inputs can be A/D-converted successively from the channel set using these bits to the channel set using CE[2:0] (D[13:11]) in one operation. If only one channel is to be A/D converted, set the same channel number in both CS[2:0] and CE[2:0].

D[7:6] Reserved

D5 MS: A/D Conversion Mode Selection Bit

Selects an A/D conversion mode.
1 (R/W): Continuous mode
0 (R/W): Normal mode (default)

The A/D converter is set for the continuous mode by writing 1 to MS. In this mode, A/D conversions in the range of the channels selected using CS[2:0] (D[10:8]) and CE[2:0] (D[13:11]) are executed continuously until stopped in the software.
When MS = 0, the A/D converter operates in the normal mode. In this mode, A/D conversion is completed after all inputs in the range of the channels selected by CS[2:0] and CE[2:0] are converted in one operation.

D[4:3] TS[1:0]: A/D Conversion Trigger Selection Bits

Selects a trigger to start A/D conversion.

Table VIII.1.7.3 Trigger Selection

TS[1:0]	Trigger source
0x3	External trigger (#ADTRG pin)
0x2	8-bit programmable timer (PT8) CH.0
0x1	16-bit multi-function timer (MFT)
0x0	Software trigger

(Default: 0x0)

When an external trigger is used, the #ADTRG pin must be set in advance using the Port Function Select Register. A/D conversion is started when a low level of the #ADTRG signal is detected.
When the 8-bit programmable timer CH.0 is used, since its underflow signal serves as a trigger, set the cycle and other parameters for the timer.
When the 16-bit multi-function timer is used, since its period-match signal serves as a trigger, set the cycle and other parameters for the timer.

D[2:0] CH[2:0]: A/D Conversion Channel Status Bits

Indicates the channel number (0 to 7) currently being A/D-converted. (Default: 0x0 = AIN0)
When A/D conversion is performed in multiple channels, read this bit to identify the channel in which conversion is underway.

0x5544: A/D Control/Status Register (AD_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Control/ Status Register (AD_CTL)	0x5544 (16 bits)	D15	**ADCMPE**	Upper/lower-limit comparison enable	1 Enable 0 Disable	0	R/W	Can be used when ADCADV = 1.
		D14–12	**ADCMP[2:0]**	Upper/lower-limit comparison channel selection	0 to 7	0x0	R/W	
		D11	**ADUPRST**	Upper-limit comparison status	1 Out of range 0 Within range	0	R	
		D10	**ADLWRST**	Lower-limit comparison status	1 Out of range 0 Within range	0	R	
		D9–8	**ST[1:0]**	Input signal sampling time setup	ST[1:0] Sampling time 0x3 9 clocks 0x2 7 clocks 0x1 5 clocks 0x0 3 clocks	0x3	R/W	Use with 9 clocks.
		D7	–	reserved	–	–	–	0 when being read.
		D6	**INTMODE**	Interrupt signal mode	1 Complete only 0 OR	0	R/W	Can be used when ADCADV = 1.
		D5	**CMPINTEN**	Out-of-range int. enable	1 Enable 0 Disable	0	R/W	
		D4	**CNVINTEN**	Conversion-complete int. enable	1 Enable 0 Disable	1	R/W	Can be changed when ADCADV = 1.
		D3	**ADF**	Conversion-complete flag	1 Completed 0 Run/ Standby	0	R	Reset when ADD is read.
		D2	**ADE**	A/D enable	1 Enable 0 Disable	0	R/W	
		D1	**ADST**	A/D conversion control/status	1 Start/Run 0 Stop	0	R/W	
		D0	**OWE**	Overwrite error flag	1 Error 0 Normal	0	R/W	Reset by writing 0.

D15 ADCMPE: Upper/Lower-Limit Comparison Enable Bit (for advanced mode)

Enables/disables comparison between converted data and upper-/lower-limit values.
1 (R/W): Enabled
0 (R/W): Disabled (default)

ADCMPE selects whether the converted data is compared with the upper-/lower-limit values after A/D conversion of the channel specified using ADCMP[2:0] (D[14:12]). Set ADCMPE to 1 when using the comparison function or set to 0 when not used.

D[14:12] ADCMP[2:0]: Upper/Lower-Limit Comparison Channel Selection Bits (for advanced mode)

Set the channel number (0–7) to compare its converted data with the upper-/ lower-limit values. (Default: 0x0 = AIN0)
These bits do not affect the CH.0 out-of-range signal output for the MFT.

D11 ADUPRST: Upper-Limit Comparison Status Bit (for advanced mode)

Indicates the results of comparison between the A/D converted data and the upper-limit value.
1 (R): Exceeded the upper limit
0 (R): Within the range (default)

When the upper-/lower-limit comparison function is enabled (ADCMPE (D15) = 1), the converted data is compared with the upper-/lower-limit values after A/D conversion of the channel specified using ADCMP[2:0] (D[14:12]) has completed. If the converted data exceeds the upper-limit value set in ADUPR[9:0] (D[9:0]/AD_UPPER register), ADUPRST is set to 1. If the converted data is equal to or less than the upper-limit value, ADUPRST is set to 0. An interrupt occurs when ADUPRST is set to 1 if the out-of-range interrupt is enabled.

D10 ADLWRST: Lower-Limit Comparison Status Bit (for advanced mode)

Indicates the results of comparison between the A/D converted data and the lower-limit value.
1 (R): Under the lower limit
0 (R): Within the range (default)

When the upper-/lower-limit comparison function is enabled (ADCMPE (D15) = 1), the converted data is compared with the upper-/lower-limit values after A/D conversion of the channel specified using ADCMP[2:0] (D[14:12]) has completed. If the converted data is less than the lower-limit value set in ADLWR[9:0] (D[9:0]/AD_LOWER register), ADLWRST is set to 1. If the converted data is equal to or more than the lower-limit value, ADLWRST is set to 0. An interrupt occurs when ADLWRST is set to 1 if the out-of-range interrupt is enabled.

D[9:8] ST[1:0]: Input Signal Sampling Time Setup Bits

Sets the analog input sampling time.

Table VIII.1.7.4 Sampling Time

ST[1:0]	Sampling time
0x3	9-clock period
0x2	7-clock period
0x1	5-clock period
0x0	3-clock period

(Default: 0x3)

The A/D converter conversion clock is used for counting.
To maintain the conversion accuracy, use ST as set by default (9-clock period).
The conversion time is fixed at 11-clock period.

D7 Reserved

D6 INTMODE: Interrupt Signal Mode Bit (for advanced mode)

Configures the conversion-complete interrupt signal delivered to the ITC.
1 (R/W): Conversion-complete signal only
0 (R/W): OR between conversion-complete and out-of-range signals (default)

INTMODE selects whether the conversion-complete interrupt signal line connected to the ITC is used to send the conversion-complete signal only or used to send the signal of which the conversion-complete and out-of-range signal are ORed.
Set INTMODE to 1 when handling the out-of-range interrupt as another interrupt. When using the out-of-range interrupt, set CMPINTEN (D5) to 1.

D5 CMPINTEN: Out-of-Range Interrupt Enable Bit (for advanced mode)

Enables/disables the out-of-range interrupt.
1 (R/W): Enabled
0 (R/W): Disabled (default)

When CMPINTEN is set to 1, upper and lower-limit comparison results become a cause of interrupt. When it is set to 0, an out-of-range interrupt is not generated.

D4 CNVINTEN: Conversion-Complete Interrupt Enable Bit

Enables/disables the conversion-complete interrupt.
1 (R/W): Enabled (default)
0 (R/W): Disabled

When CNVINTEN is set to 1, completion of an A/D conversion becomes a cause of interrupt. When it is set to 0, a conversion-complete interrupt is not generated.

Note: CNVINTEN is effective in standard mode. Set CNVINTEN to 1 to enable the A/D converter interrupt in standard and advanced modes. However, CNVINTEN can only be changed in advanced mode (ADCADV = 1).

D3 ADF: Conversion-Complete Flag Bit

Indicates that A/D conversion has been completed.
1 (R): Conversion completed
0 (R): Being converted or standing by (default)

This flag is set to 1 when A/D conversion is completed, and the converted data is stored in the data register and is reset to 0 when the converted data is read out. When A/D conversion is performed in multiple channels, if the next A/D conversion is completed while ADF = 1 (before the converted data is read out), the data register is overwritten with the new conversion results, causing an overrun error to occur. Therefore, ADF must be reset by reading out the converted data before the next A/D conversion is completed.

D2 ADE: A/D Enable Bit

Enables the A/D converter (ready for conversion).
1 (R/W): Enabled
0 (R/W): Disabled (default)

When ADE is set to 1, the A/D converter is enabled, meaning it is ready to start A/D conversion (i.e., ready to accept a trigger). When ADE = 0, the A/D converter is disabled, meaning it is unable to accept a trigger.
Before setting the conversion mode, start/end channels, etc. for the A/D converter, be sure to reset ADE to 0. This helps to prevent the A/D converter from operating erratically.

D1 ADST: A/D Conversion Control/Status Bit

Controls A/D conversion.
1 (R/W): Software trigger
0 (R/W): A/D conversion is stopped (default)

If A/D conversion is to be started by a software trigger, set ADST to 1. If any other trigger is used, ADST is automatically set to 1 by the hardware.
ADST remains set while A/D conversion is underway.
In normal mode, upon completion of A/D conversion in selected channels, ADST is reset to 0 and the A/D conversion circuit is turned off. To stop A/D conversion during operation in continuous mode, reset ADST by writing 0.
When ADE (D2) = 0 (A/D conversion disabled), ADST is fixed to 0, with no trigger accepted.

D0 OWE: Overwrite Error Flag Bit

Indicates that the converted data has been overwritten.
1 (R): Overwritten
0 (R): Normal (default)
1 (W): Has no effect
0 (W): Flag is reset

During A/D conversion in multiple channels, if the conversion results for the next channel are written to the converted-data register (overwritten) before the converted data is read out to reset the conversion-complete flag ADF (D3) that has been set through conversion of the preceding channel, OWE is set to 1. When ADF (D3) is reset, because this means that the converted data has been read out, OWE is not set. Once OWE is set to 1, it remains set until it is reset by writing 0 in the software.

0x5546: A/D Channel Status Flag Register (AD_CH_STAT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
A/D Channel Status Flag Register (AD_CH_STAT)	0x5546 (16 bits)	D15	OWE7	CH.7 overwrite error flag	1	Error	0	Normal	0	R/W	Can be used when ADCADV = 1. Reset by writing 0.
		D14	OWE6	CH.6 overwrite error flag					0	R/W	
		D13	OWE5	CH.5 overwrite error flag					0	R/W	
		D12	OWE4	CH.4 overwrite error flag					0	R/W	
		D11	OWE3	CH.3 overwrite error flag					0	R/W	
		D10	OWE2	CH.2 overwrite error flag					0	R/W	
		D9	OWE1	CH.1 overwrite error flag					0	R/W	
		D8	OWE0	CH.0 overwrite error flag					0	R/W	
		D7	ADF7	CH.7 conversion-complete flag	1	Completed	0	Run/ Standby	0	R	Can be used when ADCADV = 1. Reset when ADxBUF [9:0] is read.
		D6	ADF6	CH.6 conversion-complete flag					0	R	
		D5	ADF5	CH.5 conversion-complete flag					0	R	
		D4	ADF4	CH.4 conversion-complete flag					0	R	
		D3	ADF3	CH.3 conversion-complete flag					0	R	
		D2	ADF2	CH.2 conversion-complete flag					0	R	
		D1	ADF1	CH.1 conversion-complete flag					0	R	
		D0	ADF0	CH.0 conversion-complete flag					0	R	

Note: The letter '*x*' in bit names, etc., denotes a channel number from 0 to 7.

D[15:8] OWE[7:0]: CH.*x* Overwrite Error Flag Bits (for advanced mode)

These bits indicate that the conversion result buffer for each channel has been overwritten.

1 (R): Overwritten
0 (R): Normal (default)
1 (W): Has no effect
0 (W): Flag is reset

During A/D conversion in continuous mode, if the new conversion results in the same channel are written to the conversion result buffer (overwritten) before the converted data is read out to reset the ADF*x* conversion-complete flag that has been set through the previous conversion, OWE*x* is set to 1. When ADF*x* is reset, because this means that the converted data has been read out, OWE*x* is not set.
Once OWE*x* is set to 1, it remains set until it is reset by writing 0 in the software.

D[7:0] ADF[7:0]: CH.*x* Conversion-Complete Flag Bits (for advanced mode)

These bits indicate that A/D conversion in each channel has been completed.

1 (R): Conversion completed
0 (R): Being converted or standing by (default)

This flag is set to 1 when A/D conversion of the corresponding channel is completed, and the converted data is stored in the conversion result buffer and is reset to 0 when the conversion result buffer is read out. When A/D conversion is performed in continuous mode, if the next A/D conversion of the same channel is completed while ADF*x* = 1 (before the conversion result buffer is read out), the buffer is overwritten with the new conversion results, causing an overrun error to occur. Therefore, ADF*x* must be reset by reading out the converted data before the next A/D conversion is completed.

0x5548–0x5556: A/D CH.*x* Conversion Result Buffer Registers (AD_CH*x*_BUF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D CH.*x* Conversion Result Buffer Register (AD_CH*x*_BUF)	**0x5548** | **0x5556** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**AD*x*BUF [9:0]**	A/D CH.*x* converted data AD*x*BUF9 = MSB AD*x*BUF0 = LSB	0x0 to 0x3ff	0x0	R	Can be used when ADCADV = 1.

Note: The letter '*x*' in bit names, etc., denotes a channel number from 0 to 7.

0x5548 A/D CH.0 Conversion Result Buffer Register (AD_CH0_BUF)
0x554a A/D CH.1 Conversion Result Buffer Register (AD_CH1_BUF)
0x554c A/D CH.2 Conversion Result Buffer Register (AD_CH2_BUF)
0x554e A/D CH.3 Conversion Result Buffer Register (AD_CH3_BUF)
0x5550 A/D CH.4 Conversion Result Buffer Register (AD_CH4_BUF)
0x5552 A/D CH.5 Conversion Result Buffer Register (AD_CH5_BUF)
0x5554 A/D CH.6 Conversion Result Buffer Register (AD_CH6_BUF)
0x5556 A/D CH.7 Conversion Result Buffer Register (AD_CH7_BUF)

D[15:10] Reserved

D[9:0] AD*x*BUF[9:0]: A/D CH.*x* Converted Data Bits (for advanced mode)

The conversion results in each channel are stored. (Default: 0x0)
This is a read-only register, so writing to this register is ignored.

0x5558: A/D Upper Limit Value Register (AD_UPPER)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Upper Limit Value Register (AD_UPPER)	0x5558 (16 bits)	D15–10	–	reserved	–	–	–	0 when being read.
		D9–0	ADUPR[9:0]	A/D conversion upper limit value ADUPR9 = MSB ADUPR0 = LSB	0x0 to 0x3ff	0x0	R/W	Can be used when ADCADV = 1.

D[15:10] Reserved

D[9:0] ADUPR[9:0]: A/D Conversion Upper Limit Value Bits (for advanced mode)

Set the upper-limit value to be compared with the A/D conversion results. (Default: 0x0)

The value set in this register is used for the range check of the A/D conversion results in the channel specified with ADCMP[2:0] (D[14:12]/AD_CTL register). If the converted data exceeds the set value, an interrupt can be generated.

0x555a: A/D Lower Limit Value Register (AD_LOWER)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Lower Limit Value Register (AD_LOWER)	**0x555a** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**ADLWR [9:0]**	A/D conversion lower limit value ADLWR9 = MSB ADLWR0 = LSB	0x0 to 0x3ff	0x0	R/W	Can be used when ADCADV = 1.

D[15:10] Reserved

D[9:0] ADLWR[9:0]: A/D Conversion Lower Limit Value Bits (for advanced mode)

Set the lower-limit value to be compared with the A/D conversion results. (Default: 0x0)

The value set in this register is used for the range check of the A/D conversion results in the channel specified with ADCMP[2:0] (D[14:12]/AD_CTL register). If the converted data is less than the set value, an interrupt can be generated.

0x555c: A/D Conversion Complete Interrupt Mask Register (AD_INTMASK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Conversion Complete Interrupt Mask Register (AD_INTMASK)	0x555c (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7	**INTMASK7**	CH.7 conversion-complete int. mask	1 Interrupt enabled / 0 Interrupt mask	1	R/W	Can be used when ADCADV = 1.
		D6	**INTMASK6**	CH.6 conversion-complete int. mask		1	R/W	
		D5	**INTMASK5**	CH.5 conversion-complete int. mask		1	R/W	
		D4	**INTMASK4**	CH.4 conversion-complete int. mask		1	R/W	
		D3	**INTMASK3**	CH.3 conversion-complete int. mask		1	R/W	
		D2	**INTMASK2**	CH.2 conversion-complete int. mask		1	R/W	
		D1	**INTMASK1**	CH.1 conversion-complete int. mask		1	R/W	
		D0	**INTMASK0**	CH.0 conversion-complete int. mask		1	R/W	

Note: The letter '*x*' in bit names, etc., denotes a channel number from 0 to 7.

D[15:8] Reserved

D[7:0] INTMASK[7:0]: CH.*x* Conversion-Complete Interrupt Mask Bits (for advanced mode)

These bits mask the A/D conversion-complete interrupt for each channel individually.
1 (R/W): Interrupt is enabled (default)
0 (R/W): Interrupt is masked

When INTMASK*x* is set to 0, the conversion-completed interrupt request of the CH.*x* is masked and the interrupt flag in the ITC will not be set to 1 even if A/D conversion is completed. When INTMASK*x* is 1, the A/D converter can generate an interrupt upon completion of A/D conversion in CH.*x*.

Note: INTMASK[7:0] must be set to 1 to generate A/D converter interrupts in standard and advanced modes.

0x555e: A/D Converter Mode Select/Internal Status Register (AD_ADVMODE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Converter Mode Select/ Internal Status Register (AD_ADVMODE)	0x555e (16 bits)	D15–9	–	reserved	–	–	–	0 when being read.
		D8	ADCADV	Standard/advanced mode selection	1 Advanced 0 Standard	0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5–4	ISTATE[1:0]	Internal status	ISTATE[1:0] Status 0x3 Converting 0x2 reserved 0x1 Sampling 0x0 Idle	0x0	R	
		D3–0	ICOUNTER [3:0]	Internal counter value	0 to 15	0x0	R	

D[15:9] Reserved

D8 ADCADV: Standard/Advanced Mode Selection Bit

Selects the A/D converter operating mode.
1 (R/W): Advanced mode
0 (R/W): Standard mode (default)

When ADCADV is set to 1, the A/D converter is set in the advanced mode, and the registers/bits for the extended function can be used.
When ADCADV is set to 0, only the standard A/D converter functions can be used. In this mode, the extended registers/bits for advanced mode become read only and writing operation is disabled.

D[7:6] Reserved

D[5:4] ISTATE[1:0]: Internal Status Bits

Indicates the A/D converter internal status.

Table VIII.1.7.5 Internal Status

ISTATE[1:0]	Status
0x3	Converting
0x2	Reserved
0x1	Sampling
0x0	Idle

(Default: 0x0)

D[3:0] ICOUNTER[3:0]: Internal Counter Value Bits

Indicates the internal counter value. (Default: 0x0)

VIII.1.8 Precautions

- Before setting the conversion mode, start/end channels, etc. for the A/D converter, be sure to disable ADE (D2/AD_CTL register). A change in settings while the A/D converter is enabled could cause it to operate erratically.

  * **ADE**: A/D Enable Bit in the A/D Control/Status (AD_CTL) Register (D2/0x5544)

- In consideration of the conversion accuracy, we recommend that the A/D conversion clock be min. 16 kHz to max. 2 MHz.
- Do not start an A/D conversion when the clock supplied from the prescaler to the A/D converter is turned off, and do not turn off the prescaler's clock output when an A/D conversion is underway, as doing so could cause the A/D converter to operate erratically.
- When the MFT period-match signal is used as a trigger factor, the division ratio of the prescaler in the MFT module must not be set to PCLK/1.
- When using an external trigger to start A/D conversion, the low period of the trigger signal to be applied to the #ADTRG pin must be two or more CPU operating clock cycles. Furthermore, return the #ADTRG input level to high within 20 cycles of the A/D input clock set. Otherwise, it will be detected as the trigger for the next A/D conversion.
- When the A/D converter is set to enabled state, a current flows between AV_{DD} and V_{SS}, and power is consumed, even when A/D operations are not performed. Therefore, when the A/D converter is not used, it must be set to the disabled state (default 0 setting of ADE (D2/AD_CTL register)).
  The A/D converter must always be enabled if a timer output or an external input is used as the trigger but it increases current consumption. To reduce current consumption, an A/D conversion control procedure as shown below is recommended so that the A/D converter will be enabled as short as possible using the software trigger.

  (1) Place the S1C17501 into HALT mode with the OSC1 clock set as the CPU clock if A/D conversion control is not necessary.
  (2) Generate an interrupt according to the sampling frequency to cancel HALT mode.
  (3) Run the CPU with the OSC3 clock.
  (4) Enable the A/D converter.
  (5) Start an A/D conversion using the software trigger.
  (6) Read the conversion results.
  (7) Disable the A/D converter.
  (8) Return to Step (1) after the necessary processing has been finished.

IX S1C17501 USB MODULE

IX.1 USB Function Controller (USB)

IX.1.1 Outline of the USB Function Controller

The S1C17501 has a built-in USB function controller that supports the Full-Speed mode defined in the USB2.0 Specification. The features are shown below.

- Supports transfer at FS (12 Mbps).
- Supports control, bulk, isochronous and interrupt transfers.
- Supports four general-purpose endpoints and endpoint 0.
- Incorporate 1KB programmable FIFO for endpoints.
- Supports asynchronous procedures.
  - Supports a slave configuration.
  - Can be used with a bus width of 8 bits.
- Inputs 48 MHz clock.
- Supports snooze mode.

Figure IX.1.1.1 shows the block diagram of the USB function controller.

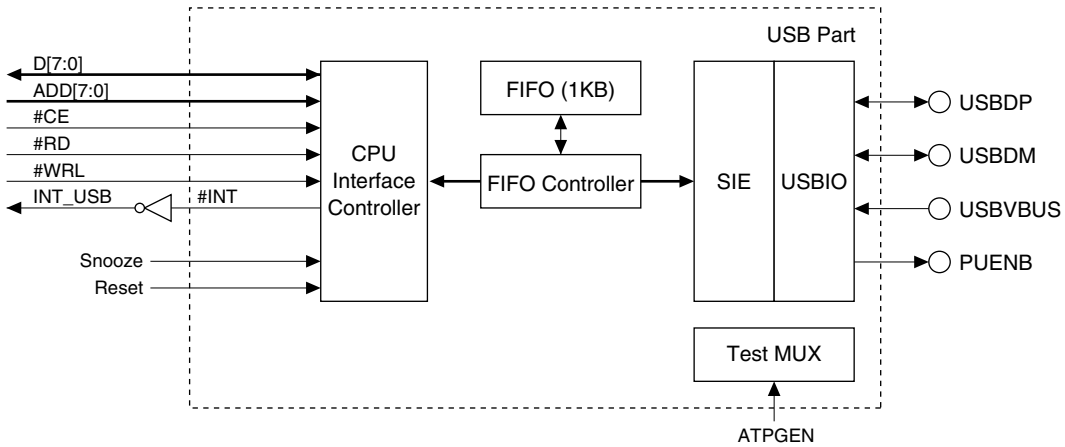


Figure IX.1.1.1 USB Function Controller Block Diagram

Serial Interface Engine (SIE)

The SIE manages transactions and generates packets. It also controls bus events such as suspend, resume and reset operations.

FIFO

This is a 1KB buffer for endpoints.

FIFO Controller

This controller performs FIFO SRAM address management (user-programmable), timing generation, arbitration and more.

Port Interface Controller

This controller performs asynchronous handshakes.

CPU Interface Controller

This controller controls timings of the CPU interface and enables register access.

Test Mux

Switches the operational mode (test mode) using the Input signal.

IX.1.2 Pins for the USB Interface

Table IX.1.2.1 list the pins used for the USB interface.

Table IX.1.2.1 USB Interface Pins

Pin name	I/O	Function
USBDP	I/O	USB D+ pin
USBDM	I/O	USB D- pin
USBVBUS	I	USB VBUS pin. Allows input of 5 V.
PUENB	O	USB DP pull-up enable output pin (P channel open drain output)

Note: The USBDP, USBDM, and PUENB pins must be left open when the USB function controller is not used.
The USBVBUS pin must be connected to GND (V_{SS}) when the USB function controller is not used.

IX.1.3 System Settings

IX.1.3.1 Controlling the USB Clocks

The CMU provides the clock paths with the control bit shown below for the USB module. The clock supply turns on when the control bit is set to 1 and it turns off when the control bit is set to 0.

(1) USB clock (USB_CLK)

This clock (OSC3 = 48 MHz) is used for the USB interface module. USB_CLK_EN (D4/CMU_GATEDCLK0 register) is used for clock supply control (default: off).

* **USB_CLK_EN**: USB IP 48 MHz Clock Control Bit in the Gated Clock Control 0 (CMU_GATEDCLK0) Register (D4/0x4906)

(2) Control register clock (USB_SAPB_CLK)

This clock is used to control the USB registers. This clock is required for accessing the USB registers and it can be stopped when not in use. USB_SAPB_CLK_EN (D5/CMU_GATEDCLK0 register) is used for clock supply control (default: off).

* **USB_SAPB_CLK_EN**: USB SAPB I/F Clock Control Bit in the Gated Clock Control 0 (CMU_GATEDCLK0) Register (D5/0x4906)

IX.1.3.2 USB Wait Control

The USB Wait Control Register (0x4909) contains the control bits USBWT[2:0] (D[2:0]) used to set the number of wait cycles to be inserted when accessing the USB registers.

* **USBWT[2:0]**: USB Register Access Wait Control Bits in the USB Wait Control (CMU_USBWT) Register (D[2:0]/0x4909)

Table IX.1.3.2.1 Number of Wait Cycles during USB Access

USBWT[2:0]	Number of wait cycles (in units of system clock cycles)	System clock frequency
0x7	7 cycles	Reserved
0x6	6 cycles	48 MHz or less
0x5	5 cycles	45 MHz or less
0x4	4 cycles	36 MHz or less
0x3	3 cycles	24 MHz or less
0x2	2 cycles	16 MHz or less
0x1	1 cycle	8 MHz or less
0x0	0 cycles	8 MHz or less

(Default: 0x7 = 7 cycles)

The number of wait cycles should be set according to the system clock frequency.

Also the USB Wait Control Register (0x4909) contains the USBSNZ bit (D5) that controls Snooze mode for the USB function controller. Setting USBSNZ to 1 enables Snooze mode.

* **USBSNZ**: USB Snooze Control Bit in the USB Wait Control (CMU_USBWT) Register (D5/0x4909)

Set USBSNZ to 0 during normal operation.

IX.1.3.3 Setting the ITC

The USB module has one interrupt signal to be output to the ITC and it is shared with all causes of USB interrupts. To determine the cause of interrupt that has occurred, read the interrupt status registers in the USB module.

ITC registers for USB interrupt

When a cause of USB interrupt that has been enabled occurs, the USB module asserts the interrupt signal sent to the ITC. To generate a USB interrupt, set the interrupt level and enable the interrupt using the ITC registers. The following shows the control bits for the USB interrupt in the ITC.

Interrupt flag in the ITC

* **AIFT11**: USB Interrupt Flag Bit in the Additional Interrupt Flag (ITC_AIFLG) Register (D11/0x42e0)

Interrupt enable bit in the ITC

* **AIEN11**: USB Interrupt Enable Bit in the Additional Interrupt Enable (ITC_AEN) Register (D11/0x42e2)

Interrupt level setup bits in the ITC

* **AILV11[2:0]**: USB Interrupt Level Bits in the Additional Interrupt Level Setup (ITC_AILV5) Register 5 (D[10:8]/0x42f0)

The interrupt signal sent from the USB module sets AIFT11 to 1. If AIEN11 has been set to 1, the ITC sends an interrupt request to the S1C17 Core. To disable the USB interrupt, set AIEN11 to 0. AIFT11 is always set to 1 by the interrupt signal sent from the USB module, regardless of how AIEN11 is set (even when set to 0).
AILV11[2:0] sets the interrupt level (0 to 7) of the USB interrupt.

An interrupt request to the S1C17 Core is accepted only when all the conditions described below are met.

- The interrupt enable bit is set to 1.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the S1C17 Core is set to 1.
- The USB interrupt has a higher interrupt level than the value that is set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

For details on these interrupt control registers, as well as the device operation when an interrupt has occurred, see Section IV.1, “Interrupt Controller (ITC).”

Interrupt vector

The following shows the vector number and vector address for the USB interrupt:

Vector number: 27 (0x1b)
Vector address: TTBR + 0x6c

IX.1.4 Functional Description

This section describes the functionality of the USB function controller.
In the subsequent sections, the register names follow the notational convention below:

* When a register for one address is referred to:
  Register name + register.
  Example: “MainInt register”

* When a discrete bit is referred to:
  Register name. bit name + bit, or bit name + bit.
  Example: “MainIntStat.RcvEP0SETUP bit”, or “ForceNAK bit of the EP0ControlOUT register”

* When a register present for a specific end-point is referred to:
  EP*x*{*x*=0,a,b,c,d}register name + register, EP*x*{*x*=a,b,c,d}register name + register, and so forth.
  Example: “EP*x*{*x*=0,a,b,c,d}IntStat register”, “EP*x*{*x*=a,b,c,d}Control register”

IX.1.4.1 USB Control

End points

This macro has an endpoint (EP0) for control transfer and four general purpose-endpoints (EPa, EPb, EPc, EPd). Endpoints, EPa, EPb, EPc and EPd can be used as endpoints for bulk- or interrupt- or isochronous-type transfer, respectively. There is no difference between bulk and interrupt transfers in terns of hardware.
The macro hardware provides endpoints and manages transactions. However, it does not provide a management function in the interface defined for the USB (hereinafter referred to as USB-defined interface). The USB-defined interface should be implemented in your firmware. According to the device-specific descriptor definition, set endpoints as required and configure the USB-defined interface using an appropriate endpoint combination.
Besides variable control items and statuses that are controlled for each transfer operation, each endpoint has fixed basic setting items determined by the USB-defined interface. The basic setting items should be set up when initializing the chip or when the USB-defined interface is switched in response to a SetInterface() request. Table IX.1.4.1.1 lists the basic setting items for the EP0 endpoint (default control pipe).
The EP0 endpoint shares the register set and FIFO region between the In and OUT directions. For data and status stages at the EP0 endpoint, set the data transaction direction in your firmware before executing such stages.

Table IX.1.4.1.1 Basic Setting Items for Endpoint EP0

Item	Register/bit	Description
Max. packet size	EP0MaxSize	Sets the maximum packet size to 8, 16, 32 or 64 for the FS-mode operation. The EP0 endpoint is assigned a region of the size that is set in the EP0MaxSize register, starting with FIFO address 0.

Table IX.1.4.1.2 lists the basic setting items for the general-purpose endpoints (EPa, EPb, EPc, and EPd). The EPa, EPb, EPc, and EPd endpoints allow optional settings for the transaction directions and the endpoint numbers, which allows up to four discrete endpoints to be used. Set up and/or enable these endpoints as appropriate according to the definitions for the USB-defined interface.

Table IX.1.4.1.2 Setting Items for Endpoints EPa, EPb, EPc and EPd

Item	Register/bit	Description
Transaction direction	EP*x*{*x*=a,b,c,d}Config.INxOUT	Sets the transfer direction for each endpoint.
Max. packet size	EP*x*{*x*=a,b,c,d}MaxSize_H, EP*x*{*x*=a,b,c,d}MaxSize_L	Sets the maximum packet size of each endpoint to any desired value between 1 and 1024 bytes. For endpoints that perform bulk transfers, set them to 8, 16, 32 or 64 bytes in FS mode.
Endpoint number	EP*x*{*x*=a,b,c,d}Config.EndPointNumber	Sets each endpoint number to any desired value between 0x1 and 0xf.
Toggle mode	EP*x*{*x*=a,b,c,d}Config.ToggleMode	Sets a mode for a toggle sequence. Set it to 0 for an endpoint that performs bulk transfer. 0: Toggles only in successful transactions. 1: Toggles for every transaction.
Enable endpoint	EP*x*{*x*=a,b,c,d}Config.EnEndPoint	Enables each endpoint. Set it up when the USB-defined interface that uses the relevant endpoint is enabled.
FIFO region	EP*x*{*x*=a,b,c,d}StartAdrs_H, EP*x*{*x*=a,b,c,d}StartAdrs_L	Sets a region to be assigned to each endpoint using FIFO addresses. For a FIFO region, assign a region equivalent to the maximum packet size set for the relevant endpoint or greater. Note that the size of the FIFO region affects data transfer throughput. For details of FIFO region assignment, see the "FIFO Management" section.

Transaction

This macro hardware executes transactions while its interface provides the firmware with utilities for executing transactions. The interface to the firmware is implemented through control and status registers as well as the interrupt signal which is asserted depending on the status. For settings that enable asserting interruption according to the status, see the section on register description.

The macro issues a status to the firmware for each transaction. However, the firmware does not always have to control respective transactions. The macro references the FIFO when responding to a transaction and determines if data transfer is possible based on the number of data or vacancies to automatically handle the transaction.

For example, for an OUT endpoint, the firmware can smoothly and sequentially process OUT transactions by reading data from the FIFO region via either the Port interface (EPa, EPb, EPc, EPd) or the CPU interface (EP0, EPa, EPb, EPc, EPd) to create a space in the FIFO region. On the other hand, for an IN endpoint, the firmware can smoothly and sequentially process IN transactions by writing data in the FIFO region via either the Port interface (EPa, EPb, EPc, EPd) or the CPU interface (EP0, EPa, EPb, EPc, EPd) to create valid data.

Table IX.1.4.1.3 lists control items and statuses related to transaction control on the EP0 endpoint.

Table IX.1.4.1.3 Control Items and Statuses for Endpoint EP0

Item	Register/bit	Description
Transaction direction	EP0Control.INxOUT	Sets the transfer direction at the data and status stages.
Enable descriptor return	EP0Control.ReplyDescriptor	Activates automatic descriptor return.
Enable short packet transmission	EP0ControlIN.EnShortPkt	Enables transmission of short packets that are under the maximum packet size. This setting is cleared after the IN transaction that has transmitted a short packet is completed.
Toggle sequence bit	EP0ControlIN.ToggleStat, EP0ControlOUT.ToggleStat	Indicates the state of the toggle sequence bit. This setting is automatically initialized by the SETUP stage.
Set toggle	EP0ControlIN.ToggleSet, EP0ControlOUT.ToggleSet	Sets the toggle sequence bit.
Clear toggle	EP0ControlIN.ToggleClr, EP0ControlOUT.ToggleClr	Clears the toggle sequence bit.
Forced NAK response	EP0ControlIN.ForceNAK, EP0ControlOUT.ForceNAK	Returns a NAK response to IN or OUT transactions regardless of the number of data or vacancies in the FIFO region.
STALL response	EP0ControlIN.ForceSTALL, EP0ControlOUT.ForceSTALL	Returns a STALL response to IN or OUT transactions.
Set automatic ForceNAK	EP0ControlOUT.AutoForceNAK	Sets the EP0ControlOUT.ForceNAK bit whenever an OUT transaction is completed.
SETUP reception status	MainIntStat.RcvEP0SETUP	Indicates that a SETUP transaction is executed.
Transaction status	EP0IntStat.IN_TranACK, EP0IntStat.OUT_TranACK, EP0IntStat.IN_TranNAK, EP0IntStat.OUT_TranNAK, EP0IntStat.IN_TranErr, EP0IntStat.OUT_TranErr	Indicates the result of the transaction.

Table IX.1.4.1.4 lists control items and statuses related to transaction processing on the EPa, EPb, EPc, and EPd endpoints.

Table IX.1.4.1.4 Control Items and Statuses for Endpoints EPa, EPb, EPc, and EPd

Item	Register/bit	Description
Set automatic ForceNAK	EP*x*{*x*=a,b,c,d}Control.AutoForceNAK	Sets the endpoint's EP*x*{*x*=a,b,c,d}Control.ForceNAK bit whenever an OUT transaction is completed.
Enable short packet transmission	EP*x*{*x*=a,b,c,d}Control.EnShortPkt	Enables transmission of short packets that are under the maximum packet size for IN transactions. This setting is cleared after the IN transaction that has transmitted a short packet is completed.
Disable automatic ForceNAK setting upon short packet reception	EP*x*{*x*=a,b,c,d}Control. DisAF_NAK_Short	In OUT transactions, reception of a short packet automatically disables the function that sets the endpoint's EP*x*{*x*=a,b,c,d}Control.ForceNAK bit.
Toggle sequence bit	EP*x*{*x*=a,b,c,d}Control.ToggleStat	Indicates the state of the toggle sequence bit.
Set toggle	EP*x*{*x*=a,b,c,d}Control.ToggleSet	Sets the toggle sequence bit.
Clear toggle	EP*x*{*x*=a,b,c,d}Control.ToggleClr	Clears the toggle sequence bit.
Forced NAK response	EP*x*{*x*=a,b,c,d}Control.ForceNAK	Returns a NAK response to a transaction regardless of the number of data or vacancies in the FIFO region.
STALL response	EP*x*{*x*=a,b,c,d}Control.ForceSTALL	Returns a STALL response to the transaction.
Transaction status	EP*x*{*x*=a,b,c,d}IntStat.OUT_ShortACK, EP*x*{*x*=a,b,c,d}IntStat.IN_TranACK, EP*x*{*x*=a,b,c,d}IntStat.OUT_TranACK, EP*x*{*x*=a,b,c,d}IntStat.IN_TranNAK, EP*x*{*x*=a,b,c,d}IntStat.OUT_TranNAK, EP*x*{*x*=a,b,c,d}IntStat.IN_TranErr, EP*x*{*x*=a,b,c,d}IntStat.OUT_TranErr	Indicates the result of the transaction.

SETUP transaction

The SETUP transaction addressed to the EP0 endpoint of the macro's own node is automatically executed. (The USB function must be enabled for this to happen.)

When a SETUP transaction is issued, all the contents of the data packet (8 bytes) are stored in the registers EP0Setup_0 through EP0Setup_7, followed by an ACK response. Meanwhile, a RcvEP0SETUP status is issued to the firmware.

If an error occurs during a SETUP transaction, no response or status is issued.

When the SETUP transaction is completed, the ForceNAK bit of the EP0ControlIN and EP0ControlOUT registers are set and the ForceSTALL bit is cleared. The ToggleStat bit is also set. After the firmware completes setting the EP0 endpoint and becomes ready to proceed to the next stage, clear the ForceNAK bit of the relevant direction in the EP0ControlIN or EP0ControlOUT register.

Figure IX.1.4.1.1 illustrates how the SETUP transaction is executed.

(a) The host issues a SETUP token addressed to the EP0 endpoint of this node.
(b) Next, the host sends an 8-byte long data packet. The macro writes these data in the EP0Setup_0 through EP0Setup_7 registers.
(c) The macro automatically returns an ACK response. In addition, it sets registers to be automatically set up and issues a status to the firmware.

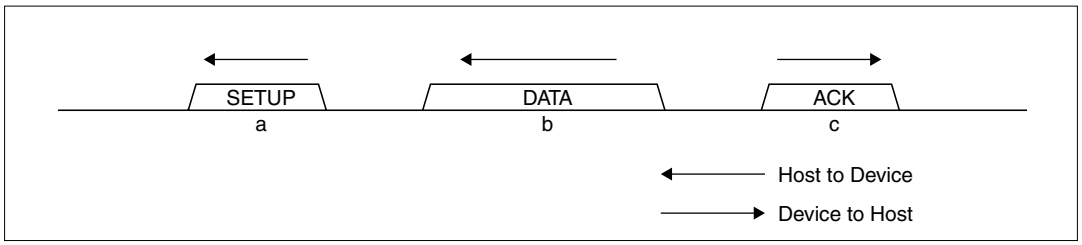


Figure IX.1.4.1.1 SETUP Transaction

OUT transaction

In OUT transactions, data reception is started regardless of the available space in the FIFO. Thus, this product provides satisfactory throughput by assigning a FIFO region about twice as large as the maximum packet size since it can read the FIFO data via the Port interface, for example, and receive data while creating an available space concurrently.

After all data are successfully received in an OUT transaction, the transaction is closed and an ACK response is returned. In addition, the firmware receives an OUT_TranACK status of the relevant endpoint (EP*x*{*x*=0,a,b,c,d}IntStat.OUT_TranACK bit). Furthermore, the FIFO is updated to acknowledge the data reception and to secure a space for the data.

In OUT transactions on the EPa, EPb, EPc, and EPd endpoints, reception of all short-packet data causes an OUT_ShortACK status (EP*x*{*x*=a,b,c,d}IntStat.OUT_ShortACK bit) to be issued, in addition to executing the above closing process. If the EP*x*{*x*=a,b,c,d}Control.DisAF_NAK_Short bit is cleared, the relevant endpoint's EP*x*{*x*=a,b,c,d}ForceNAK bit is set.

If a toggle miss-match has occurred in an OUT transaction, an ACK response is returned to the transaction but no status is issued. Accordingly, the FIFO is not updated.

In the event of an error in an OUT transaction, no response is returned to the transaction. And an OUT_TranErr status (EP*x*{*x*=0,a,b,c,d}IntStat.OUT_TranErr bit) is issued. Accordingly, the FIFO is not updated.

If not all data are received in an OUT transaction, a NAK response is returned to the transaction and the OUT_TranNAK status (EP*x*{*x*=a,b,c,d}IntStat.OUT_TranNAK bit) is issued. Accordingly, the FIFO is not updated.

Figure IX.1.4.1.2 illustrates how a successful OUT transaction is executed and closed.

(a) The host issues an OUT token addressed to an OUT endpoint present on this node.
(b) Next, the host sends a data packet under the maximum packet size. The macro writes these data in the relevant endpoint's FIFO.
(c) Upon data reception, the macro automatically returns an ACK response. In addition, it sets registers to be automatically set up and issues a status to the firmware.

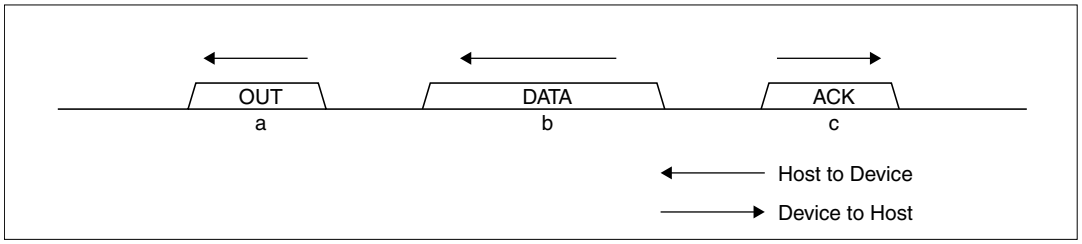


Figure IX.1.4.1.2 OUT Transaction

IN transaction

On an IN endpoint, if maximum packet size data exist in the FIFO or if the firmware has granted permission for short-packet transmission, the macro responds to the IN transaction, returning the data packet.

A permission for short-packet data transmission (including zero-length packets) is granted by setting the EP0ControlIN.EnShortPkt bit and the EP*x*{*x*=a,b,c,d}Control.EnShortPkt bit. When transmitting a short-packet data, make sure that no attempt is made to write any new data into the endpoint's FIFO after the transmission permission is granted and until the transaction is closed.

On the EP0 endpoint, the EP0ControlIN.ForceNAK bit is set after the IN transaction that transmits the short-packet data is closed.

After an ACK response is received in the IN transaction that has returned the data, the transaction is closed, followed by issuance of an IN_TranACK status (EP*x*{*x*=0,a,b,c,d}IntStat.IN_TranACK bit). Also, the FIFO is updated to acknowledge completion of the data transmission and to free the space.

If an ACK response is not received in the IN transaction that has returned the data, the transaction is considered as a failure, followed by issuance of an IN_TranErr status (EP*x*{*x*=0,a,b,c,d}IntStat.IN_TranErr bit). Accordingly, the FIFO is not updated, or no space is freed.

In on an IN endpoint, if no maximum packet size data exist in the FIFO and no permission is granted for short-packet transmission, the IN transaction receives a NAK response and an IN_TranNAK status (EP*x*{*x*=0,a,b,c,d}IntStat.IN_TranNAK bit) is issued to the firmware. Accordingly, the FIFO is not updated, or no space is freed.

Figure IX.1.4.1.3 illustrates how a successful IN transaction is executed and closed.

(a) The host issues an IN token addressed to an IN endpoint present on this node.
(b) If response is possible for this IN transaction, the macro transmits a data packet under the maximum packet size.
(c) The host returns an ACK response. After receiving an ACK response, the macro sets registers to be automatically set up and issues a status to the firmware.

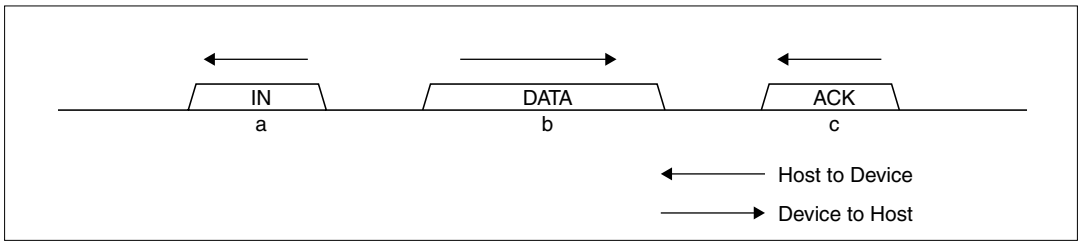


Figure IX.1.4.1.3 IN Transaction

Control transfer

Control transfer on the EP0 endpoint is controlled as a combination of a number of discrete transactions. Figure IX.1.4.1.4 illustrates how control transfer is executed for an OUT data stage.

(a) The host starts control transfer in a SETUP transaction. The device's firmware analyzes the request contents to prepare for responding to a data stage.
(b) The host issues an OUT transaction and executes a data stage, and the device receives data.
(c) The host issues an IN transaction and executes a status stage, and the device returns a zero-length data packet.

Control transfer without a data stage is executed as in this example but without the data stage.
Transition to a status stage is triggered by an issuance of a transaction by the host whose direction is opposite to that of the data stage. Have your firmware monitor an IN_TranNAK status (EP0IntStat.IN_TranNAK bit) as a trigger to transit to a status stage from a data stage.

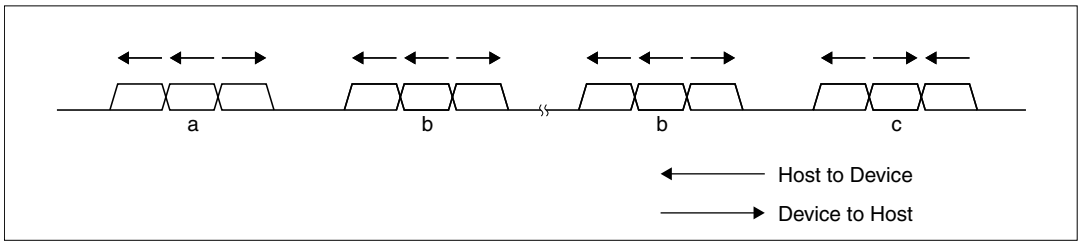


Figure IX.1.4.1.4 Control Transfer Having an OUT Data Stage

Figure IX.1.4.1.5 illustrates how control transfer is executed for an IN data stage.

(a) The host starts control transfer in a SETUP transaction. The device's firmware analyzes the request contents to prepare for responding to a data stage.
(b) The host issues an IN transaction and executes a data stage, and the device transmits data.
(c) The host issues an OUT transaction and executes a status stage, and the device returns an ACK response.

Transition to a status stage is triggered by an issuance of a transaction by the host whose direction is opposite to that of the data stage. Have your firmware monitor an OUT_TranNAK status (EP0IntStat.OUT_TranNAK bit) as a trigger to transit to a status stage from a data stage.

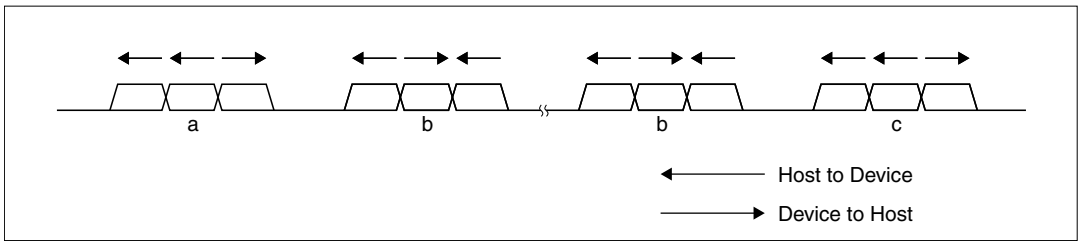


Figure IX.1.4.1.5 Control Transfer Having an IN Data Stage

Since status and data stages in control transfer execute ordinary OUT and IN transactions, flow control using NAK responses works effectively. The device is allowed to prepare for returning responses within a specified time frame.

SETUP stage

The macro automatically executes a SETUP transaction upon reception of a SETUP token addressed to its own node. Have your firmware monitor a RcvEP0SETUP status and analyze the request referring to the EP0Setup_0 through EP0Setup_7 registers to control "control transfer".
If the host has received a request that involves an OUT data stage, clear the INxOUT bit of the EP0Control register to set the EP0 endpoint direction to OUT.
If the host has received a request that involves an IN data stage, set the INxOUT bit of the EP0Control register to set the EP0 endpoint direction to IN.
If the host has received a request that involves no data stage, set the INxOUT bit of the EP0Control register to set the EP0 endpoint direction to IN in order to transit to a status stage.

Data stage/status stage

Transit to the next stage according to the result of request analysis executed by reading the EP0Setup_0 through EP0Setup_7 registers.
If it is an OUT stage, clear the INxOUT of the EP0Control register to set the direction to OUT and control the stage by setting the EP0ControlOUT accordingly. When the SETUP stage is completed, the ForceNAK bit is set.
If it is an IN stage, set the INxOUT of the EP0Control register to set the direction to IN and control the stage by setting the EP0ControlIN accordingly. When the SETUP stage is completed, the ForceNAK bit is set.

Automatic address setting function

This macro provides an automatic address setting function when processing a SetAddress() request in a control transfer at the EP0 endpoint.
This function is available for the firmware when the EP0Setup_0 through EP0Setup_7 registers are checked to confirm the contents and it is proven to be a valid SetAddress() request.
If it is determined to be a valid SetAddress() request, clear or set the EP0ControlIN.ForceNAK and EP0ControlIN.EnShortPkt bits accordingly and set the USB_Address.AutoSetAddress bit before responding to the status stage.
After this function is enabled and the IN transaction at the EP0 endpoint is completed, the macro extracts the address from the data in the SetAddress() request and sets it on the USB_Address.USB_Address bit.
Meanwhile, a SetAddressCmp status (SIE_IntStat.SetAddressCmp bit) is issued to the firmware.
After this function is enabled, if any other transaction is invoked at the EP0 endpoint before an IN transaction is executed, this function is cancelled and the USB_Address.AutoSetAddress bit is cleared. Accordingly, a SetAddressCmp status is not issued to the firmware.

Descriptor return function

This macro provides a descriptor return function that is useful for an request that requires data and is issued more than once during control transfer at the EP0 endpoint (for example, during a GetDescriptor() request).
The firmware can use this function for a request that involves an IN data stage.
Clear the EP0ControlIN.ForceNAK bit, and before starting responding to the data stage, set the top address of the data to be returned that is within the FIFO's descriptor region on the DescAdrs_H, L register as well as the total number of bytes in the return data on the DescSize_H, L register and set the EP0Control.ReplyDescriptor bit.
The descriptor return function executes IN transactions by returning data packets in response to IN transactions until it finishes sending all of a specified number of data. If a fractional number of data exist against the maximum packet size, the descriptor return function sets EP0ControlIN.EnShortPkt, enabling response to IN transactions until the entire data return is completed.
After returning all the specified number of data, the macro clears the EP0Control.ReplyDescriptor bit and issues a DescriptorCmp status (FIFO_IntStat.DescriptorCmp bit) to the firmware.
For details of the descriptor region, see the section on the FIFO in the functional description.

Bulk transfer/interrupt transfer

Bulk and interrupt transfers at the general-purpose endpoints, EPa, EPb, EPc, and EPd, can be controlled either as a data flow or as a series of discrete transactions (see the "Transaction" section).

Data flow control

This section describes controlling standard data flows in OUT and IN transfers.

OUT transfer

Data received from an OUT transfer are placed on the FIFO region at the respective endpoints.
To read the FIFO data, select one and only one endpoint using the CPU_JoinRd register. The FIFO data of the selected endpoint can be read sequentially with the EPnFIFOforCPU, according to the order of reception. Also, you can refer to the EPnRdRemain_H and EPnRdRemain_L registers to check the number of remaining data. Reading from an blank FIFO causes dummy reading to be performed.
Be sure to start reading data after ensuring that no data return responses are returned to IN transactions by setting the ForceNAK bit, for example, if you want to set an IN endpoint for data reading using the CPU_JoinRd register.
If the FIFO has available space for receiving data packets, the macro automatically responds to OUT transactions to receive data. This enables the firmware to perform OUT transfer without individual transaction control. Note, however, that the EP*x*{*x*=a,b,c,d}Control.ForceNAK bit of the endpoint is set if short packets are received (including zero-length data packet) when the EP*x*{*x*=a,b,c,d}Control.DisAF_NAK_Short bit is cleared. Clear this bit when the next data transfer is ready.
Figure IX.1.4.1.6 illustrates the data flow in OUT transfer. The FIFO region for an OUT endpoint is connected to the Port interface. Also, the FIFO region assigned to this endpoint is assumed to be twice as large as the maximum packet size.

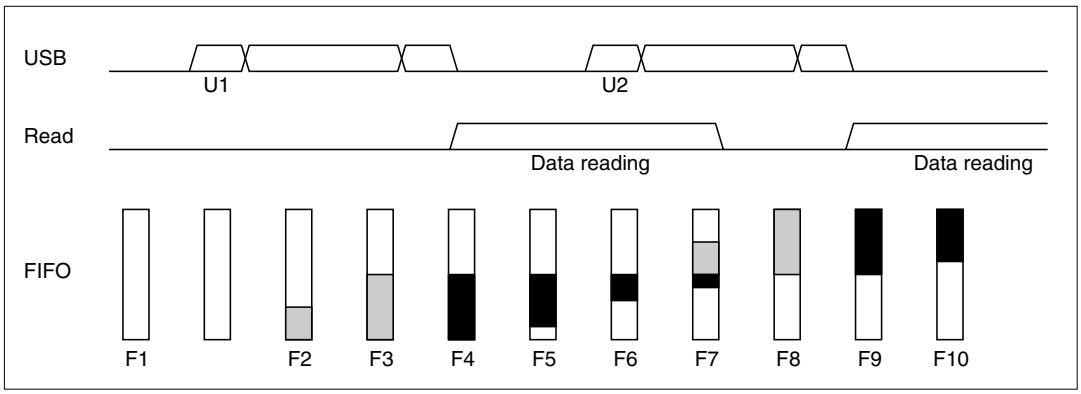


Figure IX.1.4.1.6 Example of Data Flow in OUT Transfer

(U1) Data transfer of the maximum packet size is performed in the first OUT transaction.
(U2) Data transfer of the maximum packet size is performed in the second OUT transaction.
(F1) The FIFO is blank.
(F2) An OUT transaction is developing, and data reception has started in the FIFO. At this point, the FIFO data is not considered to be valid since the transaction is not closed.
(F3) Although data packet reception is completed from the OUT transaction, the FIFO data is not considered to be valid since the transaction is not closed.
(F4) The OUT transaction is closed and the received data are considered to be valid.
(F5) The CPU reads the received data.
(F6) The amount of the remaining valid data in the FIFO is reduced by reading.
(F7) Starting the next transaction starts writing data. Data reading can be continued as long as any valid data remains.
(F8) Data reading is stopped as there is no valid data left. The second OUT transaction is not closed yet.
(F9) The second OUT transaction is closed, causing the FIFO data to become valid.
(F10) The CPU restarts reading the received data.

IN transfer

Place data transmitted thorough IN transfer on each endpoint's FIFO.

To write data into the FIFO, select one and only one endpoint using the CPU_JoinWr register. Data can be written in the selected endpoint's FIFO by using the EPnFIFOforCPU register, which are transmitted in data packets in the order of writing. Also, you can refer to the EPnWrRemain_H and EPnWrRemain_L registers to check the available space in the FIFO. An attempt to write in a full FIFO causes dummy writing to be performed. Be sure to start writing data after ensuring that no data are received from the OUT transactions by setting the ForceNAK bit, for example, if you want to set an OUT endpoint for data writing using the CPU_JoinWr register.

If the FIFO contains data exceeding the maximum packet size, the macro automatically responds to IN transactions to perform data transmission. This enables the firmware to perform IN transfer without individual transaction control. Note, however, that you should set the EnShortPkt bit if you need to transmit a short packet at the end of the data transfer. Since this bit is cleared when the IN transaction which has transmitted the short packet is closed, you can set it after data is completely written into the FIFO.

Figure IX.1.4.1.7 illustrates the data flow in IN transfer. The FIFO region for an IN endpoint is connected to the Port interface. Also, the FIFO region assigned to this endpoint is assumed to be twice as large as the maximum packet size.

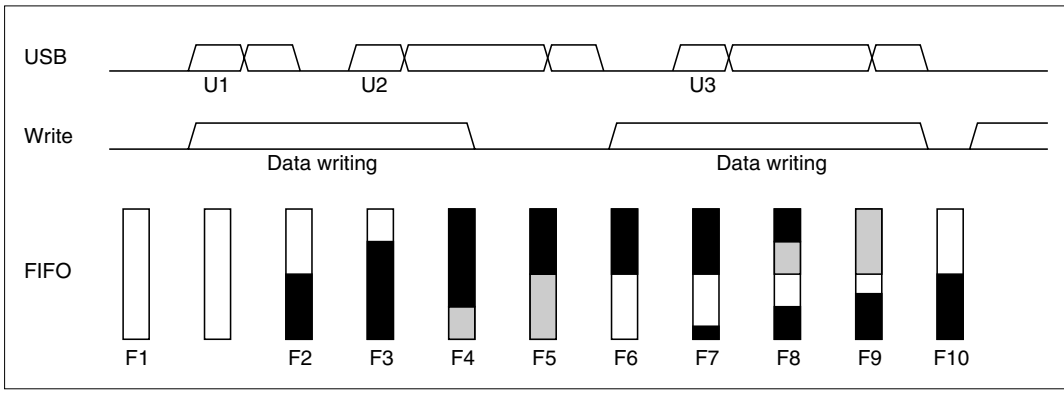


Figure IX.1.4.1.7 Example of Data Flow in IN Transfer

(U1) In the first IN transaction, an NAK response is returned since the FIFO has no valid maximum packet size data.
(U2) Data transfer of the maximum packet size is performed in the second IN transaction.
(U3) Data transfer of the maximum packet size is performed in the third IN transaction.
(F1) The FIFO is blank.
(F2) Valid data is written into the FIFO by the CPU.
(F3) As the FIFO still has an available space, the CPU continues writing data.
(F4) Since the FIFO contains valid maximum packet size data, the macro responds to the IN transaction with data packet transmission. As the transaction is not closed yet, the region from which data are transmitted is not freed. The FIFO is full, causing the CPU to stop writing.
(F5) Although data packet transmission in the IN transaction has been completed, the FIFO region is not freed since the transaction is not closed.
(F6) The FIFO region is freed as the transaction is closed upon reception of an ACK handshake packet.
(F7) As the FIFO now has some available space, data writing is resumed.
(F8) The macro responds to an IN transaction and transmits a data packet. Since the FIFO has some available space, the CPU continues writing data.
(F9) Although data packet transmission in the IN transaction has been completed, the FIFO region is not freed since the transaction is not closed. Since the FIFO has some available space, the CPU continues writing data.
(F10) The FIFO region is freed when the transaction is closed upon reception of an ACK handshake packet. Although the CPU stops writing as all the available space has been consumed, it is resumed upon closing of the IN transaction that creates available space.

Auto-negotiation function

This function automatically performs Suspend detection, Reset detection and Resume detection, checking the state of the USB bus for each operation. You can check each interruption (DetectReset and DetectSuspend) to confirm what has been actually performed.

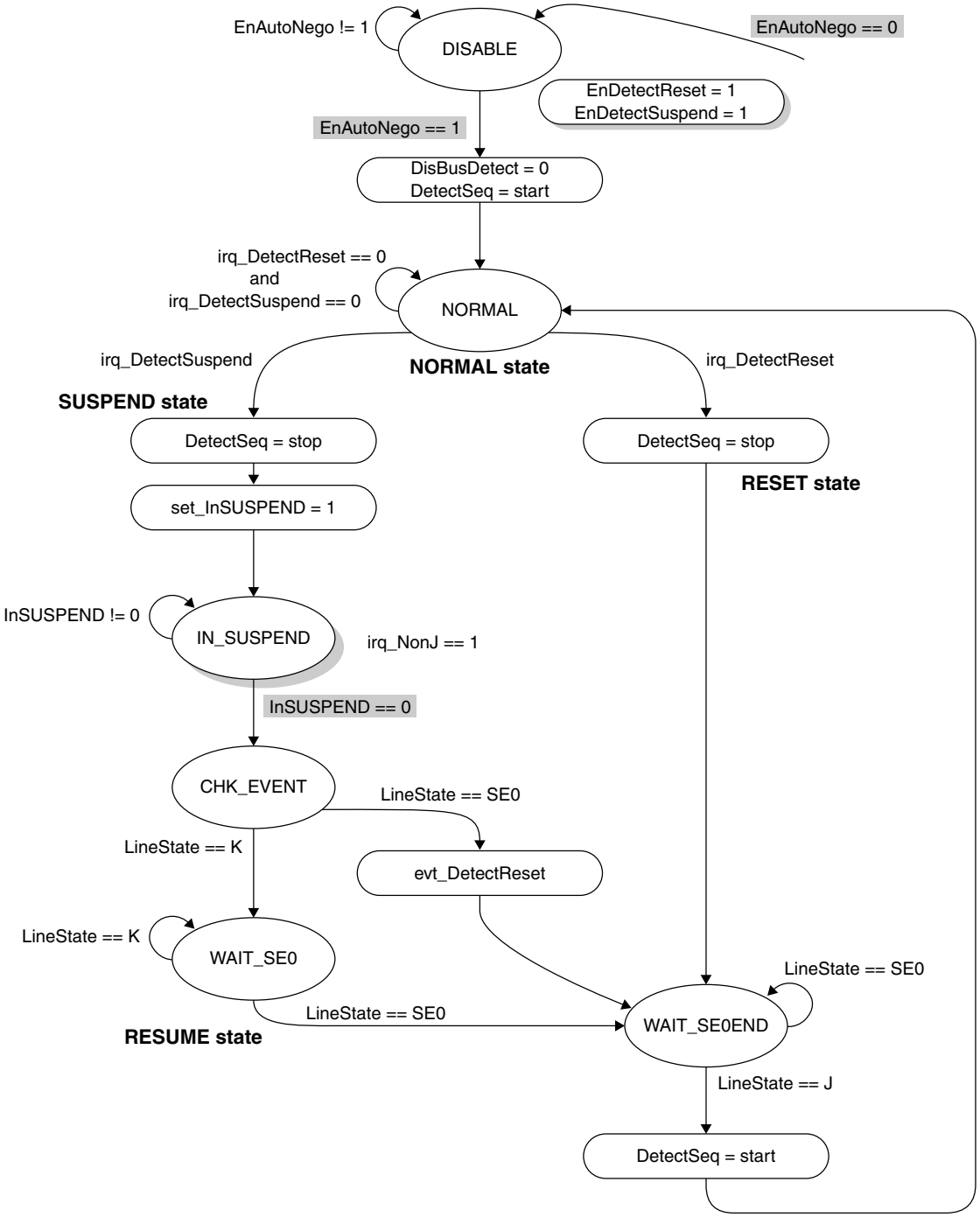


Figure IX.1.4.1.8 Auto-negotiator

(1) DISABLE

The macro enters this state when the USB_Control.EnAutoNego bit is cleared.

To enable the auto-negotiation function, set interruptions for Reset detection (SIE_IntEnb.EnDetectReset) and Suspend detection (SIE_IntEnb.EnDetectSuspend) before setting the USB_Control.EnAutoNego bit and give permission to event detection interruption.

Enabling the auto-negotiation function automatically clears the USB_Control.DisBusDetect bit and enables the event detection function. While the auto-negotiation function is enabled, never set the USB_Control. DisBusDetect bit.

(2) NORMAL

This is a state of waiting for Reset or Suspend detection.

The state is determined to be Reset if SE0 of 2.5 μs or greater, and it is determined to be Suspend if no activities are detected beyond 3 ms. Concurrently with judgment as described above, an interruption for Reset detection or Suspend detection is generated, and the SIE_IntStat.DetectReset bit and the SIE_IntStat. DetectSuspend bit are set.

If the state is determined to be Suspend, suspend the event detection function once and enter the IN_SUSPEND state.

(3) IN_SUSPEND

When the state is determined to be suspended, H/W automatically sets the USB_Control.InSUSPEND bit and the macro enters the IN_SUSPEND state. This USB_Control.InSUSPEND bit enables the function of detecting changes of buses from FS-J, only enabling detection of Resume or Reset from the host.

The ability to reduce current consumption during Suspend depends on the application. This macro provides SNOOZE function for reducing current consumption. To use the function of reducing current consumption when the auto-negotiation function is enabled, be sure to check that the USB_Control.InSUSPEND bit is set before staring the current consumption reducing function.

At this time, in order to detect Resume (FS-K) that indicates the end of Suspend, set the SIE_IntEnb. EnNonJ bit in the firmware when the macro enters this state to give permission to NonJ interruption.

When NonJ interruption status (SIE_IntStat.NonJ) is set, it is interpreted as an indication of return from Suspend, and the macro enters the CHK_EVENT state after the USB_Control.InSUSPEND bit is cleared in the firmware.

In an application with a remote wake-up function enabled, if it is determined that the macro must return from Suspend, set the USB_Control.SendWakeup bit in this state and output FS-K at least for 1 ms but do not exceed 15 ms.

(4) CHK_EVENT

In this state, the macro checks the USB cable and determines that the state is Resume if FS-K is detected, and that it is Reset if SE0 is detected. When determined to be Reset, set the SIE_IntStat.DetectReset bit.

Note that you should terminate this auto-negotiation function as soon as the USB cable is unplugged; in none of the above states, the macro does not consider the implication of USB cable disconnection.

Description by negotiation function

Suspend detection

When the USB_Control.DisBusDetect bit is set to 0, the macro hardware automatically performs the following Suspend detection sequence.

(1) The internal timer checks that there is no data transmission/reception (continues to detect "J" in USB_Status.LineState[1:0]) for 3 ms or longer (T_1).
(2) At T_2, if "J" is detected in USB_Status.LineState[1:0], set the SIE_IntStat.DetectSuspend bit.
(3) If the SIE_IntEnb.EnDetectSuspend and MainIntEnb.EnSIE_IntStat bits are set, the macro asserts the #INT signal.

If the SIE_IntStat.DetectSuspend bit is set, on the firmware that controls this macro, set the USB_Control. DisBusDetect bit to 1 and USBSNZ (D5/0x4909) to 1 to start processing Snooze before reaching T_4. As for self-powered products, however, the firmware does not have to perform Snooze. (Figure IX.1.4.1.9 shows the operation when Snooze is performed.)

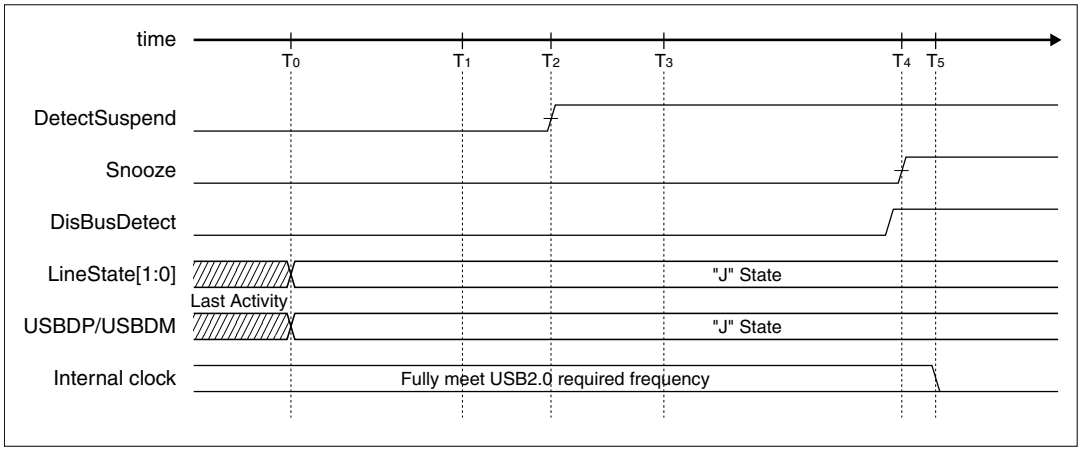


Figure IX.1.4.1.9 Suspend Timing (FS mode)

Reset detection

When the USB_Control.DisBusDetect bit is set to 0, the macro hardware automatically performs the following Reset detection sequence.

(1) The internal timer checks that it has continued to detect "SE0" in USB_Status.LineState[1:0]) for 2.5 μs or longer (T_1).
(2) At T_2, if "SE0" is detected in USB_Status.LineState[1:0], the macro sets the SIE_IntStat.DetectReset bit.
(3) If the SIE_IntEnb.EnDetectReset and MainIntEnb.EnSIE_IntStat bits are set, the macro asserts the #INT signal.

If the SIE_IntStat.DetectReset bit is set, on the firmware that controls this macro, set the USB_Control. DisBusDetect bit to 1.

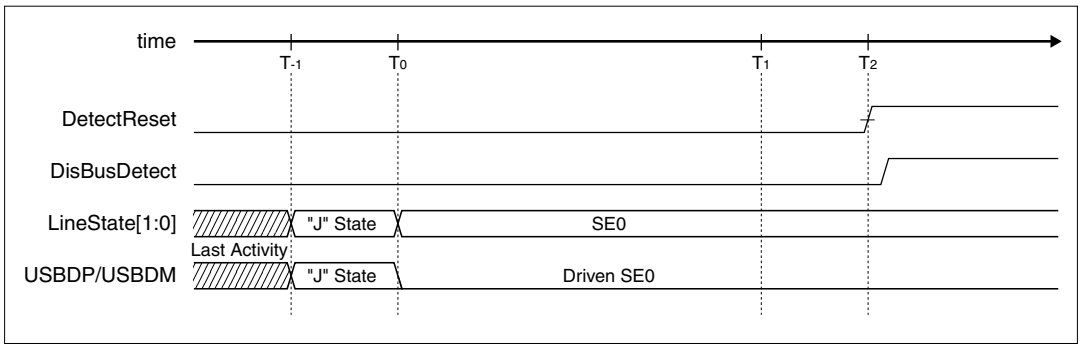


Figure IX.1.4.1.10 Reset Timing (FS mode)

Issuing resume

This section describes how to enable automatic resume to be triggered by some cause when remote wakeup is supported and the remote wakeup function is enabled from the host.
Remote wakeup can only be enabled 5 ms after the bus enters the Idle state. Furthermore, the current used before the USB Suspend state cannot be pulled from the VBUS until 10 ms has elapsed after the Resume signal output.

The S1C17501 supports Snooze state. This section describes the operation for issuing Resume when the oscillation circuit is in operation (USB_CLK_EN (D4/0x4906) = 1, not in Sleep). Steps (3), (4), (8) and (9) below are handled by the macro hardware automatically. Perform steps (1), (2), (6), (6a) and (10) on the firmware that controls this macro.

(1) Clear the SIE_IntEnb.EnNonJ and USBSNZ (D5/0x4909) bits. This is to cause this macro return from Snooze for automatic wakeup.
(2) Set the USB_Control.SendWakeup bit and send out the Resume signal.
(3) The macro sets XcvrControl.OpMode[1:0] to “Disable Bit Stuffing and NRZI encoding” and prepares for transmission of “All 0” data.
(4) The macro starts data transmission and sends out “FS K” (the Resume signal) to a downstream port.
(5) The downstream port detects this Resume signal and returns “FS K” (the Resume signal) onto the bus.
(6) Clear the USB_Control.SendWakeup bit and suspend Resume signal send-out. After that, clear the USB_Control.InSUSPEND bit.
(7) The downstream port suspends Resume signal send-out. Here, note that the Resume signal from downstream port (host) has EOP of LS at the end.

To detect the Resume signal sent from downstream port, the following procedure is needed after step (6) is performed.

(6a) Set the USB_Control.StartDetectJ bit.
(7) The downstream port suspends Resume signal send-out. Here, note that the Resume signal from downstream port (host) has EOP of LS at the end.
(8) The SIE_IntStat.DetectJ bit is set.
(9) If the SIE_IntEnb.EnDectectJ bit is set, the macro asserts the #INT signal.
(10) Clear the USB_Control.StartDetectJ bit.

However, steps (6a)–(10) is not necessary when the auto-negotiation function is used, so just wait another event.

This section describes the operation of the oscillation circuit by assuming that it is in operation (USB_CLK_EN (D4/0x4906) = 1, not in Sleep). If the oscillation circuit is in the Sleep state (deactivated), OSC power-up time is needed before returning from the Snooze state (with USBSNZ (D5/0x4909) reset from 1 to 0).

Detecting resume

When the USB is suspended, "J" is observed on the bus (USB_Status.LineState[1:0] is "J"). If "K" is observed on the bus, it means the instruction for wakeup (Resume) is received from the downstream port. This section describes the operation when Resume is detected, assuming that this macro is in the Snooze state when the USB is suspended. Use the firmware that controls this macro to perform steps (4), (5), (5a) and (9). The other steps are handled by the macro hardware automatically.

(1) The bus transits from "J" to "K".
(2) The macro sets the SIE_IntStat.NonJ bit.
(3) If the SIE_IntEnb.EnNonJ and MainIntEnb.EnSIE_IntStat bits are set, the macro asserts the #INT signal.
(4) Clear USBSNZ (D5/0x4909).
(5) Clear the USB_Control.SendWakeup bit and suspend Resume signal send-out. After that, clear the USB_Control.InSUSPEND bit.
(6) The downstream port suspends "K" send-out.

To detect the Resume signal sent from downstream port, the following procedure is needed after step (5) is performed.

(5a) Set the USB_Control.StartDetectJ bit.
(6) The downstream port suspends "K" send-out.
(7) The SIE_IntStat.DetectJ bit is set.
(8) If the SIE_IntEnb.EnDectectJ bit is set, the macro asserts the #INT signal.
(9) Clear the USB_Control.StartDetectJ bit.

However, steps (5a)–(9) is not necessary when the auto-negotiation function is used, so just wait another event.

This section describes the operation of the oscillation circuit by assuming that it is in operation (USB_CLK_EN (D4/0x4906) = 1, not in Sleep). If the oscillation circuit is in the Sleep state (deactivated), OSC power-up time is needed before returning from the Snooze state (with USBSNZ (D5/0x4909) reset from 1 to 0).

Cable plug-in

This section describes the operation that is carried out when the macro is connected to the hub or the host (via cable plug-in). Use the firmware that controls this macro to perform steps (3) and (4). Steps (1) and (2) are handled by the macro hardware automatically.

(1) When the cable is connected, VBUS turns to HIGH and the macro sets the USB_Status.VBUS and SIE_IntStat.VBUS_Changed bits (T_0).
(2) If the SIE_IntEnb.EnVBUS_Changed and MainIntEnb.EnSIE_IntStat bits are set, the macro asserts the #INT signal.
(3) Set USB_CLK_EN (D4/0x4906) to start supplying the USB clock (T_1).
(4) Clear USBSNZ (D5/0x4909) (T_2).
(5) The downstream port sends out Reset (T_4).

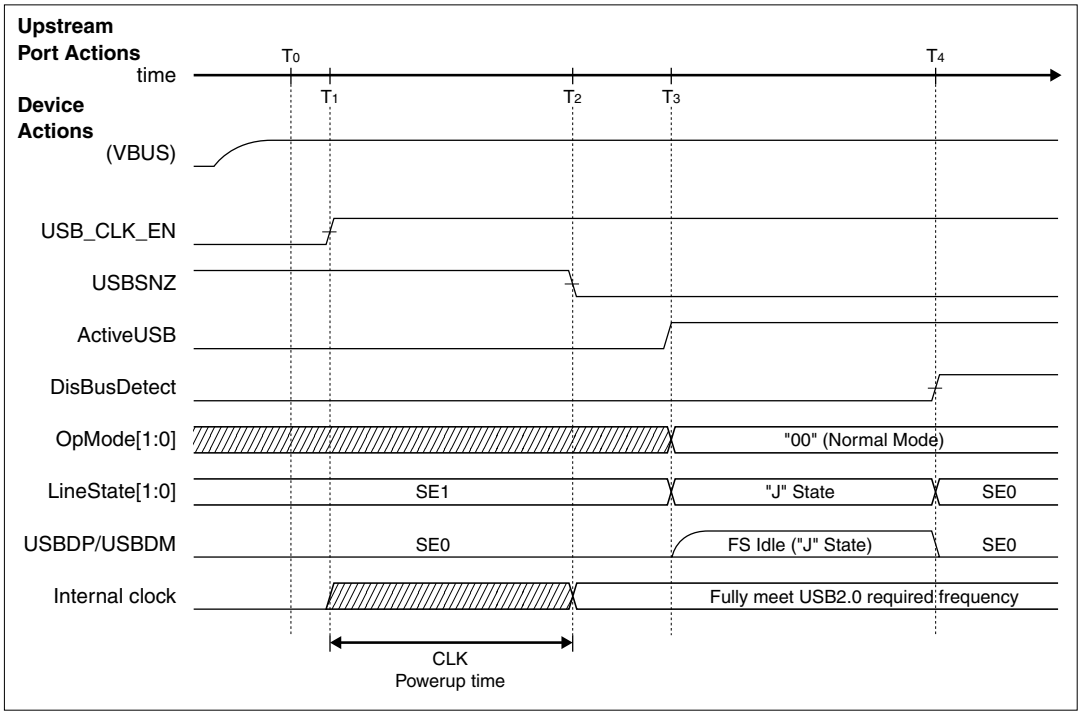


Figure IX.1.4.1.11 Device Attach Timing

Table IX.1.4.1.5 Device Attach Timing Values

Timing parameter	Description	Value
T_0	VBUS is enabled.	0 (Reference)
T_1	Set USB_CLK_EN to 1 (on the firmware). The clock input starts.	T_1
T_2	Clear USBSNZ to 0 (on the firmware).	T_1 + 250 ms < T_2
T_3	Set ActiveUSB to 1. Set OpMode[1:0] to 00 (on the firmware).	T_0 + 100 ms < T_3
T_4	The downstream port sends out Reset. Set DisBusDetect to 1 (on the firmware).	T_3 + 100 ms < T_4

IX.1.4.2 FIFO Management

FIFO memory map

This section describes the memory map for the FIFO region.

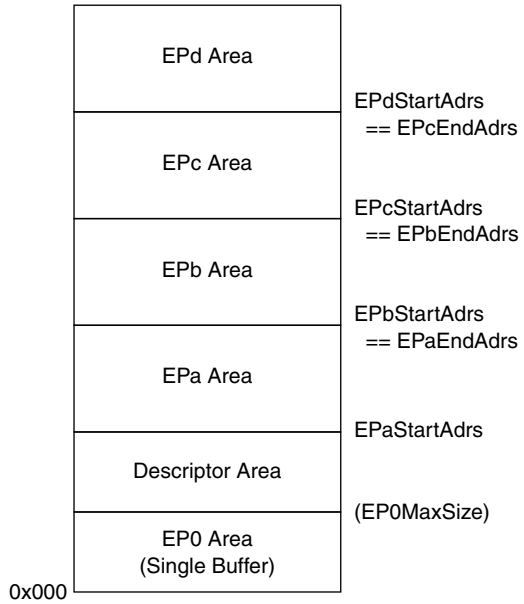


Figure IX.1.4.2.1 FIFO Memory Map

The FIFO memory is roughly divided into six areas: EP0 area, descriptor area, EPa area, EPb area, EPc area, and EPd area, and each of these areas can be divided according to the settings for the EP0MaxSize register, EPaStartAdrs register, EPbStartAdrs register, EPcStartAdrs register, and EPdStartAdrs register.

The EP0 area is used for the required USB endpoint 0, and can be used both for IN and OUT directions. This area is uniquely determined to be the maximum packet size of endpoint 0 that is set up in the EP0MaxSize register. This means that it can only receive/transmit one packet (Single Buffer) at a time.

EPa, EPb, EPc, and EPd areas are for the general-purpose endpoint that can take an endpoint number and an IN/OUT setting. The EPa area extends from the address set in the EPaStartAdrs register up to the point before the address set in the EPbStartAdrs register. The EPb area extends from the address set in the EPbStartAdrs register up to the point before the address set in the EPcStartAdrs register. The EPd area extends from the address set in the EPdStartAdrs register up to the end of FIFO RAM. The addresses available in the area setup registers must be written in the unit of four bytes (meaning that the lowest two bits cannot be written). Additionally, a space exceeding the maximum packet size must be assigned to these areas. Although there should be no problem as far a value larger than the maximum packet size is assigned, we recommend that you use its integral multiple to set them up.

The descriptor area extends from the address set in the EP0MaxSize register up to the point before the address set in the EPaStartAdrs. (Actually, the entire FIFO region can be used as the descriptor area. We recommend, however, that the area described here be used in order to avoid operational contentions.) The practical use is described later.

Set the EPnControl.AllFIFO_Clr bit for the initial setting or re-setting of an area set-up register. Once the initial setting for an area is established, the EPnControl.AllFIFO_Clr bit is cleared. This bit will never cause the RAM data to be cleared. Therefore, unless you have changed the descriptor area, there is no need to re-set the information recorded within the area since will never be cleared otherwise.

Using the descriptor area

The descriptor area provides high-speed, straightforward execution of part of operations for packets received/ transmitted via EP0, or a standard request. Among contents of standard requests, write those in this area that are uniquely determined by the device during the initial setup stage following power-on to automatically execute the data stage included in the request simply by setting the top address and the data size in response to a request from the host. Accordingly, this technique eliminates the need of writing data in the EP0 area, enabling very quick response to a request.

Writing data in the descriptor area

To write data in the descriptor area, first set the write start address in the DescAdrs_H and DescAdrs_L registers, and then write data in the DescDoor register (RegWindowSel == 0x2). After completing writing data, the DescAdrs_H and DescAdrs_L registers are automatically incremented by one, enabling sequential writing in the DescDoor register (RegWindowSel == 0x2) when writing data at a series of adjacent addresses. Note that this incrementing function does not mean that written data can be read when writing and reading are executed sequentially; it only increments by one for both writing and reading.

Reading data from descriptor area

To read data from the descriptor area, first set the read start address in the DescAdrs_H and DescAdrs_L registers, and then read data from the DescDoor register (RegWindowSel == 0x2). After completing reading data, the DescAdrs_H and DescAdrs_L registers are automatically incremented by one, enabling sequential reading in the DescDoor register (RegWindowSel == 0x2) when reading data from a series of adjacent addresses. Note that this incrementing function does not mean that written data can be read when writing and reading are executed sequentially; it only increments by one for both writing and reading.

Executing data stage (IN) in the descriptor area

To use written data in response to a request from EP0, set the top address of the data to be transmitted to the data stage, set the data size specified in the request in the DescSize_H and DescSize_L registers, and then set the EP0Control.ReplyDescriptor bit to 1.

After receiving the IN token from the host, the macro start transmitting data to the host, automatically splitting them into the maximum packet size (set in the EP0MaxSize). In addition, if the value in the DescSize_H or DescSize_L register is under the maximum packet size, or if the remaining number of data after splitting, the macro automatically transmits such data as short packets. When the specified number of data are completely transmitted, the EP0Control.ReplyDescriptor is cleared and the FIFO_IntStat.DescriptorCmp is set. At this stage, the FIFO_IntEnb.EnDescriptorCmp bit is set and the MainIntEnb.EnEPrIntStat bit is set as well, the #INT signal is asserted at the same time.

If the process enters a status stage before the transmitted amount reaches the specified number of data (that is, if an OUT token is received), the EP0Control.ReplyDescriptor is automatically cleared to suspend this function. At the same time, the EP0IntStat.OUT_TranNAK status and the FIFO_IntStat.DescriptorCmp status are set. If either of the following sets of bits are set, the #INT signal is asserted at the same time:

(1) The EP0IntEnb.EnOUT_TranNAK, MainIntEnb.EnEP0IntStat and MainIntEnb.EnEPrIntStat bits, or
(2) The FIFO_IntEnb.EnDescriptorCmp and MainIntEnb.EnEPrIntStat bits.

Accessing to FIFO by CPU

To enable the CPU to access the FIFO, set the bit of the relevant endpoint of the CPU_JoinRd and CPU_JoinWr registers to 1 and execute reading and writing via the EPnFIFOforCPU register. For each of the CPU_JoinRd and CPU_JoinWr registers, you can only set one bit out of the four bits. If you attempt to set more than one bit at a time, only the highest bit is set.
The EPnRdRemain_H and EPnRdRemain_L registers indicate the remaining number of data that can be read at the endpoint set in the CPU_JoinRd register. The EPnWrRemain_H and EPnWrRemain_L registers indicate the remaining area space available for writing at the endpoint set in the CPU_JoinWr register.

Note that, if the CPU_JoinRd register is set when register dumping is planned for debugging of a CPU using ICE, data will be read from the FIFO upon dumping the register.

Limiting access to FIFO

The FIFO of this macro allows concurrent execution of data reception/transmission between the macro and the USB and/or the Port and writing/reading to and from the CPU. Because of this, there are two limitations for accessing the FIFO (for writing and reading) from the CPU (the firmware):

(1) From the CPU, no writing is allowed to the same endpoint while the USB or the Port is writing data to the FIFO.
(2) No reading from the CPU is allowed from the same endpoint while the USB or the Port is reading from the FIFO.

Never execute these operations; they may destroy data continuity.

IX.1.4.3 Snooze

This macro has Snooze function which enables very low power operation when USB is not active.
When the SNOOZE signal is asserted by writing 1 to USBSNZ (D5/0x4909), the following procedure will be performed and allows to stop feeding 48 MHz clock after 5 clocks inputs.

- Disable USB differential comparator
- Allow asynchronous access for VBUS_Changed and NonJ bits of the SIE_IntStat register. (Monitor the USB interface input pins)
- Mask Read/Write for synchronous registers
- Mask synchronous interrupt

This macro will resume after 5 clocks (oscillation must be stable) when the SNOOZE signal is negated.

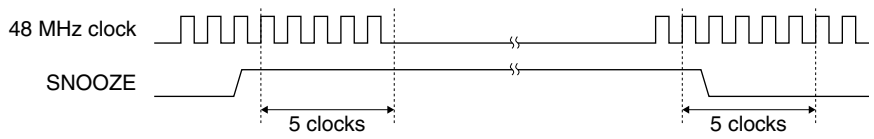


Figure IX.1.4.3.1 Snooze Sequence

Snooze mode should be set or canceled by the following procedure:

Setting snooze mode

(1) Write 0x96 to the CMU Write Protect Register (0x4920) to disable write protection for the CMU registers.
(2) Set USBSNZ (D5/0x4909) in the USB Wait Control Register to 1 to enable the snooze control.
(3) Set USB_CLK_EN (D4/0x4906) in the Gated Clock Control Register 0 to 0 to stop supplying the USB clock.
(4) Write a value other than 0x96 to the CMU Write Protect Register (0x4920) to enable write protection for the CMU registers.

Canceling snooze mode

(1) Write 0x96 to the CMU Write Protect Register (0x4920) to disable write protection for the CMU registers.
(2) Set USB_CLK_EN (D4/0x4906) in the Gated Clock Control Register 0 to 1 to start supplying the USB clock.
(3) Set USBSNZ (D5/0x4909) in the USB Wait Control Register to 0 to disable the snooze control.
(4) Write a value other than 0x96 to the CMU Write Protect Register (0x4920) to enable write protection for the CMU registers.

IX.1.5 Registers

IX.1.5.1 List of Registers

• ***Italic & bold*** represents readable/writable registers in SNOOZE/SLEEP mode.

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4500	MainIntStat	R/(W)	0x00	***SIE_IntStat***	EPrIntStat		FIFO_IntStat			EP0IntStat	RcvEP0SETUP
0x4501	SIE_IntStat	R/(W)	0x00	***VBUS_Changed***	***NonJ***	DetectReset	DetectSuspend	RcvSOF	DetectJ		SetAddressCmp
0x4502	EPrIntStat	R	0x00					EPdIntStat	EPcIntStat	EPbIntStat	EPaIntStat
0x4503											
0x4504	FIFO_IntStat	R/(W)	0x00	DescriptorCmp							
0x4505											
0x4506											
0x4507	EP0IntStat	R/(W)	0x00			IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x4508	EPaIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x4509	EPbIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x450a	EPcIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x450b	EPdIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x450c											
0x450d											
0x450e											
0x450f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4510	MainIntEnb	R/W	0x00	***EnSIE_IntStat***	EnEPrIntStat		EnFIFO_IntStat			EnEP0IntStat	EnRcvEP0SETUP
0x4511	SIE_IntEnb	R/W	0x00	***EnVBUS_Changed***	***EnNonJ***	EnDetectReset	EnDetectSuspend	EnRcvSOF	EnDetectJ		EnSetAddressCmp
0x4512	EPrIntEnb	R/W	0x00					EnEPdIntStat	EnEPcIntStat	EnEPbIntStat	EnEPaIntStat
0x4513											
0x4514	FIFO_IntEnb	R/W	0x00	EnDescriptorCmp							
0x4515											
0x4516											
0x4517	EP0IntEnb	R/W	0x00			EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x4518	EPaIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x4519	EPbIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x451a	EPcIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x451b	EPdIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x451c											
0x451d											
0x451e											
0x451f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4520	RevisionNum	R	0x12	***Revision Number[7:0]***							
0x4521	USB_Control	R/W	0x00	DisBusDetect	EnAutoNego	InSUSPEND	StartDetectJ	SendWakeup			ActiveUSB
0x4522	USB_Status	R	0xXX	***VBUS***	***1(FS)***					***LineState[1:0]***	
0x4523	XcvrControl	R/W	0x81	RpuEnb						OpMode[1:0]	
0x4524	USB_Test	R/W	0x00	EnUSB_Test				Test_SE0_NAK	Test_J	Test_K	Test_Packet
0x4525	EPnControl	W	0x00	AllForceNAK	EPrForceSTALL	AllFIFO_Clr					EP0FIFO_Clr
0x4526	EPrFIFO_Clr	W	0x00					EPdFIFO_Clr	EPcFIFO_Clr	EPbFIFO_Clr	EPaFIFO_Clr
0x4527											
0x4528											
0x4529											
0x452a											
0x452b											
0x452c											
0x452d											
0x452e	FrameNumber_H	R	0x80	FnInvalid					FrameNumber[10:8]		
0x452f	FrameNumber_L	R	0x00	FrameNumber[7:0]							

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4530	EP0Setup_0	R	0x00								
0x4531	EP0Setup_1	R	0x00								
0x4532	EP0Setup_2	R	0x00								
0x4533	EP0Setup_3	R	0x00								
0x4534	EP0Setup_4	R	0x00								
0x4535	EP0Setup_5	R	0x00								
0x4536	EP0Setup_6	R	0x00								
0x4537	EP0Setup_7	R	0x00								
0x4538	USB_Address	R/W	0x00	AutoSetAddress	USB_Address[6:0]						
0x4539	EP0Control	R/W	0x00	INxOUT							ReplyDescriptor
0x453a	EP0ControlIN	R/W	0x00		EnShortPkt		ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x453b	EP0ControlOUT	R/W	0x00	AutoForceNAK			ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x453c											
0x453d											
0x453e											
0x453f	EP0MaxSize	R/W	0x08		EP0MaxSize[6:3]						

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4540	EPaControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x4541	EPbControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x4542	EPcControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x4543	EPdControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x4544											
0x4545											
0x4546											
0x4547											
0x4548											
0x4549											
0x454a											
0x454b											
0x454c											
0x454d											
0x454e											
0x454f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4550	EPaMaxSize_H	R/W	0x00							EPaMaxSize[9:8]	
0x4551	EPaMaxSize_L	R/W	0x00	EPaMaxSize[7:0]							
0x4552	EPaConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint		EndPointNumber[3:0]			
0x4553	EPaConfig_1	R/W	0x00	ISO							
0x4554	EPbMaxSize_H	R/W	0x00							EPbMaxSize[9:8]	
0x4555	EPbMaxSize_L	R/W	0x00	EPbMaxSize[7:0]							
0x4556	EPbConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint		EndPointNumber[3:0]			
0x4557	EPbConfig_1	R/W	0x00	ISO							
0x4558	EPcMaxSize_H	R/W	0x00							EPcMaxSize[9:8]	
0x4559	EPcMaxSize_L	R/W	0x00	EPcMaxSize[7:0]							
0x455a	EPcConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint		EndPointNumber[3:0]			
0x455b	EPcConfig_1	R/W	0x00	ISO							
0x455c	EPdMaxSize_H	R/W	0x00							EPdMaxSize[9:8]	
0x455d	EPdMaxSize_L	R/W	0x00	EPdMaxSize[7:0]							
0x455e	EPdConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint		EndPointNumber[3:0]			
0x455f	EPdConfig_1	R/W	0x00	ISO							

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4570	EPaStartAdrs_H	R/W	0x00					EPaStartAdrs[11:8]			
0x4571	EPaStartAdrs_L	R/W	0x00	EPaStartAdrs[7:2]							
0x4572	EPbStartAdrs_H	R/W	0x00					EPbStartAdrs[11:8]			
0x4573	EPbStartAdrs_L	R/W	0x00	EPbStartAdrs[7:2]							
0x4574	EPcStartAdrs_H	R/W	0x00					EPcStartAdrs[11:8]			
0x4575	EPcStartAdrs_L	R/W	0x00	EPcStartAdrs[7:2]							
0x4576	EPdStartAdrs_H	R/W	0x00					EPdStartAdrs[11:8]			
0x4577	EPdStartAdrs_L	R/W	0x00	EPdStartAdrs[7:2]							
0x4578											
0x4579											
0x457a											
0x457b											
0x457c											
0x457d											
0x457e											
0x457f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x4580	CPU_JoinRd	R/W	0x00					JoinEPdRd	JoinEPcRd	JoinEPbRd	JoinEPaRd
0x4581	CPU_JoinWr	R/W	0x00					JoinEPdWr	JoinEPcWr	JoinEPbWr	JoinEPaWr
0x4582	EnEPnFIFO_Access	R/W	0x00							EnEPnFIFO_Wr	EnEPnFIFO_Rd
0x4583	EPnFIFOforCPU	R/W	0xXX	EPnFIFOData[7:0]							
0x4584	EPnRdRemain_H	R	0x00					EPnRdRemain[11:8]			
0x4585	EPnRdRemain_L	R	0x00	EPnRdRemain[7:0]							
0x4586	EPnWrRemain_H	R	0x00					EPnWrRemain[11:8]			
0x4587	EPnWrRemain_L	R	0x00	EPnWrRemain[7:0]							
0x4588	DescAdrs_H	R/W	0x00					DescAdrs[11:8]			
0x4589	DescAdrs_L	R/W	0x00	DescAdrs[7:0]							
0x458a	DescSize_H	R/W	0x00							DescSize[9:8]	
0x458b	DescSize_L	R/W	0x00	DescSize[7:0]							
0x458c											
0x458d											
0x458e											
0x458f	DescDoor	R/W	0x00	DescMode[7:0]							

IX.1.5.2 Detailed Description of Registers

0x4500: MainIntStat (Main Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
MainIntStat (Main interrupt status)	**0x4500** (8 bits)	D7	**SIE_IntStat**	1	SIE interrupts	0	None	0	R	
		D6	**EPrIntStat**	1	EPr interrupts	0	None	0	R	
		D5	**–**	–				–	–	0 when being read.
		D4	**FIFO_IntStat**	1	FIFO interrupts	0	None	0	R	
		D3–2	**–**	–				–	–	0 when being read.
		D1	**EP0IntStat**	1	EP0 interrupts	0	None	0	R	
		D0	**RcvEP0SETUP**	1	Receive EP0 SETUP	0	None	0	R(W)	

This register displays causes of interrupt having occurred in the USB function controller. This register has the bit indirectly showing causes of interrupt and the bit directly showing causes of interrupt.
The bit indirectly showing causes of interrupt can be traced to the bit directly showing causes of interrupt by reading the relevant status registers. The bit showing causes of interrupt is read-only, and is automatically cleared by clearing the bit directly showing causes of interrupt at the main source. The bits showing causes of interrupt are writable, and the causes of interrupt can be cleared by setting the relevant bits to 1. When the corresponding bits are enabled by the MainIntEnb register, setting the cause of interrupt to 1 asserts the #INT signal, and causes an interruption of the CPU. Clearing all relevant causes of interrupt negates the #INT signal.

D7 SIE_IntStat

Shows a cause of interrupt indirectly.
When the SIE_IntStat register has a cause of interrupt and the SIE_IntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1. Reading this bit is valid during snooze as well.

D6 EPrIntStat

Shows a cause of interrupt indirectly.
When the EPrIntStat register has a cause of interrupt and the EPrIntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

D5 Reserved

D4 FIFO_IntStat

Shows a cause of interrupt indirectly.
When the FIFO_IntStat register has a cause of interrupt and the FIFO_IntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

D[3:2] Reserved

D1 EP0IntStat

Shows a cause of interrupt indirectly.
When the EP0IntStat register has a cause of interrupt and the EP0IntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

D0 RcvEP0SETUP

Shows a cause of interrupt directly.
Set to 1 when the received data are set to the EP0Setup_0 to EP0Setup_7 after the set up stage has been completed. At the same time, the ForceSTALL bit, the ForceNAK bit and the ToggleStat bit of the EP0ControlIN and EP0ControlOUT registers are automatically set to 0 and 1, respectively.

0x4501: SIE_IntStat (SIE Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
SIE_IntStat (SIE interrupt status)	**0x4501** (8 bits)	D7	**VBUS_Changed**	1	VBUS is changed	0	None	0	R(W)	
		D6	**NonJ**	1	Detect non J state	0	None	0	R(W)	
		D5	**DetectReset**	1	Detect USB reset	0	None	0	R(W)	
		D4	**DetectSuspend**	1	Detect USB suspend	0	None	0	R(W)	
		D3	**RcvSOF**	1	Receive SOF token	0	None	0	R(W)	
		D2	**DetectJ**	1	Detect J state	0	None	0	R(W)	
		D1	**–**	–				–	–	0 when being read.
		D0	**SetAddressCmp**	1	AutoSetAddress complete	0	None	0	R(W)	

This register displays the interrupts related to SIE.

D7 VBUS_Changed

Shows a cause of interrupt directly.
When the condition of the VBUS terminal changes, this bit is set to 1.
Check the condition of the VBUS by the VBUS bit in the USB_Status register. If the VBUS is 0, it shows that the cable is pulled off. This bit is valid during snooze as well.

D6 NonJ

Shows a cause of interrupt directly.
Set to 1 when the status other than the J state is detected in the USB bus. This bit is valid when the InSUSPEND bit of the USB_Control register is set to 1.

D5 DetectReset

Shows a cause of interrupt directly.
Set to 1 when the reset state of the USB is detected. This reset detection is valid when the ActiveUSB bit of the USB_Control register is set to 1.
When the AutoNegotiation function is not used, if this bit is set to 1, set to the DisBusDetect bit of the USB_Control register to 1, not to detect the succeeding reset wrongly by disabling detection of the reset/suspend state. Set the DisBusDetect bit to 0 (to be cleared) after completing the process for reset, to enable the reset/suspend state detection.
Refer to the item on the EnAutoNego bit of the USB_Control register, for the AutoNegotiation function.

D4 DetectSuspend

Shows a cause of interrupt directly.
Set to 1 when the suspend state of the USB is detected.
After detecting the USB suspend state, setting the USBSNZ bit of the USB Wait Control Register (0x4909) to 1 enables the IC to enter the snooze mode (to stop the built-in PLL oscillation).

D3 RcvSOF

Shows a cause of interrupt directly.
Set to 1 when the SOF token is received.

D2 DetectJ

Shows a cause of interrupt directly.
Set to 1 when the J-state is detected.

D1 Reserved

D0 SetAddressCmp

Shows a cause of interrupt directly.
When the AutoSetAddress function (refer to the USB_Address register) ends normally, this bit is set to 1. The case when AutoSetAddress function ends normally is that when ACK is received during IN transaction.

0x4502: EPrIntStat (EPr Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPrIntStat	**0x4502**	D7–4	**–**	–				–	–	0 when being read.
(EPr interrupt	(8 bits)	D3	**EPdIntStat**	1	EPd interrupt	0	None	0	R	
status)		D2	**EPcIntStat**	1	EPc interrupt	0	None	0	R	
		D1	**EPbIntStat**	1	EPb interrupt	0	None	0	R	
		D0	**EPaIntStat**	1	EPa interrupt	0	None	0	R	

D[7:4] Reserved

D3 EPdIntStat

Shows a cause of interrupt indirectly.
When the EPdIntStat register has a cause of interrupt and the EPdIntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

D2 EPcIntStat

Shows a cause of interrupt indirectly.
When the EPcIntStat register has a cause of interrupt and the EPcIntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

D1 EPbIntStat

Shows a cause of interrupt indirectly.
When the EPbIntStat register has a cause of interrupt and the EPbIntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

D0 EPaIntStat

Shows a cause of interrupt indirectly.
When the EPaIntStat register has a cause of interrupt and the EPaIntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

0x4504: FIFO_IntStat (FIFO Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
FIFO_IntStat	**0x4504**	D7	**DescriptorCmp**	1	Descriptor complete	0	None	0	R(W)	
(FIFO interrupt status)	(8 bits)	D6–0	–	–				–	–	0 when being read.

This register displays the interrupt status of the FIFO.

D7 DescriptorCmp

Shows a cause of interrupt directly.

Set to 1 when as many data as specified in the DescSize register have been replied in the Description Reply function.

And the OUT_TranNAK bit of the EP0IntStat register is set to 1 as well as this bit, when changing to the status stage takes place (the OUT token is received) before sending data up to the quantity specified in the DescSize register.

D[6:0] Reserved

0x4507: EP0IntStat (EP0 Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0IntStat (EP0 interrupt status)	0x4507 (8 bits)	D7–6	–	–				–	–	0 when being read.
		D5	IN_TranACK	1	In transaction ACK	0	None	0	R(W)	
		D4	OUT_TranACK	1	Out transaction ACK	0	None	0	R(W)	
		D3	IN_TranNAK	1	In transaction NAK	0	None	0	R(W)	
		D2	OUT_TranNAK	1	Out transaction NAK	0	None	0	R(W)	
		D1	IN_TranErr	1	In transaction error	0	None	0	R(W)	
		D0	OUT_TranErr	1	Out transaction error	0	None	0	R(W)	

This register displays the interrupt status of the endpoint EP0.

D[7:6] Reserved

D5 IN_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is received in the IN transaction.

D4 OUT_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is replied in the OUT transaction.

D3 IN_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the IN transaction.

D2 OUT_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the OUT transaction.

D1 IN_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the IN transaction, when an error occurred in the packet or when the handshake is failed in Time-Out.

D0 OUT_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the OUT transaction or when an error occurred in the packet.

0x4508: EPaIntStat (EPa Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaIntStat	**0x4508**	D7	**–**	–				–	–	0 when being read.
(EPa interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	

This register displays the interrupt status of the endpoint EPa.

D7 Reserved

D6 OUT_ShortACK

Shows a cause of interrupt directly.
Set to 1 when a short packet is received and ACK is replied in the OUT transaction, OUT_TranACK and this bits are set to 1 at the same time.

D5 IN_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is received in the IN transaction.

D4 OUT_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is replied in the OUT transaction.

D3 IN_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the IN transaction.

D2 OUT_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the OUT transaction.

D1 IN_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the IN transaction, when an error occurred in the packet or when the handshake is failed in Time-Out.

D0 OUT_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the OUT transaction or when an error occurred in the packet.

0x4509: EPbIntStat (EPb Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbIntStat	**0x4509**	D7	**–**	–				–	–	0 when being read.
(EPb interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	

This register displays the interrupt status of the endpoint EPb.

D7 Reserved

D6 OUT_ShortACK

Shows a cause of interrupt directly.
Set to 1 when a short packet is received and ACK is replied in the OUT transaction, OUT_TranACK and this bits are set to 1 at the same time.

D5 IN_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is received in the IN transaction.

D4 OUT_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is replied in the OUT transaction.

D3 IN_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the IN transaction.

D2 OUT_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the OUT transaction.

D1 IN_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the IN transaction, when an error occurred in the packet or when the handshake is failed in Time-Out.

D0 OUT_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the OUT transaction or when an error occurred in the packet.

0x450a: EPcIntStat (EPc Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcIntStat	**0x450a**	D7	**–**	–				–	–	0 when being read.
(EPc interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	

This register displays the interrupt status of the endpoint EPc.

D7 Reserved

D6 OUT_ShortACK

Shows a cause of interrupt directly.
Set to 1 when a short packet is received and ACK is replied in the OUT transaction, OUT_TranACK and this bits are set to 1 at the same time.

D5 IN_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is received in the IN transaction.

D4 OUT_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is replied in the OUT transaction.

D3 IN_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the IN transaction.

D2 OUT_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the OUT transaction.

D1 IN_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the IN transaction, when an error occurred in the packet or when the handshake is failed in Time-Out.

D0 OUT_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the OUT transaction or when an error occurred in the packet.

0x450b: EPdIntStat (EPd Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdIntStat (EPd interrupt status)	0x450b (8 bits)	D7	–	–				–	–	0 when being read.
		D6	OUT_ShortACK	1	Out short packet ACK	0	None	0	R(W)	
		D5	IN_TranACK	1	In transaction ACK	0	None	0	R(W)	
		D4	OUT_TranACK	1	Out transaction ACK	0	None	0	R(W)	
		D3	IN_TranNAK	1	In transaction NAK	0	None	0	R(W)	
		D2	OUT_TranNAK	1	Out transaction NAK	0	None	0	R(W)	
		D1	IN_TranErr	1	In transaction error	0	None	0	R(W)	
		D0	OUT_TranErr	1	Out transaction error	0	None	0	R(W)	

This register displays the interrupt status of the endpoint EPd.

D7 Reserved

D6 OUT_ShortACK

Shows a cause of interrupt directly.
Set to 1 when a short packet is received and ACK is replied in the OUT transaction, OUT_TranACK and this bits are set to 1 at the same time.

D5 IN_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is received in the IN transaction.

D4 OUT_TranACK

Shows a cause of interrupt directly.
Set to 1 when ACK is replied in the OUT transaction.

D3 IN_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the IN transaction.

D2 OUT_TranNAK

Shows a cause of interrupt directly.
Set to 1 when NAK is replied in the OUT transaction.

D1 IN_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the IN transaction, when an error occurred in the packet or when the handshake is failed in Time-Out.

D0 OUT_TranErr

Shows a cause of interrupt directly.
Set to 1 when STALL is replied in the OUT transaction or when an error occurred in the packet.

0x4510: MainIntEnb (Main Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
MainIntEnb (Main interrupt enable)	**0x4510** (8 bits)	D7	**EnSIE_IntStat**	1	Enabled	0	Disabled	0	R/W	
		D6	**EnEPrIntStat**					0	R/W	
		D5	**–**	–				–	–	0 when being read.
		D4	**EnFIFO_IntStat**	1	Enabled	0	Disabled	0	R/W	
		D3–2	**–**	–				–	–	0 when being read.
		D1	**EnEP0IntStat**	1	Enabled	0	Disabled	0	R/W	
		D0	**EnRcvEP0SETUP**					0	R/W	

This register enables/disables assertion of the interrupt signal (#INT) with the cause of interrupt of the MainIntStat register. Setting the corresponding bit to 1 enables interrupt. EnSIE_IntStat bit is valid during snooze as well.

0x4511: SIE_IntEnb (SIE Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
SIE_IntEnb (SIE interrupt enable)	**0x4511** (8 bits)	D7	**EnVBUS_Changed**	1	Enabled	0	Disabled	0	R/W	
		D6	**EnNonJ**					0	R/W	
		D5	**EnDetectReset**					0	R/W	
		D4	**EnDetectSuspend**					0	R/W	
		D3	**EnRcvSOF**					0	R/W	
		D2	**EnDetectJ**					0	R/W	
		D1	**–**	–				–	–	0 when being read.
		D0	**EnSetAddressCmp**	1	Enabled	0	Disabled	0	R/W	

This register enables/disables assertion of the SIE_IntStat bit of the MainIntStat register with the cause of interrupt of the SIE_IntStat register. EnVBUS_Changed and EnNonJ bits are valid during snooze as well.

0x4512: EPrIntEnb (EPr Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPrIntEnb	**0x4512**	D7–4	**–**	–				–	–	0 when being read.
(EPr interrupt	(8 bits)	D3	**EnEPdIntStat**	1	Enabled	0	Disabled	0	R/W	
enable)		D2	**EnEPcIntStat**					0	R/W	
		D1	**EnEPbIntStat**					0	R/W	
		D0	**EnEPaIntStat**					0	R/W	

This register enables/disables assertion of the EPrIntStat bit of the MainIntStat register with the cause of interrupt of the EPrIntStat register.

0x4514: FIFO_IntEnb (FIFO Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
FIFO_IntEnb	**0x4514**	D7	**EnDescriptorCmp**	1	Enabled	0	Disabled	0	R/W	
(FIFO interrupt enable)	(8 bits)	D6–0	**–**	–				–	–	0 when being read.

This register enables/disables assertion of the FIFO_IntStat bit of the MainIntStat register with the cause of interrupt of the FIFO_IntStat register.

0x4517: EP0IntEnb (EP0 Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0IntEnb	**0x4517**	D7–6	**–**	–				–	–	0 when being read.
(EP0 interrupt	(8 bits)	D5	**EnIN_TranACK**	1	Enabled	0	Disabled	0	R/W	
enable)		D4	**EnOUT_TranACK**					0	R/W	
		D3	**EnIN_TranNAK**					0	R/W	
		D2	**EnOUT_TranNAK**					0	R/W	
		D1	**EnIN_TranErr**					0	R/W	
		D0	**EnOUT_TranErr**					0	R/W	

This register enables/disables assertion of the EP0IntStat bit of the MainIntStat register with the cause of interrupt of the EP0IntStat register.

0x4518: EPaIntEnb (EPa Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaIntEnb	**0x4518**	D7	**–**	–				–	–	0 when being read.
(EPa interrupt	(8 bits)	D6	**EnOUT_ShortACK**	1	Enabled	0	Disabled	0	R/W	
enable)		D5	**EnIN_TranACK**					0	R/W	
		D4	**EnOUT_TranACK**					0	R/W	
		D3	**EnIN_TranNAK**					0	R/W	
		D2	**EnOUT_TranNAK**					0	R/W	
		D1	**EnIN_TranErr**					0	R/W	
		D0	**EnOUT_TranErr**					0	R/W	

This register enables/disables assertion of the EPaIntStat bit of the EPrIntStat register with the cause of interrupt of the EPaIntStat register.

0x4519: EPbIntEnb (EPb Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbIntEnb	**0x4519**	D7	**–**	–				–	–	0 when being read.
(EPb interrupt	(8 bits)	D6	**EnOUT_ShortACK**	1	Enabled	0	Disabled	0	R/W	
enable)		D5	**EnIN_TranACK**					0	R/W	
		D4	**EnOUT_TranACK**					0	R/W	
		D3	**EnIN_TranNAK**					0	R/W	
		D2	**EnOUT_TranNAK**					0	R/W	
		D1	**EnIN_TranErr**					0	R/W	
		D0	**EnOUT_TranErr**					0	R/W	

This register enables/disables assertion of the EPbIntStat bit of the EPrIntStat register with the cause of interrupt of the EPbIntStat register.

0x451a: EPcIntEnb (EPc Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcIntEnb	**0x451a**	D7	–	–				–	–	0 when being read.
(EPc interrupt	(8 bits)	D6	**EnOUT_ShortACK**	1	Enabled	0	Disabled	0	R/W	
enable)		D5	**EnIN_TranACK**					0	R/W	
		D4	**EnOUT_TranACK**					0	R/W	
		D3	**EnIN_TranNAK**					0	R/W	
		D2	**EnOUT_TranNAK**					0	R/W	
		D1	**EnIN_TranErr**					0	R/W	
		D0	**EnOUT_TranErr**					0	R/W	

This register enables/disables assertion of the EPcIntStat bit of the EPrIntStat register with the cause of interrupt of the EPcIntStat register.

0x451b: EPdIntEnb (EPd Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdIntEnb	**0x451b**	D7	**–**	–				–	–	0 when being read.
(EPd interrupt	(8 bits)	D6	**EnOUT_ShortACK**	1	Enabled	0	Disabled	0	R/W	
enable)		D5	**EnIN_TranACK**					0	R/W	
		D4	**EnOUT_TranACK**					0	R/W	
		D3	**EnIN_TranNAK**					0	R/W	
		D2	**EnOUT_TranNAK**					0	R/W	
		D1	**EnIN_TranErr**					0	R/W	
		D0	**EnOUT_TranErr**					0	R/W	

This register enables/disables assertion of the EPdIntStat bit of the EPrIntStat register with the cause of interrupt of the EPdIntStat register.

0x4520: RevisionNum (Revision Number)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
RevisionNum	**0x4520**	D7	**RevisionNum[7]**	Revision number	0	R	
(Revision	(8 bits)	D6	**RevisionNum[6]**	(0x12)	0		
number)		D5	**RevisionNum[5]**		0		
		D4	**RevisionNum[4]**		1		
		D3	**RevisionNum[3]**		0		
		D2	**RevisionNum[2]**		0		
		D1	**RevisionNum[1]**		1		
		D0	**RevisionNum[0]**		0		

This register shows the revision number of the USB function controller. This register is valid during snooze as well.

0x4521: USB_Control (USB Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
USB_Control (USB control register)	**0x4521** (8 bits)	D7	**DisBusDetect**	1	Disable bus detect	0	Enable bus detect	0	R/W	
		D6	**EnAutoNego**	1	Enable auto negotiation	0	Disable auto negotiation	0	R/W	
		D5	**InSUSPEND**	1	Monitor NonJ	0	Do nothing	0	R/W	
		D4	**StartDetectJ**	1	Start J-state detection	0	Do nothing	0	R/W	
		D3	**SendWakeup**	1	Send remote wakeup signal	0	Do nothing	0	R/W	
		D2–1	**–**	–				–	–	0 when being read.
		D0	**ActiveUSB**	1	Activate USB	0	Disactivate USB	0	R/W	

The operation setting is done for the USB.

D7 DisBusDetect

Setting this bit to 1 disables the automatic detection of the USB reset/suspend state.
When this bit is set to 0 (to be cleared), activities on the USB bus is monitored to detect the reset/suspend state.
If the bus activities cannot be detected within 3 ms, the USB is determined to be suspend state. And if "SE0" longer than 2.5 microseconds is detected, the USB is determined to be reset state, and then the relevant cause of interrupt (DetectReset, DetectSuspend) is set.
If the DetectReset or the DetectSuspend bit is set to 1, set the DisBusDetect bit to 1 to disable detection when the reset/suspend state is continued.
When using the Auto Negotiation function, do not set this bit to 1.

D6 EnAutoNego

This bit enables the Auto Negotiation function. The Auto Negotiation function automates the work sequence to be done after detecting the reset, from the end of the speed negotiation to determination of the speed mode. Refer to the section describing operations for details of the Auto Negotiation.

D5 InSUSPEND

This bit enables the detection of the NonJ state. If the USB suspend state is detected and f/w is prepared. set this bit to 1. To return from the suspended state, set this bit to 0 (to be cleared).
The NonJ state can be detected only when this bit is set. If the Snooze function is not be used when the USB goes into the suspend state, set this bit.
Refer to description on operations for how to use the Auto Negotiation function.

D4 StartDetectJ

This bit enables the detection of the J state. After setting this bit and J-state is coming, DetectJ interrupt is set when EnDetectJ is set.

D3 SendWakeup

Setting this bit to 1 outputs the RemoteWakeup signal (K) to the USB port.
Within the time between 1 ms and 15 ms after starting to send the RemoteWakeup signal, set this bit to 0 (to be cleared) to stop sending the signals.

D[2:1] Reserved

D0 ActiveUSB

Since this bit is set to 0 (to be cleared) after hardware reset, all USB functions are stopped. The operation as a USB will be enabled by setting this bit to 1 after completing the setting of this IC.

0x4522: USB_Status (USB Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
USB_Status (USB status register)	**0x4522** (8 bits)	D7	**VBUS**	1	VBUS=High	0	VBUS=Low	X	R	
		D6	**FS**	1	FS mode (fixed)	0	–	1	R	
		D5–2	**–**	–				–	–	0 when being read.
		D1	**LineState[1]**	LineState[1:0]			DP/DM	X	R	
		D0	**LineState[0]**	1	1		SE1	X		
				1	0		K			
				0	1		J			
				0	0		SE0			

This register displays the status related to the USB.
This register is valid during snooze as well.

D7 VBUS

This bit displays the status of the USBVBUS pin.

D6 FS

Returns always 1 (FS mode).

D[5:2] Reserved

D[1:0] LineState[1:0]

Shows the signal status on the USB cable.
Shows the value received by the FS receiver of the DP/DM.

LineState

LineState[1]	LineState[0]	DP/DM
1	1	SE1
1	0	K
0	1	J
0	0	SE0

0x4523: XcvrControl (Xcvr Control)

Register name	Address	Bit	Name	Setting			Init.	R/W	Remarks
XcvrControl	**0x4523**	D7	**RpuEnb**	1	Enable pull-up	0 Disable pull-up	1	R/W	
(Xcvr control	(8 bits)	D6–2	**–**	–			–	–	0 when being read.
register)		D1	**OpMode[1]**	OpMode[1:0]		Operation mode	0	R/W	
		D0	**OpMode[0]**	1	1	reserved	1		
				1	0	Disable bit stuffing and NRZI encoding			
				0	1	Non-driving			
				0	0	Normal operation			

The operation setting is done for the Transceiver macro.

D7 RpuEnb

This bit enables the D+ pull-up resistor.

D[6:2] Reserved

D[1:0] OpMode

This bit sets the operation mode of the Transceiver macro.
This bit needs not be set up normally, excluding when the USB cable is pulled off (*) and during the test mode.

OpMode

OpMode[1]	OpMode[0]	Operation mode
1	1	Reserved
1	0	Disable bit stuffing and NRZI encoding
0	1	Non-driving
0	0	Normal operation

* When the USB cable is pulled off, it is recommended to set this register to 0x01.

0x4524: USB_Test (USB Test)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
USB_Test (USB test)	0x4524 (8 bits)	D7	**EnUSB_Test**	1	Enable USB test	0	Do nothing	0	R/W	
		D6–4	–	–				–	–	0 when being read.
		D3	**Test_SE0_NAK**	1	Test_SE0_NAK	0	Do nothing	0	R/W	
		D2	**Test_J**	1	Test_J	0	Do nothing	0	R/W	
		D1	**Test_K**	1	Test_K	0	Do nothing	0	R/W	
		D0	**Test_Packet**	1	Test_Packet	0	Do nothing	0	R/W	

The operation setting is done in this register for the USB test mode. Set the bit corresponding to the test mode specified by the SetFeature request, and after completing the status stage, set the EnUSB_Test bit to 1 and perform the test mode operation defined by the USB standard.

D7 EnUSB_Test

When this bit is set to 1, if one of the lower order 4 bits in the USB_Test register is set to 1, the IC will go into the test mode corresponding to the bit. When performing the test mode, the DisBusDetect bit of the USB_Control register must be set to 1 not to detect the USB suspend and the reset before performing the test. In addition, set the EnAutoNego bit of the USB_Control register to 0 (to be cleared) to disable the Auto Negotiation.

Note that the change to the test mode must be done after completing the status stage for the SetFeature request.

D[6:4] Reserved

D3 Test_SE0_NAK

By setting this bit to 1 and the EnUSB_Test bit to 1, the Test_SE0_NAK test mode can start.

D2 Test_J

By setting this bit to 1 and the EnUSB_Test bit to 1, the Test_J test mode can start. In this test mode, before EnUSB_Test bit is set to 1, set OpMode to 10 (Disable Bit stuffing and NRZI encoding).

D1 Test_K

By setting this bit to 1 and the EnUSB_Test bit to 1, the Test_K test mode can start. In this test mode, before EnUSB_Test bit is set to 1, set OpMode to 10 (Disable Bit stuffing and NRZI encoding).

D0 Test_Packet

By setting this bit to 1, the Test_Packet test mode can start.

Since this test mode uses the endpoint EPc, set the followings.

(1) Set the MaxPacketSize of the endpoint EPc to 64 or more, the direction of transfer to IN and the EndPointNumber to 0xf to make the endpoint be ready to use. And allocate the FIFO of the endpoint EPc for 64 bytes or more.
(2) Do not overlap the above setting with the settings of the endpoints EPa and EPb.
Or clear the EPaConfig_0.EnEndPoint bit and EPbConfig_0.EnEndPoint bit.
(3) Clear the FIFO of the EPc and write data for the following test packet into this FIFO.
(4) Set the EnIN_TranErr of the EPcIntEnb register to 0 (clear this bit).
IN_TranErr status is set to 1 at every time the Test Packet transmission completes.

The data to write into the FIFO in the packet transmission test mode are the following 53 bytes.

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0xaa, 0xaa, 0xaa, 0xaa, 0xaa, 0xaa, 0xaa,
0xaa, 0xee, 0xee, 0xee, 0xee, 0xee, 0xee, 0xee,
0xee, 0xfe, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0x7f, 0xbf, 0xdf,
0xef, 0xf7, 0xfb, 0xfd, 0xfc, 0x7e, 0xbf, 0xdf,
0xef, 0xf7, 0xfb, 0xfd, 0x7e
```

Since the SIE adds the PID and CRC to the test packet when sending it, the data to write into the FIFO are from “the data after the DATA 0 PID” to “the data before the CRC16” that are described as the test packet data in the USB standard Rev.2.0. (Note that Test Packet is defined only HS mode in USB specification.)

0x4525: EPnControl (Endpoint Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPnControl (Endpoint control)	**0x4525** (8 bits)	D7	**AllForceNAK**	1	Set all ForceNAK	0	Do nothing	0	W	0 when being read.
		D6	**EPrForceSTALL**	1	Set EP's ForceSTALL	0	Do nothing	0	W	
		D5	**AllFIFO_Clr**	1	Clear all FIFO	0	Do nothing	0	W	
		D4–1	**–**	–				–	–	
		D0	**EP0FIFO_Clr**	1	Clear EP0 FIFO	0	Do nothing	0	W	

This register sets operations of entire endpoints, and display them.

D7 AllForceNAK

Set the ForceNAK bit of all endpoints to 1.

D6 EPrForceSTALL

Set the ForceSTALL bit of EPa, EPb, EPc and EPd endpoints to 1.

D5 AllFIFO_Clr

Clear the FIFOs of all endpoints. After setting the area of the respective endpoints, be sure to set this bit to 1 to clear the FIFOs of all endpoints. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

D[4:1] Reserved

D0 EP0FIFO_Clr

Clear the FIFO of the endpoint EP0. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

0x4526: EPrFIFO_Clr (EPr FIFO Clear)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPrFIFO_Clr (EPr FIFO clear)	0x4526 (8 bits)	D7–4	–	–				–	–	0 when being read.
		D3	EPdFIFO_Clr	1	Clear EPd FIFO	0	Do nothing	0	W	
		D2	EPcFIFO_Clr	1	Clear EPc FIFO	0	Do nothing	0	W	
		D1	EPbFIFO_Clr	1	Clear EPb FIFO	0	Do nothing	0	W	
		D0	EPaFIFO_Clr	1	Clear EPa FIFO	0	Do nothing	0	W	

This register clears the FIFO of the endpoints.

D[7:4] Reserved

D3 EPdFIFO_Clr

Clear the FIFO of the endpoint EPd. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

D2 EPcFIFO_Clr

Clear the FIFO of the endpoint EPc. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

D1 EPbFIFO_Clr

Clear the FIFO of the endpoint EPb. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

D0 EPaFIFO_Clr

Clear the FIFO of the endpoint EPa. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

0x452e: FrameNumber_H (Frame Number HIGH)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
FrameNumber_H (Frame number high)	**0x452e** (8 bits)	D7	**FnInvalid**	1 Invalid frame number 0 Valid frame number	1	R	
		D6–3	**–**	–	–	–	0 when being read.
		D2	**FrameNumber[10]**	Frame number high	0	R	
		D1	**FrameNumber[9]**		0		
		D0	**FrameNumber[8]**		0		

This register displays the USB frame number that is updated every time the SOF token is received. When frame numbers are acquired, the FrameNumber_H and the FrameNumber_L registers must be accessed as a pair. When accessing them, access the FrameNumber_H register first.

D7 FnInvalid

When an error occurs in the received SOF packet, this bit is set to 1.

D[6:3] Reserved

D[2:0] FrameNumber[10:8]

The upper order 3 bits in the FrameNumber field of the received SOF packet are stored in these bits.

0x452f: FrameNumber_L (Frame Number LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
FrameNumber_L	**0x452f**	D7	**FrameNumber[7]**	Frame number low	0	R	
(Frame number	(8 bits)	D6	**FrameNumber[6]**		0		
low)		D5	**FrameNumber[5]**		0		
		D4	**FrameNumber[4]**		0		
		D3	**FrameNumber[3]**		0		
		D2	**FrameNumber[2]**		0		
		D1	**FrameNumber[1]**		0		
		D0	**FrameNumber[0]**		0		

D[7:0] FrameNumber[7:0]

The lower order 8 bits in the FrameNumber field of the received SOF packet are stored in these bits.

0x4530–0x4537: EP0Setup_0 (EP0 Setup 0)–EP0Setup_7 (EP0 Setup 7)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EP0Setup_0	**0x4530**	D7	**EP0Setup_n[7]**	Endpoint 0 set-up data 0	0	R	
(EP0 set-up 0)	**|**	D6	**EP0Setup_n[6]**	**|**	0		
|	**0x4537**	D5	**EP0Setup_n[5]**	Endpoint 0 set-up data 7	0		
EP0Setup_7	(8 bits)	D4	**EP0Setup_n[4]**		0		
(EP0 set-up 7)		D3	**EP0Setup_n[3]**		0		
		D2	**EP0Setup_n[2]**		0		
		D1	**EP0Setup_n[1]**		0		
		D0	**EP0Setup_n[0]**		0		

Eight-byte data received at the endpoint EP0 setup stage are stored from the EP0Setup_0 sequentially.

0x4530: EP0Setup_0

BmRequestType is set.

0x4531: EP0Setup_1

BRequest is set.

0x4532: EP0Setup_2

The lower order 8 bits in Wvalue are set.

0x4533: EP0Setup_3

The upper order 8 bits in Wvalue are set.

0x4534: EP0Setup_4

The lower order 8 bits in WIndex are set.

0x4535: EP0Setup_5

The upper order 8 bits in WIndex are set.

0x4536: EP0Setup_6

The lower order 8 bits in WLength are set.

0x4537: EP0Setup_7

The upper order 8 bits in WLength are set.

0x4538: USB_Address (USB Address)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
USB_Address	0x4538	D7	AutoSetAddress	1 Auto set address 0 Do nothing	0	R/W	
(USB address)	(8 bits)	D6	USB_Address[6]	USB address	0	R/W	
		D5	USB_Address[5]		0		
		D4	USB_Address[4]		0		
		D3	USB_Address[3]		0		
		D2	USB_Address[2]		0		
		D1	USB_Address[1]		0		
		D0	USB_Address[0]		0		

This register sets up the USB address.

D7 AutoSetAddress

Sets up the USB Address automatically. If this bit is set to 1 after receiving the SetAddress request and before implementing the status stage, the address received by the SetAddress request will be written into the USB_Address register when the status stage completes.

The processing procedure of the SetAddress request using this function is as follows.

(1) The SETUP transaction of the SetAddress request completes.
The RcvEP0SETUP bit of the MainIntStat register is set to 1. Read the EP0Setup_0–7 registers and interpret the request.
(2) Set the AutoSetAddress bit.
(3) Set the INxOUT bit of the EP0Control register.
(4) Clear the ForceNAK bit of the EP0ControlIN register, and set the EnShortPkt bit.
(5) Wait for the end of the status stage.
The SetAddressCmp bit of the SIE_IntStat register is set to 1.

D[6:0] USB_Address

These bits set up the USB address.

The USB address is written automatically by the AutoSetAddress function. Or it can be written.

0x4539: EP0Control (EP0 Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0Control (EP0 control)	0x4539 (8 bits)	D7	INxOUT	1	IN	0	OUT	0	R/W	
		D6–1	–	–				–	–	0 when being read.
		D0	ReplyDescriptor	1	Reply descriptor	0	Do nothing	0	W	

This register sets up the endpoint EP0.

D7 INxOUT

Set the transfer direction of the endpoint EP0.

Judging from the request received at the setup stage, set a value in this bit.

If the data stage exists, set the transfer direction at the data stage into this bit. As the setup of the ForceNAK bits of the EP0ControlIN and EP0ControlOUT registers completes when the setup stage completes, clear them during execution of the data stage or the status stage.

After the data stage is completed, set this bit again conforming to the direction of the status stage. When the transfer direction of the data stage is IN, the transfer direction of the status stage is OUT. Therefore, set this bit to 0. When the transfer direction of the data stage is OUT, or there is no data stage, the transfer direction of the status stage is IN. Therefore, clear the FIFO of the endpoint EP0, and set this bit to 1.

For the IN or OUT transactions which have a transfer direction different from that of this bit, NAK response is done. However, if the ForceSTALL bit of the EP0ControlIN or EP0ControlOUT register with the transaction direction corresponding to the above one, is set, the STALL response will be done.

D[6:1] Reserved

D0 ReplyDescriptor

Executes the Descriptor reply function.

If this bit is set to 1, this bit replies as much Descriptor data as specified as MaxPacketSize from the FIFO, responding to the IN transaction of the endpoint EP0. The Descriptor data start from the address specified in the DescAdrs_H, L register, and its data size is specified in the DescSize_H, L register. Since these setting values are updated during execution of the Descriptor reply function, set these setting values every time setting the ReplyDescriptor bit.

In every transaction, the DescAdrs_H, L register is incremented as many as the number of data that were sent, while the DescSize_H, L register is decremented as many as the number of data that were sent.

When the data transmission ends after sending as many data as specified in the DescSize_H, L or when a transaction other than the IN transaction is done, the Descriptor reply function ends, the ReplyDescriptor bit is set to 0 (to be cleared) and the IN_TranACK bit of the EPnIntStat register is set to 1.

Refer to the section describing operations, for details.

0x453a: EP0ControlIN (EP0 Control IN)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0ControlIN (EP0 control IN)	0x453a (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnShortPkt	1	Enable short packet	0	Do nothing	0	R/W	
		D5	–	–				–	–	0 when being read.
		D4	ToggleStat	Toggle sequence bit				0	R	
		D3	ToggleSet	1	Set toggle sequence bit	0	Do nothing	0	R/W	
		D2	ToggleClr	1	Clear toggle sequence bit	0	Do nothing	0	R/W	
		D1	ForceNAK	1	Force NAK	0	Do nothing	0	R/W	
		D0	ForceSTALL	1	Force STALL	0	Do nothing	0	R/W	

This register sets the operations related to the IN transaction of the endpoint EP0 and displays their status.

D7 Reserved

D6 EnShortPkt

Setting this bit to 1 enables to send the data within the FIFO that is less than the quantity specified for the MaxPacketSize, as a short packet for the IN transaction of the endpoint EP0. When the IN transaction that transmitted short packets completes, this bit is automatically set to 0 (to be cleared). When a packet of the max packet size is transmitted, this bit is not cleared.

If this bit is set to 1 when the FIFO has no data, a zero-length packet can be transmitted for the IN token from the host. If the data is written into the FIFO that is in the transmission process with the packet to which this bit is set, that data may be included in transmission. Therefore, do not write into the FIFO until the packet transmission completes and this bit is cleared.

D5 Reserved

D4 ToggleStat

Shows the status of the toggle sequence bit in the IN transaction of the endpoint EP0.

D3 ToggleSet

Sets the toggle sequence bit in the IN transaction of the endpoint EP0, to 1.

D2 ToggleClr

Sets the toggle sequence bit in the IN transaction of the endpoint EP0, to 0 (clear).

D1 ForceNAK

If this bit is set to 1, the NAK response is done for the IN transaction of the endpoint EP0, regardless of the FIFO data quantity.

When the RcvEP0SETUP bit of the MainIntStat register is set to 1 after completion of the setup stage, this bit is set to 1, and this bit cannot be set to 0 (to be cleared) as long as the RcvEP0SETUP bit is 1.

When the IN transaction that transmitted short packets completes, this bit is set to 1.

When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

D0 ForceSTALL

If this bit is set to 1, the STALL response is done for the IN transaction of the endpoint EP0. This bit has a priority over the setting of the ForceNAK bit.

When the RcvEP0SETUP bit of the MainIntStat register is set to 1 after completion of the setup stage, this bit is set to 0 (to be cleared), and this bit cannot be set to 1 as long as the RcvEP0SETUP bit is 1.

When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

0x453b: EP0ControlOUT (EP0 Control OUT)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0ControlOUT (EP0 control OUT)	0x453b (8 bits)	D7	**AutoForceNAK**	1	Auto force NAK	0	Do nothing	0	R/W	
		D6–5	–	–				–	–	0 when being read.
		D4	**ToggleStat**	Toggle sequence bit				0	R	
		D3	**ToggleSet**	1	Set toggle sequence bit	0	Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1	Clear toggle sequence bit	0	Do nothing	0	W	
		D1	**ForceNAK**	1	Force NAK	0	Do nothing	0	R/W	
		D0	**ForceSTALL**	1	Force STALL	0	Do nothing	0	R/W	

This register sets the operations related to the OUT transaction of the endpoint EP0 and displays their status.

D7 AutoForceNAK

Sets the ForceNAK bit of this register to 1 when the OUT transaction of the endpoint EP0 completes normally.

D[6:5] Reserved

D4 ToggleStat

Shows the status of the toggle sequence bit in the OUT transaction of the endpoint EP0.

D3 ToggleSet

Sets the toggle sequence bit in the OUT transaction of the endpoint EP0, to 1.

D2 ToggleClr

Sets the toggle sequence bit in the OUT transaction of the endpoint EP0, to 0 (clear).

D1 ForceNAK

If this bit is set to 1, the NAK response is done for the OUT transaction of the endpoint EP0, regardless of the FIFO space capacity.
When the RcvEP0SETUP bit of the MainIntStat register is set to 1 after completion of the setup stage, this bit is set to 1, and this bit cannot be set to 0 (to be cleared) as long as the RcvEP0SETUP bit is 1. When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

D0 ForceSTALL

If this bit is set to 1, the STALL response is done for the OUT transaction of the endpoint EP0. This bit has a priority over the setting of the ForceNAK bit.
When the RcvEP0SETUP bit of the MainIntStat register is set to 1 after completion of the setup stage, this bit is set to 0 (to be cleared), and this bit cannot be set to 1 as long as the RcvEP0SETUP bit is 1.
When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

0x453f: EP0MaxSize (EP0 Max Packet Size)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EP0MaxSize	**0x453f**	D7	**–**	–	–	–	0 when being read.
(EP0 max	(8 bits)	D6	**EP0MaxSize[6]**	Endpoint EP0 max packet size	0	R/W	
packet size)		D5	**EP0MaxSize[5]**		0		
		D4	**EP0MaxSize[4]**		0		
		D3	**EP0MaxSize[3]**		1		
		D2–0	**–**	–	–	–	0 when being read.

D7 Reserved

D[6:3] EP0MaxSize[6:3]

This register sets the MaxPacketSize of the endpoint EP0.
The size of this endpoint can be set to 8, 16, 32 or 64 bytes.

D[2:0] Reserved

0x4540: EPaControl (EPa Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaControl (EPa control)	0x4540 (8 bits)	D7	**AutoForceNAK**	1	Auto force NAK	0	Do nothing	0	R/W	
		D6	**EnShortPkt**	1	Enable short packet	0	Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1	Disable auto force	0	Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit				0	R	
		D3	**ToggleSet**	1	Set toggle sequence bit	0	Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1	Clear toggle sequence bit	0	Do nothing	0	W	
		D1	**ForceNAK**	1	Force NAK	0	Do nothing	0	R/W	
		D0	**ForceSTALL**	1	Force STALL	0	Do nothing	0	R/W	

This register sets operations of the endpoint EPa.

D7 AutoForceNAK

Sets the ForceNAK bit of this register to 1 when the transaction of the endpoint EPa completes normally.

D6 EnShortPkt

Setting this bit to 1 enables to send the data within the FIFO that is less than the quantity specified for the MaxPacketSize, as a short packet for the IN transaction of the endpoint EPa. When the IN transaction that transmitted short packets completes, this bit is automatically set to 0 (to be cleared). When a packet of the max packet size is transmitted, this bit is not cleared.

If this bit is set to 1 when the FIFO has no data, a zero-length packet can be transmitted for the IN token from the host. If the data is written into the FIFO that is in the transmission process with the packet to which this bit is set, that data may be included in transmission. Therefore, do not write into the FIFO until the packet transmission completes and this bit is cleared.

D5 DisAF_NAK_Short

When this bit is set to 0 (default setting) and the packet that was received at normal completion time of the OUT transaction is a short packet, the ForceNAK bit is automatically set to 1. When this bit is set to 1, this function is disabled.

When the AutoForceNAK bit is set to 1, the AutoForceNAK bit has a priority.

D4 ToggleStat

Shows the status of the toggle sequence bit of the endpoint EPa.

D3 ToggleSet

Set the toggle sequence bit of the endpoint EPa to 1.

D2 ToggleClr

Set the toggle sequence bit of the endpoint EPa to 0 (to be cleared).

D1 ForceNAK

If this bit is set to 1, the NAK response is done for the transaction of the endpoint EPa regardless of the FIFO data quantity and space capacity.

When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

D0 ForceSTALL

If this bit is set to 1, the STALL response is done for the transaction of the endpoint EPa. This bit has a priority over the setting of the ForceNAK bit.

When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

0x4541: EPbControl (EPb Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbControl (EPb control)	0x4541 (8 bits)	D7	**AutoForceNAK**	1	Auto force NAK	0	Do nothing	0	R/W	
		D6	**EnShortPkt**	1	Enable short packet	0	Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1	Disable auto force	0	Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit				0	R	
		D3	**ToggleSet**	1	Set toggle sequence bit	0	Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1	Clear toggle sequence bit	0	Do nothing	0	W	
		D1	**ForceNAK**	1	Force NAK	0	Do nothing	0	R/W	
		D0	**ForceSTALL**	1	Force STALL	0	Do nothing	0	R/W	

This register sets operations of the endpoint EPb.

D7 AutoForceNAK

Sets the ForceNAK bit of this register to 1 when the transaction of the endpoint EPb completes normally.

D6 EnShortPkt

Setting this bit to 1 enables to send the data within the FIFO that is less than the quantity specified for the MaxPacketSize, as a short packet for the IN transaction of the endpoint EPb. When the IN transaction that transmitted short packets completes, this bit is automatically set to 0 (to be cleared). When a packet of the max packet size is transmitted, this bit is not cleared.
If this bit is set to 1 when the FIFO has no data, a zero-length packet can be transmitted for the IN token from the host. If the data is written into the FIFO that is in the transmission process with the packet to which this bit is set, that data may be included in transmission. Therefore, do not write into the FIFO until the packet transmission completes and this bit is cleared.

D5 DisAF_NAK_Short

When this bit is set to 0 (default setting) and the packet that was received at normal completion time of the OUT transaction is a short packet, the ForceNAK bit is automatically set to 1. When this bit is set to 1, this function is disabled.
When the AutoForceNAK bit is set to 1, the AutoForceNAK bit has a priority.

D4 ToggleStat

Shows the status of the toggle sequence bit of the endpoint EPb.

D3 ToggleSet

Set the toggle sequence bit of the endpoint EPb to 1.

D2 ToggleClr

Set the toggle sequence bit of the endpoint EPb to 0 (to be cleared).

D1 ForceNAK

If this bit is set to 1, the NAK response is done for the transaction of the endpoint EPb regardless of the FIFO data quantity and space capacity.
When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

D0 ForceSTALL

If this bit is set to 1, the STALL response is done for the transaction of the endpoint EPb. This bit has a priority over the setting of the ForceNAK bit.
When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

0x4542: EPcControl (EPc Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcControl (EPc control)	0x4542 (8 bits)	D7	AutoForceNAK	1	Auto force NAK	0	Do nothing	0	R/W	
		D6	EnShortPkt	1	Enable short packet	0	Do nothing	0	R/W	
		D5	DisAF_NAK_Short	1	Disable auto force	0	Auto force NAK short	0	R/W	
		D4	ToggleStat	Toggle sequence bit				0	R	
		D3	ToggleSet	1	Set toggle sequence bit	0	Do nothing	0	W	0 when being read.
		D2	ToggleClr	1	Clear toggle sequence bit	0	Do nothing	0	W	
		D1	ForceNAK	1	Force NAK	0	Do nothing	0	R/W	
		D0	ForceSTALL	1	Force STALL	0	Do nothing	0	R/W	

This register sets operations of the endpoint EPc.

D7 AutoForceNAK

Sets the ForceNAK bit of this register to 1 when the transaction of the endpoint EPc completes normally.

D6 EnShortPkt

Setting this bit to 1 enables to send the data within the FIFO that is less than the quantity specified for the MaxPacketSize, as a short packet for the IN transaction of the endpoint EPc. When the IN transaction that transmitted short packets completes, this bit is automatically set to 0 (to be cleared). When a packet of the max packet size is transmitted, this bit is not cleared.

If this bit is set to 1 when the FIFO has no data, a zero-length packet can be transmitted for the IN token from the host. If the data is written into the FIFO that is in the transmission process with the packet to which this bit is set, that data may be included in transmission. Therefore, do not write into the FIFO until the packet transmission completes and this bit is cleared.

D5 DisAF_NAK_Short

When this bit is set to 0 (default setting) and the packet that was received at normal completion time of the OUT transaction is a short packet, the ForceNAK bit is automatically set to 1. When this bit is set to 1, this function is disabled.

When the AutoForceNAK bit is set to 1, the AutoForceNAK bit has a priority.

D4 ToggleStat

Shows the status of the toggle sequence bit of the endpoint EPc.

D3 ToggleSet

Set the toggle sequence bit of the endpoint EPc to 1.

D2 ToggleClr

Set the toggle sequence bit of the endpoint EPc to 0 (to be cleared).

D1 ForceNAK

If this bit is set to 1, the NAK response is done for the transaction of the endpoint EPc regardless of the FIFO data quantity and space capacity.

When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

D0 ForceSTALL

If this bit is set to 1, the STALL response is done for the transaction of the endpoint EPc. This bit has a priority over the setting of the ForceNAK bit.

When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

0x4543: EPdControl (EPd Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdControl (EPd control)	0x4543 (8 bits)	D7	**AutoForceNAK**	1	Auto force NAK	0	Do nothing	0	R/W	
		D6	**EnShortPkt**	1	Enable short packet	0	Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1	Disable auto force	0	Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit				0	R	
		D3	**ToggleSet**	1	Set toggle sequence bit	0	Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1	Clear toggle sequence bit	0	Do nothing	0	W	
		D1	**ForceNAK**	1	Force NAK	0	Do nothing	0	R/W	
		D0	**ForceSTALL**	1	Force STALL	0	Do nothing	0	R/W	

This register sets operations of the endpoint EPd.

D7 AutoForceNAK

Sets the ForceNAK bit of this register to 1 when the transaction of the endpoint EPd completes normally.

D6 EnShortPkt

Setting this bit to 1 enables to send the data within the FIFO that is less than the quantity specified for the MaxPacketSize, as a short packet for the IN transaction of the endpoint EPd. When the IN transaction that transmitted short packets completes, this bit is automatically set to 0 (to be cleared). When a packet of the max packet size is transmitted, this bit is not cleared.
If this bit is set to 1 when the FIFO has no data, a zero-length packet can be transmitted for the IN token from the host. If the data is written into the FIFO that is in the transmission process with the packet to which this bit is set, that data may be included in transmission. Therefore, do not write into the FIFO until the packet transmission completes and this bit is cleared.

D5 DisAF_NAK_Short

When this bit is set to 0 (default setting) and the packet that was received at normal completion time of the OUT transaction is a short packet, the ForceNAK bit is automatically set to 1. When this bit is set to 1, this function is disabled.
When the AutoForceNAK bit is set to 1, the AutoForceNAK bit has a priority.

D4 ToggleStat

Shows the status of the toggle sequence bit of the endpoint EPd.

D3 ToggleSet

Set the toggle sequence bit of the endpoint EPd to 1.

D2 ToggleClr

Set the toggle sequence bit of the endpoint EPd to 0 (to be cleared).

D1 ForceNAK

If this bit is set to 1, the NAK response is done for the transaction of the endpoint EPd regardless of the FIFO data quantity and space capacity.
When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

D0 ForceSTALL

If this bit is set to 1, the STALL response is done for the transaction of the endpoint EPd. This bit has a priority over the setting of the ForceNAK bit.
When a transaction has been being done for a certain period of time, the setting of this bit will be enabled from the next transaction.

0x4550: EPaMaxSize_H (EPa Max Packet Size HIGH)
0x4551: EPaMaxSize_L (EPa Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPaMaxSize_H (EPa max packet size high)	**0x4550** (8 bits)	D7–2	–	–	–	–	0 when being read.
		D1	**EPaMaxSize[9]**	Endpoint EPa max packet size	0	R/W	
		D0	**EPaMaxSize[8]**		0		
EPaMaxSize_L (EPa max packet size low)	**0x4551** (8 bits)	D7	**EPaMaxSize[7]**	Endpoint EPa max packet size	0	R/W	
		D6	**EPaMaxSize[6]**		0		
		D5	**EPaMaxSize[5]**		0		
		D4	**EPaMaxSize[4]**		0		
		D3	**EPaMaxSize[3]**		0		
		D2	**EPaMaxSize[2]**		0		
		D1	**EPaMaxSize[1]**		0		
		D0	**EPaMaxSize[0]**		0		

EPaMaxSize[9:0]

This register sets the MaxPacketSize of the endpoint EPa.
When using this endpoint for the bulk transfer, 8, 16, 32, or 64 bytes should be set.
When using this endpoint for the interrupt transfer, up to 64 bytes can be set.
If the area of the endpoint EPa is smaller than specified here, the macro does not operate normally.

0x4552: EPaConfig_0 (EPa Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaConfig_0 (EPa configuration 0)	0x4552 (8 bits)	D7	INxOUT	1	In	0	Out	0	R/W	
		D6	ToggleMode	1	Always toggle	0	Normal toggle	0	R/W	
		D5	EnEndPoint	1	Enable endpoint	0	Disable endpoint	0	R/W	
		D4	–	–				–	–	0 when being read.
		D3	EndPointNumber[3]	Endpoint number (0x1 to 0xf)				0	R/W	
		D2	EndPointNumber[2]					0		
		D1	EndPointNumber[1]					0		
		D0	EndPointNumber[0]					0		

This register sets up the endpoint EPa.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 INxOUT

Set the transfer direction of the endpoint.

D6 ToggleMode

Set the operation mode of the toggle sequence bit. (Only for the IN transaction)
Normal toggle - Perform the toggle only when the transaction ends normally.
Always toggle - Always perform the toggle for every transaction.

D5 EnEndPoint

Setting this bit to 1 enables this endpoint.
When this bit is 0, access to an endpoint is neglected.
Perform the setup according to the SetConfiguration request from the host.

D4 Reserved

D[3:0] EndPointNumber

Set an endpoint number between 0x1 and 0xf.

0x4553: EPaConfig_1 (EPa Configuration 1)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaConfig_1 (EPa configuration 1)	**0x4553** (8 bits)	D7	**ISO**	1	ISO	0	Non-ISO	0	R/W	
		D6–0	–	–				–	–	0 when being read.

This register sets up the endpoint EPa.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 ISO

Set the isochronous mode.

D[6:0] Reserved

0x4554: EPbMaxSize_H (EPb Max Packet Size HIGH)
0x4555: EPbMaxSize_L (EPb Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPbMaxSize_H	**0x4554**	D7–2	–	–	–	–	0 when being read.
(EPb max	(8 bits)	D1	**EPbMaxSize[9]**	Endpoint EPb max packet size	0	R/W	
packet size		D0	**EPbMaxSize[8]**		0		
high)							
EPbMaxSize_L	**0x4555**	D7	**EPbMaxSize[7]**	Endpoint EPb max packet size	0	R/W	
(EPb max	(8 bits)	D6	**EPbMaxSize[6]**		0		
packet size		D5	**EPbMaxSize[5]**		0		
low)		D4	**EPbMaxSize[4]**		0		
		D3	**EPbMaxSize[3]**		0		
		D2	**EPbMaxSize[2]**		0		
		D1	**EPbMaxSize[1]**		0		
		D0	**EPbMaxSize[0]**		0		

EPbMaxSize[9:0]

This register sets the MaxPacketSize of the endpoint EPb.
When using this endpoint for the bulk transfer, 8, 16, 32, or 64 bytes should be set.
When using this endpoint for the interrupt transfer, up to 64 bytes can be set.
If the area of the endpoint EPb is smaller than specified here, the macro does not operate normally.

0x4556: EPbConfig_0 (EPb Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbConfig_0 (EPb configuration 0)	**0x4556** (8 bits)	D7	**INxOUT**	1	In	0	Out	0	R/W	
		D6	**ToggleMode**	1	Always toggle	0	Normal toggle	0	R/W	
		D5	**EnEndPoint**	1	Enable endpoint	0	Disable endpoint	0	R/W	
		D4	**–**	–				–	–	0 when being read.
		D3	**EndPointNumber[3]**	Endpoint number (0x1 to 0xf)				0	R/W	
		D2	**EndPointNumber[2]**					0		
		D1	**EndPointNumber[1]**					0		
		D0	**EndPointNumber[0]**					0		

This register sets up the endpoint EPb.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 INxOUT

Set the transfer direction of the endpoint.

D6 ToggleMode

Set the operation mode of the toggle sequence bit. (Only for the IN transaction)
Normal toggle - Perform the toggle only when the transaction ends normally.
Always toggle - Always perform the toggle for every transaction.

D5 EnEndPoint

Setting this bit to 1 enables this endpoint.
When this bit is 0, access to an endpoint is neglected.
Perform the setup according to the SetConfiguration request from the host.

D4 Reserved

D[3:0] EndPointNumber

Set an endpoint number between 0x1 and 0xf.

0x4557: EPbConfig_1 (EPb Configuration 1)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPbConfig_1 (EPb configuration 1)	**0x4557** (8 bits)	D7	**ISO**	1 ISO / 0 Non-ISO	0	R/W	
		D6–0	**–**	–	–	–	0 when being read.

This register sets up the endpoint EPb.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 ISO

Set the isochronous mode.

D[6:0] Reserved

0x4558: EPcMaxSize_H (EPc Max Packet Size HIGH)
0x4559: EPcMaxSize_L (EPc Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcMaxSize_H	**0x4558**	D7–2	–	–	–	–	0 when being read.
(EPc max	(8 bits)	D1	**EPcMaxSize[9]**	Endpoint EPc max packet size	0	R/W	
packet size		D0	**EPcMaxSize[8]**		0		
high)							
EPcMaxSize_L	**0x4559**	D7	**EPcMaxSize[7]**	Endpoint EPc max packet size	0	R/W	
(EPc max	(8 bits)	D6	**EPcMaxSize[6]**		0		
packet size		D5	**EPcMaxSize[5]**		0		
low)		D4	**EPcMaxSize[4]**		0		
		D3	**EPcMaxSize[3]**		0		
		D2	**EPcMaxSize[2]**		0		
		D1	**EPcMaxSize[1]**		0		
		D0	**EPcMaxSize[0]**		0		

EPcMaxSize[9:0]

This register sets the MaxPacketSize of the endpoint EPc.

When using this endpoint for the bulk transfer, 8, 16, 32, or 64 bytes should be set.

When using this endpoint for the interrupt transfer, up to 64 bytes can be set.

If the area of the endpoint EPc is smaller than specified here, the macro does not operate normally.

0x455a: EPcConfig_0 (EPc Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcConfig_0 (EPc configuration 0)	0x455a (8 bits)	D7	INxOUT	1	In	0	Out	0	R/W	
		D6	ToggleMode	1	Always toggle	0	Normal toggle	0	R/W	
		D5	EnEndPoint	1	Enable endpoint	0	Disable endpoint	0	R/W	
		D4	–	–				–	–	0 when being read.
		D3	EndPointNumber[3]	Endpoint number (0x1 to 0xf)				0	R/W	
		D2	EndPointNumber[2]					0		
		D1	EndPointNumber[1]					0		
		D0	EndPointNumber[0]					0		

This register sets up the endpoint EPc.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 INxOUT

Set the transfer direction of the endpoint.

D6 ToggleMode

Set the operation mode of the toggle sequence bit. (Only for the IN transaction)
Normal toggle - Perform the toggle only when the transaction ends normally.
Always toggle - Always perform the toggle for every transaction.

D5 EnEndPoint

Setting this bit to 1 enables this endpoint.
When this bit is 0, access to an endpoint is neglected.
Perform the setup according to the SetConfiguration request from the host.

D4 Reserved

D[3:0] EndPointNumber

Set an endpoint number between 0x1 and 0xf.

0x455b: EPcConfig_1 (EPc Configuration 1)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcConfig_1 (EPc configuration 1)	**0x455b** (8 bits)	D7	**ISO**	1	ISO	0	Non-ISO	0	R/W	
		D6–0	–	–				–	–	0 when being read.

This register sets up the endpoint EPc.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 ISO

Set the isochronous mode.

D[6:0] Reserved

0x455c: EPdMaxSize_H (EPd Max Packet Size HIGH)
0x455d: EPdMaxSize_L (EPd Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdMaxSize_H	**0x455c**	D7–2	–	–	–	–	0 when being read.
(EPd max	(8 bits)	D1	**EPdMaxSize[9]**	Endpoint EPd max packet size	0	R/W	
packet size		D0	**EPdMaxSize[8]**		0		
high)							
EPdMaxSize_L	**0x455d**	D7	**EPdMaxSize[7]**	Endpoint EPd max packet size	0	R/W	
(EPd max	(8 bits)	D6	**EPdMaxSize[6]**		0		
packet size		D5	**EPdMaxSize[5]**		0		
low)		D4	**EPdMaxSize[4]**		0		
		D3	**EPdMaxSize[3]**		0		
		D2	**EPdMaxSize[2]**		0		
		D1	**EPdMaxSize[1]**		0		
		D0	**EPdMaxSize[0]**		0		

EPdMaxSize[9:0]

This register sets the MaxPacketSize of the endpoint EPd.

When using this endpoint for the bulk transfer, 8, 16, 32, or 64 bytes should be set.

When using this endpoint for the interrupt transfer, up to 64 bytes can be set.

If the area of the endpoint EPd is smaller than specified here, the macro does not operate normally.

0x455e: EPdConfig_0 (EPd Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdConfig_0 (EPd configuration 0)	**0x455e** (8 bits)	D7	**INxOUT**	1	In	0	Out	0	R/W	
		D6	**ToggleMode**	1	Always toggle	0	Normal toggle	0	R/W	
		D5	**EnEndPoint**	1	Enable endpoint	0	Disable endpoint	0	R/W	
		D4	**–**	–				–	–	0 when being read.
		D3	**EndPointNumber[3]**	Endpoint number (0x1 to 0xf)				0	R/W	
		D2	**EndPointNumber[2]**					0		
		D1	**EndPointNumber[1]**					0		
		D0	**EndPointNumber[0]**					0		

This register sets up the endpoint EPd.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 INxOUT

Set the transfer direction of the endpoint.

D6 ToggleMode

Set the operation mode of the toggle sequence bit. (Only for the IN transaction)
Normal toggle - Perform the toggle only when the transaction ends normally.
Always toggle - Always perform the toggle for every transaction.

D5 EnEndPoint

Setting this bit to 1 enables this endpoint.
When this bit is 0, access to an endpoint is neglected.
Perform the setup according to the SetConfiguration request from the host.

D4 Reserved

D[3:0] EndPointNumber

Set an endpoint number between 0x1 and 0xf.

0x455f: EPdConfig_1 (EPd Configuration 1)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdConfig_1 (EPd configuration 1)	**0x455f** (8 bits)	D7	**ISO**	1 ISO / 0 Non-ISO	0	R/W	
		D6–0	**–**	–	–	–	0 when being read.

This register sets up the endpoint EPd.
Perform the setup so that combination of the EndpointNumber and the INxOUT does not overlap with those of other endpoints.

D7 ISO

Set the isochronous mode.

D[6:0] Reserved

0x4570: EPaStartAdrs_H (EPa FIFO Start Address HIGH)
0x4571: EPaStartAdrs_L (EPa FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPaStartAdrs_H	**0x4570**	D7–4	–	–	–	–	0 when being read.
(EPa FIFO start	(8 bits)	D3	**EPaStartAdrs[11]**	Endpoint EPa start address	0	R/W	
address high)		D2	**EPaStartAdrs[10]**		0		
		D1	**EPaStartAdrs[9]**		0		
		D0	**EPaStartAdrs[8]**		0		
EPaStartAdrs_L	**0x4571**	D7	**EPaStartAdrs[7]**	Endpoint EPa start address	0	R/W	
(EPa FIFO start	(8 bits)	D6	**EPaStartAdrs[6]**		0		
address low)		D5	**EPaStartAdrs[5]**		0		
		D4	**EPaStartAdrs[4]**		0		
		D3	**EPaStartAdrs[3]**		0		
		D2	**EPaStartAdrs[2]**		0		
		D1–0	–	–	–	–	0 when being read.

EPaStartAdrs[11:2]

Sets the start address of the FIFO area allocated to the endpoint EPa.

The area that is allocated to the endpoint EPa is from the address set by the EPaStartAdrs and to the address one byte before the one set by the EPbStartAdrs.

After setting the StartAdrs of all endpoints, be sure to set the AllFIFO_Clr bit of the EPnControl register to 1 to clear all FIFOs.

If the EPaMaxSize of the endpoint EPa is larger than the area specified in here, the macro does not operate normally.

Set the total of the FIFO area secured for all endpoints does not exceed the total capacity of the built-in RAM.

Allocate the FIFO area to the endpoints in the order from the lower order address to higher order address like EP0, EPa, EPb, EPc, EPd.

The FIFO of the endpoint EP0 is allocated from the address 0 to up to the size specified as the MaxPacketSize of the endpoint EP0 set in the EP0MaxSize register. Allocate the succeeding area for other endpoints.

Since the FIFO capacity is 1K bytes, do not let the EPd end address exceed 0x3ff. And do not let the EPaStartAdrs exceed the setting value of the EPbStartAdrs.

0x4572: EPbStartAdrs_H (EPb FIFO Start Address HIGH)
0x4573: EPbStartAdrs_L (EPb FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPbStartAdrs_H	**0x4572**	D7–4	–	–	–	–	0 when being read.
(EPb FIFO start	(8 bits)	D3	**EPbStartAdrs[11]**	Endpoint EPb start address	0	R/W	
address high)		D2	**EPbStartAdrs[10]**		0		
		D1	**EPbStartAdrs[9]**		0		
		D0	**EPbStartAdrs[8]**		0		
EPbStartAdrs_L	**0x4573**	D7	**EPbStartAdrs[7]**	Endpoint EPb start address	0	R/W	
(EPb FIFO start	(8 bits)	D6	**EPbStartAdrs[6]**		0		
address low)		D5	**EPbStartAdrs[5]**		0		
		D4	**EPbStartAdrs[4]**		0		
		D3	**EPbStartAdrs[3]**		0		
		D2	**EPbStartAdrs[2]**		0		
		D1–0	–	–	–	–	0 when being read.

EPbStartAdrs[11:2]

Sets the start address of the FIFO area allocated to the endpoint EPb.

The area that is allocated to the endpoint EPb is from the address set by the EPbStartAdrs and to the address one byte before the one set by the EPcStartAdrs.

After setting the StartAdrs of all endpoints, be sure to set the AllFIFO_Clr bit of the EPnControl register to 1 to clear all FIFOs.

If the EPbMaxSize of the endpoint EPb is larger than the area specified in here, the macro does not operate normally.

Set the total of the FIFO area secured for all endpoints does not exceed the total capacity of the built-in RAM.

Allocate the FIFO area to the endpoints in the order from the lower order address to higher order address like EP0, EPa, EPb, EPc, EPd.

The FIFO of the endpoint EP0 is allocated from the address 0 to up to the size specified as the MaxPacketSize of the endpoint EP0 set in the EP0MaxSize register. Allocate the succeeding area for other endpoints.

Since the FIFO capacity is 1K bytes, do not let the EPd end address exceed 0x3ff. And do not let the EPbStartAdrs exceed the setting value of the EPcStartAdrs.

0x4574: EPcStartAdrs_H (EPc FIFO Start Address HIGH)
0x4575: EPcStartAdrs_L (EPc FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcStartAdrs_H	**0x4574**	D7–4	–	–	–	–	0 when being read.
(EPc FIFO start	(8 bits)	D3	**EPcStartAdrs[11]**	Endpoint EPc start address	0	R/W	
address high)		D2	**EPcStartAdrs[10]**		0		
		D1	**EPcStartAdrs[9]**		0		
		D0	**EPcStartAdrs[8]**		0		
EPcStartAdrs_L	**0x4575**	D7	**EPcStartAdrs[7]**	Endpoint EPc start address	0	R/W	
(EPc FIFO start	(8 bits)	D6	**EPcStartAdrs[6]**		0		
address low)		D5	**EPcStartAdrs[5]**		0		
		D4	**EPcStartAdrs[4]**		0		
		D3	**EPcStartAdrs[3]**		0		
		D2	**EPcStartAdrs[2]**		0		
		D1–0	–	–	–	–	0 when being read.

EPcStartAdrs[11:2]

Sets the start address of the FIFO area allocated to the endpoint EPc.

The area that is allocated to the endpoint EPc is from the address set by the EPcStartAdrs and to the address one byte before the one set by the EPdStartAdrs.

After setting the StartAdrs of all endpoints, be sure to set the AllFIFO_Clr bit of the EPnControl register to 1 to clear all FIFOs.

If the EPcMaxSize of the endpoint EPc is larger than the area specified in here, the macro does not operate normally.

Set the total of the FIFO area secured for all endpoints does not exceed the total capacity of the built-in RAM.

Allocate the FIFO area to the endpoints in the order from the lower order address to higher order address like EP0, EPa, EPb, EPc, EPd.

The FIFO of the endpoint EP0 is allocated from the address 0 to up to the size specified as the MaxPacketSize of the endpoint EP0 set in the EP0MaxSize register. Allocate the succeeding area for other endpoints.

Since the FIFO capacity is 1K bytes, do not let the EPd end address exceed 0x3ff. And do not let the EPcStartAdrs exceed the setting value of the EPdStartAdrs.

0x4576: EPdStartAdrs_H (EPd FIFO Start Address HIGH)
0x4577: EPdStartAdrs_L (EPd FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdStartAdrs_H	**0x4576**	D7–4	–	–	–	–	0 when being read.
(EPd FIFO start	(8 bits)	D3	**EPdStartAdrs[11]**	Endpoint EPd start address	0	R/W	
address high)		D2	**EPdStartAdrs[10]**		0		
		D1	**EPdStartAdrs[9]**		0		
		D0	**EPdStartAdrs[8]**		0		
EPdStartAdrs_L	**0x4577**	D7	**EPdStartAdrs[7]**	Endpoint EPd start address	0	R/W	
(EPd FIFO start	(8 bits)	D6	**EPdStartAdrs[6]**		0		
address low)		D5	**EPdStartAdrs[5]**		0		
		D4	**EPdStartAdrs[4]**		0		
		D3	**EPdStartAdrs[3]**		0		
		D2	**EPdStartAdrs[2]**		0		
		D1–0	–	–	–	–	0 when being read.

EPdStartAdrs[11:2]

Sets the start address of the FIFO area allocated to the endpoint EPd.

The area that is allocated to the endpoint EPd is from the address set by the EPdStartAdrs and to the end address of the FIFO.

After setting the StartAdrs of all endpoints, be sure to set the AllFIFO_Clr bit of the EPnControl register to 1 to clear all FIFOs.

If the EPdMaxSize of the endpoint EPd is larger than the area specified in here, the macro does not operate normally.

Set the total of the FIFO area secured for all endpoints does not exceed the total capacity of the built-in RAM.

Allocate the FIFO area to the endpoints in the order from the lower order address to higher order address like EP0, EPa, EPb, EPc, EPd.

The FIFO of the endpoint EP0 is allocated from the address 0 to up to the size specified as the MaxPacketSize of the endpoint EP0 set in the EP0MaxSize register. Allocate the succeeding area for other endpoints.

Since the FIFO capacity is 1K bytes, do not let the EPd end address exceed 0x3ff.

0x4580: CPU_JoinRd (CPU Join FIFO Read)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
CPU_JoinRd (CPU join FIFO read)	0x4580 (8 bits)	D7–4	–	–				–	–	0 when being read.
		D3	JoinEPdRd	1	Join EPd FIFO read	0	Do nothing	0	R/W	
		D2	JoinEPcRd	1	Join EPc FIFO read	0	Do nothing	0	R/W	
		D1	JoinEPbRd	1	Join EPb FIFO read	0	Do nothing	0	R/W	
		D0	JoinEPaRd	1	Join EPa FIFO read	0	Do nothing	0	R/W	

This register can be set up to read the FIFO data of the endpoint through the CPU Interface. When the EPnFIFOforCPU register is read after the setup of this register is completed, the FIFO data of the relevant endpoint can be read. The remained data quantity of the FIFO can be referred by the EPnRdRemain_H, L register.
This register can set only one bit to 1 at the same time. When 1 is written into multiple bits at the same time, writing in higher order bit is regarded as valid. When all bits are set to 0, EP0 will be joined.
The reading data from CPU I/F through the endpoint used by USB I/F is not allowed.
If CPU I/F needs to read from the IN direction endpoint, use the ForceNAK bit to avoid reading data from USB I/F.
This register is valid when EnEPnFIFO_Access.EnEPnFIFO_Rd bit is set.

D[7:4] Reserved

D3 JoinEPdRd

If this bit is set to 1, the FIFO data of the endpoint EPd can be read from the EPnFIFOforCPU register. In addition, reference to the data quantity in the FIFO of the endpoint EPd by the EPnRdRemain_H, L register is enabled.

D2 JoinEPcRd

If this bit is set to 1, the FIFO data of the endpoint EPc can be read from the EPnFIFOforCPU register. In addition, reference to the data quantity in the FIFO of the endpoint EPc by the EPnRdRemain_H, L register is enabled.

D1 JoinEPbRd

If this bit is set to 1, the FIFO data of the endpoint EPb can be read from the EPnFIFOforCPU register. In addition, reference to the data quantity in the FIFO of the endpoint EPb by the EPnRdRemain_H, L register is enabled.

D0 JoinEPaRd

If this bit is set to 1, the FIFO data of the endpoint EPa can be read from the EPnFIFOforCPU register. In addition, reference to the data quantity in the FIFO of the endpoint EPa by the EPnRdRemain_H, L register is enabled.

0x4581: CPU_JoinWr (CPU Join FIFO Write)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
CPU_JoinWr (CPU join FIFO write)	0x4581 (8 bits)	D7–4	–	–				–	–	0 when being read.
		D3	JoinEPdWr	1	Join EPd FIFO write	0	Do nothing	0	R/W	
		D2	JoinEPcWr	1	Join EPc FIFO write	0	Do nothing	0	R/W	
		D1	JoinEPbWr	1	Join EPb FIFO write	0	Do nothing	0	R/W	
		D0	JoinEPaWr	1	Join EPa FIFO write	0	Do nothing	0	R/W	

This register can be set up to write the FIFO data of the endpoint through the CPU Interface. When the EPnFIFOforCPU register is written after the setup of this register is completed, the FIFO data of the relevant endpoint can be written. The space capacity of the FIFO can be referred by the EPnWrRemain_H, L register.
This register can set only one bit to 1 at the same time. When 1 is written into multiple bits at the same time, writing in higher order bit is regarded as valid. When all bits are set to 0, EP0 will be joined.
The writing data from CPU I/F through the endpoint used by USB I/F is not allowed.
If CPU I/F needs to write to the OUT direction endpoint, use the ForceNAK bit to avoid writing data from USB I/F.
This register is valid when EnEPnFIFO_Access.EnEPnFIFO_Wr bit is set.

D[7:4] Reserved

D3 JoinEPdWr

If this bit is set to 1, the FIFO data of the endpoint EPd can be written into the EPnFIFOforCPU register. In addition, reference to the space capacity in the FIFO of the endpoint EPd by the EPnWrRemain_H, L register is enabled.

D2 JoinEPcWr

If this bit is set to 1, the FIFO data of the endpoint EPc can be written into the EPnFIFOforCPU register. In addition, reference to the space capacity in the FIFO of the endpoint EPc by the EPnWrRemain_H, L register is enabled.

D1 JoinEPbWr

If this bit is set to 1, the FIFO data of the endpoint EPb can be written into the EPnFIFOforCPU register. In addition, reference to the space capacity in the FIFO of the endpoint EPb by the EPnWrRemain_H, L register is enabled.

D0 JoinEPaWr

If this bit is set to 1, the FIFO data of the endpoint EPa can be written into the EPnFIFOforCPU register. In addition, reference to the space capacity in the FIFO of the endpoint EPa by the EPnWrRemain_H, L register is enabled.

0x4582: EnEPnFIFO_Access (Enable EPn FIFO Access)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EnEPnFIFO	0x4582	D7–2	–	–				–	–	0 when being read.
_Access	(8 bits)	D1	EnEPnFIFO_Wr	1	Enable join EPn FIFO write	0	Do nothing	0	R/W	
(Enable EPn FIFO access)		D0	EnEPnFIFO_Rd	1	Enable join EPn FIFO read	0	Do nothing	0	R/W	

This register enables the CPU_JoinRd and CPU_JoinWr registers so that the CPU can access the EPn FIFO.

D[7:2] Reserved

D1 EnEPnFIFO_Wr

If this bit is set to 1, the CPU_JoinWr register is enabled and the CPU can write data to the EPn FIFO selected by the CPU_JoinWr register.

D0 EnEPnFIFO_Rd

If this bit is set to 1, the CPU_JoinRd register is enabled and the CPU can read data from the EPn FIFO selected by the CPU_JoinRd register.

0x4583: EPnFIFOforCPU (EPn FIFO for CPU)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPnFIFOforCPU	**0x4583**	D7	**EPnFIFOData[7]**	Endpoint EP0 FIFO access from CPU	X	R/W	
(EPn FIFO for	(8 bits)	D6	**EPnFIFOData[6]**		X		
CPU)		D5	**EPnFIFOData[5]**		X		
		D4	**EPnFIFOData[4]**		X		
		D3	**EPnFIFOData[3]**		X		
		D2	**EPnFIFOData[2]**		X		
		D1	**EPnFIFOData[1]**		X		
		D0	**EPnFIFOData[0]**		X		

D[7:0] EPnFIFOData[7:0]

This register is used for accessing the FIFO of the endpoint from the CPU Interface.
When a bit of the CPU_JoinRd register is set to 1, the data can be read from the FIFO by reading values from this register.
When a bit of the CPU_JoinWr register is set to 1, the data can be written into the FIFO by writing values into this register.
If values are read from this register without setting the EnEPnFIFO_Rd bit of the EnEPnFIFO_Access register, a dummy data will be output.
If writing is done into this register without setting the EnEPnFIFO_Wr bit of the EnEPnFIFO_Access register, writing into the FIFO is not done.

If this register is read when the FIFO of the relevant endpoint is empty, a dummy data will be read.
If writing is done into this register when the FIFO of the relevant endpoint has no space, writing into the FIFO is not done.

0x4584: EPnRdRemain_H (EPn FIFO Read Remain HIGH)
0x4585: EPnRdRemain_L (EPn FIFO Read Remain LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPnRdRemain_H	**0x4584**	D7–4	–	–	–	–	0 when being read.
(EPn FIFO read	(8 bits)	D3	**EPnRdRemain[11]**	Endpoint n FIFO read remain high	0	R	
remain high)		D2	**EPnRdRemain[10]**		0		
		D1	**EPnRdRemain[9]**		0		
		D0	**EPnRdRemain[8]**		0		
EPnRdRemain_L	**0x4585**	D7	**EPnRdRemain[7]**	Endpoint n FIFO read remain low	0	R	
(EPn FIFO read	(8 bits)	D6	**EPnRdRemain[6]**		0		
remain low)		D5	**EPnRdRemain[5]**		0		
		D4	**EPnRdRemain[4]**		0		
		D3	**EPnRdRemain[3]**		0		
		D2	**EPnRdRemain[2]**		0		
		D1	**EPnRdRemain[1]**		0		
		D0	**EPnRdRemain[0]**		0		

EPnRdRemain[11:0]

This register shows the remained data quantity in the FIFO of the endpoint connected to the CPU Interface by the CPU_JoinRd register. When the remained data quantity in the FIFO is acquired, the EPnRdRemain_H and the EPnRdRemain_L registers must be accessed as a pair. When accessing them, access the EPnRdRemain_H register first.

0x4586: EPnWrRemain_H (EPn FIFO Write Remain HIGH)
0x4587: EPnWrRemain_L (EPn FIFO Write Remain LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPnWrRemain_H	**0x4586**	D7–4	**–**	–	–	–	0 when being read.
(EPn FIFO	(8 bits)	D3	**EPnWrRemain[11]**	Endpoint n FIFO write remain high	0	R	
write remain		D2	**EPnWrRemain[10]**		0		
high)		D1	**EPnWrRemain[9]**		0		
		D0	**EPnWrRemain[8]**		0		
EPnWrRemain_L	**0x4587**	D7	**EPnWrRemain[7]**	Endpoint n FIFO write remain low	0	R	
(EPn FIFO	(8 bits)	D6	**EPnWrRemain[6]**		0		
write remain		D5	**EPnWrRemain[5]**		0		
low)		D4	**EPnWrRemain[4]**		0		
		D3	**EPnWrRemain[3]**		0		
		D2	**EPnWrRemain[2]**		0		
		D1	**EPnWrRemain[1]**		0		
		D0	**EPnWrRemain[0]**		0		

EPnWrRemain[11:0]

This register shows the space capacity in the FIFO of the endpoint connected to the CPU Interface by the CPU_JoinWr register. When the space capacity in the FIFO is acquired, the EPnWrRemain_H and the EPnWrRemain_L registers must be accessed as a pair. When accessing them, access the EPnWrRemain_H register first.

0x4588: DescAdrs_H (Descriptor Address HIGH)
0x4589: DescAdrs_L (Descriptor Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescAdrs_H	**0x4588**	D7–4	–	–	–	–	0 when being read.
(Descriptor	(8 bits)	D3	**DescAdrs[11]**	Descriptor address	0	R/W	
address high)		D2	**DescAdrs[10]**		0		
		D1	**DescAdrs[9]**		0		
		D0	**DescAdrs[8]**		0		
DescAdrs_L	**0x4589**	D7	**DescAdrs[7]**	Descriptor address	0	R/W	
(Descriptor	(8 bits)	D6	**DescAdrs[6]**		0		
address low)		D5	**DescAdrs[5]**		0		
		D4	**DescAdrs[4]**		0		
		D3	**DescAdrs[3]**		0		
		D2	**DescAdrs[2]**		0		
		D1	**DescAdrs[1]**		0		
		D0	**DescAdrs[0]**		0		

DescAdrs[11:0]

Specify the start address of the FIFO used at the start of Descriptor reply operation, Descriptor write operation and Descriptor read operation in the Descriptor reply function.

The Descriptor Address does not have the function to allocate the FIFO area for the Descriptor reply function. The entire FIFO area ranging from 0x0000 to 0x03ff (1K bytes) can be specified for the Descriptor Address, regardless of the FIFO area setting.

In the Description reply, DescAdrs is updated every time the IN transaction completes at the endpoint EP0, as many times as the number of data transmitted. Refer to the item on the ReplyDescriptor of the EP0Control register, for the Descriptor reply function.

Every time data is written into or read from the Descriptor, the DescAdrs is incremented by 1.

Refer to the item on the DescDoor register, for the Descriptor write and read functions.

The FIFO area for the Descriptor reply function is not allocated explicitly. Therefore, specify the DescAdrs_H, L register and the DescSize_H, L register to avoid overlapping with FIFOs of other endpoints. Appropriate area is the area ranging from the end address of the area reserved by the endpoint EP0 (0x0040) to the start address of the endpoint EPa (EPaStartAdrs_H, L).

When referring to the Descriptor Address, read from the DescAdrs_H to the DescAdrs_L.

0x458a: DescSize_H (Descriptor Size HIGH)
0x458b: DescSize_L (Descriptor Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescSize_H	**0x458a**	D7–2	**–**	–	–	–	0 when being read.
(Descriptor	(8 bits)	D1	**DescSize[9]**	Descriptor size	0	R/W	
size high)		D0	**DescSize[8]**		0		
DescSize_L	**0x458b**	D7	**DescSize[7]**	Descriptor size	0	R/W	
(Descriptor	(8 bits)	D6	**DescSize[6]**		0		
size low)		D5	**DescSize[5]**		0		
		D4	**DescSize[4]**		0		
		D3	**DescSize[3]**		0		
		D2	**DescSize[2]**		0		
		D1	**DescSize[1]**		0		
		D0	**DescSize[0]**		0		

DescSize[9:0]

Specify the total number of the data to reply in Descriptor reply function, for the Descriptor Size. Refer to the item on the ReplyDescriptor bit of the EP0Control register, for the Descriptor reply function.
The area ranging from 0x0000 to 0x03ff can be specified for the Descriptor Size regardless of the FIFO area setting. In the Description reply, DescAdrs is updated every time the IN transaction completes at the endpoint EP0, as many times as the number of data transmitted.
The FIFO area for the Descriptor reply function is not allocated explicitly. Therefore, specify the DescAdrs_H, L register and the DescSize_H, L register to avoid overlapping with FIFOs of other endpoints. Use the area ranging from the end address of the area reserved by the endpoint EP0 (0x0040) to the start address of the endpoint EPa (EPaStartAdrs_H, L).

When referring to the Descriptor Size, read from the DescSize_H to the DescSize_L.

0x458f: DescDoor (Descriptor Door)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescDoor	**0x458f**	D7	**DescMode[7]**	Descriptor door	0	R/W	
(Descriptor	(8 bits)	D6	**DescMode[6]**		0		
door)		D5	**DescMode[5]**		0		
		D4	**DescMode[4]**		0		
		D3	**DescMode[3]**		0		
		D2	**DescMode[2]**		0		
		D1	**DescMode[1]**		0		
		D0	**DescMode[0]**		0		

D[7:0] DescMode[7:0]

This register is the access register that is used for read and write for the Descriptor.

Before starting the write operation, set the start address of the area where the FIFO Descriptor is written, into the DescAdrs_H, L register. And then performing writing one byte by one byte into this register automatically increments the DescAdrs_H, L register one byte by one byte to write data sequentially.

The data written by the DescDoor register can be used by the ReplyDescriptor function repeatedly. Thus the Descriptor reply function protects these data from deletion and overwriting. However, if the area where the Descriptor data is written into, is overlapped with the area secured by other endpoints, the data will be overwritten.

Reading this register allows the FIFO data being read from the address specified in the DescAdrs_H, L register, sequentially. At this time, the address of the DescAdrs_H, L register is also incremented every time when the data is read. Therefore, note that even if you write and read the DescDoor register, the values written just before reading cannot be read.

S1C17501 Technical Manual

APPENDIX

Appendix A List of I/O Registers

Peripheral	Address	Register name		Function
Prescaler (8-bit device)	0x4020	PSC_CTL	Prescaler Control Register	Starts/stops the prescaler.
	0x4021–0x403f	–	–	Reserved
UART (with IrDA) (8-bit device)	0x4100	UART_ST	UART Status Register	Indicates transfer, buffer and error statuses.
	0x4101	UART_TXD	UART Transmit Data Register	Transmit data
	0x4102	UART_RXD	UART Receive Data Register	Receive data
	0x4103	UART_MOD	UART Mode Register	Sets transfer data format.
	0x4104	UART_CTL	UART Control Register	Controls data transfer.
	0x4105	UART_EXP	UART Expansion Register	Sets IrDA mode.
	0x4106–0x411f	–	–	Reserved
CLG_T16U0 timer (16-bit device)	0x4200	CLG_T16U0_CLK	CLG_T16U0 Input Clock Select Register	Selects a prescaler output clock.
	0x4202	CLG_T16U0_TR	CLG_T16U0 Reload Data Register	Sets reload data.
	0x4204	CLG_T16U0_TC	CLG_T16U0 Counter Data Register	Counter data
	0x4206	CLG_T16U0_CTL	CLG_T16U0 Control Register	Sets the timer mode and starts/stops the timer.
	0x4208–0x421f	–	–	Reserved
CLG_T8S timer (16-bit device)	0x4240	CLG_T8S_CLK	CLG_T8S Input Clock Select Register	Selects a prescaler output clock.
	0x4242	CLG_T8S_TR	CLG_T8S Reload Data Register	Sets reload data.
	0x4244	CLG_T8S_TC	CLG_T8S Counter Data Register	Counter data
	0x4246	CLG_T8S_CTL	CLG_T8S Control Register	Sets the timer mode and starts/stops the timer.
	0x4248–0x425f	–	–	Reserved
CLG_T8I timer (16-bit device)	0x4260	CLG_T8I_CLK	CLG_T8I Input Clock Select Register	Selects a prescaler output clock.
	0x4262	CLG_T8I_TR	CLG_T8I Reload Data Register	Sets reload data.
	0x4264	CLG_T8I_TC	CLG_T8I Counter Data Register	Counter data
	0x4266	CLG_T8I_CTL	CLG_T8I Control Register	Sets the timer mode and starts/stops the timer.
	0x4268–0x427f	–	–	Reserved
Interrupt controller (16-bit device)	0x42e0	ITC_AIFLG	Additional Interrupt Flag Register	Indicates/resets interrupt occurrence status.
	0x42e2	ITC_AEN	Additional Interrupt Enable Register	Enables/disables each maskable interrupt.
	0x42e4	–	–	Reserved
	0x42e6	ITC_AILV0	Additional Interrupt Level Setup Register 0	Sets the MFT interrupt level.
	0x42e8	ITC_AILV1	Additional Interrupt Level Setup Register 1	Sets the ADC interrupt level.
	0x42ea	ITC_AILV2	Additional Interrupt Level Setup Register 2	Sets the RTC and PT8 CH.0 interrupt levels.
	0x42ec	ITC_AILV3	Additional Interrupt Level Setup Register 3	Sets the PT8 CH.1 and CH.2 interrupt levels.
	0x42ee	ITC_AILV4	Additional Interrupt Level Setup Register 4	Sets the PT8 CH.3
	0x42f0	ITC_AILV5	Additional Interrupt Level Setup Register 5	Sets the SPI CH.1 and USB interrupt levels.
	0x42f2	ITC_AILV6	Additional Interrupt Level Setup Register 6	Sets the I²S interrupt level.
	0x42f4	ITC_AILV7	Additional Interrupt Level Setup Register 7	Sets the REMC interrupt level.
	0x42f6–0x42ff	–	–	Reserved
	0x4300	ITC_IFLG	Interrupt Flag Register	Indicates/resets interrupt occurrence status.
	0x4302	ITC_EN	Interrupt Enable Register	Enables/disables each maskable interrupt.
	0x4304	ITC_CTL	ITC Control Register	Enables/disables the ITC.
	0x4306	ITC_ELV0	External Interrupt Level Setup Register 0	Sets the port 0 and port 1 interrupt levels and trigger modes.
	0x4308	ITC_ELV1	External Interrupt Level Setup Register 1	Sets the port 2 and port 3 interrupt levels and trigger modes.
	0x430a	ITC_ELV2	External Interrupt Level Setup Register 2	Sets the port 4 and port 5 interrupt levels and trigger modes.
	0x430c	ITC_ELV3	External Interrupt Level Setup Register 3	Sets the port 6 and port 7 interrupt levels and trigger modes.
	0x430e	ITC_ILV0	Internal Interrupt Level Setup Register 0	Sets the CLG_T16U0 timer interrupt level.
	0x4310	ITC_ILV1	Internal Interrupt Level Setup Register 1	Sets the CLG_T8S and CLG_T8I timer interrupt levels.
	0x4312	ITC_ILV2	Internal Interrupt Level Setup Register 2	Sets the UART interrupt level.
	0x4314	ITC_ILV3	Internal Interrupt Level Setup Register 3	Sets the SPI CH.0 and I²C interrupt levels.
	0x4316–0x431f	–	–	Reserved
SPI (16-bit device)	0x4320	SPI_ST0	SPI CH.0 Status Register	Indicates transfer and buffer statuses.
	0x4322	SPI_TXD0	SPI CH.0 Transmit Data Register	Transmit data
	0x4324	SPI_RXD0	SPI CH.0 Receive Data Register	Receive data
	0x4326	SPI_CTL0	SPI CH.0 Control Register	Sets the SPI CH.0 mode and enables data transfer.
	0x4328–0x433f	–	–	Reserved
I²C (16-bit device)	0x4340	I2C_EN	I²C Enable Register	Enables the I²C module.
	0x4342	I2C_CTL	I²C Control Register	Controls the I²C operation and indicates transfer status.
	0x4344	I2C_DAT	I²C Data Register	Transmit/receive data
	0x4346	I2C_ICTL	I²C Interrupt Control Register	Controls the I²C interrupt.
	0x4348–0x435f	–	–	Reserved

Peripheral	Address	Register name		Function
GPIO (8-bit device)	0x4400	P0_DAT	P0 Port Input Data Register	P0 port input data
	0x4401	–	–	Reserved
	0x4402	P1_DAT	P1 Port Input/Output Data Register	P1 port input/output data
	0x4403	P1_IOC	P1 Port I/O Control Register	Selects the P1 port I/O direction.
	0x4404	P2_DAT	P2 Port Input/Output Data Register	P2 port input/output data
	0x4405	P2_IOC	P2 Port I/O Control Register	Selects the P2 port I/O direction.
	0x4406	P3_DAT	P3 Port Input/Output Data Register	P3 port input/output data
	0x4407	P3_IOC	P3 Port I/O Control Register	Selects the P3 port I/O direction.
	0x4408	P4_DAT	P4 Port Input/Output Data Register	P4 port input/output data
	0x4409	P4_IOC	P4 Port I/O Control Register	Selects the P4 port I/O direction.
	0x440a	P5_DAT	P5 Port Input/Output Data Register	P5 port input/output data
	0x440b	P5_IOC	P5 Port I/O Control Register	Selects the P5 port I/O direction.
	0x440c	P6_DAT	P6 Port Input/Output Data Register	P6 port input/output data
	0x440d	P6_IOC	P6 Port I/O Control Register	Selects the P6 port I/O direction.
	0x440e	P7_DAT	P7 Port Input/Output Data Register	P7 port input/output data
	0x440f	P7_IOC	P7 Port I/O Control Register	Selects the P7 port I/O direction.
	0x4410	P8_DAT	P8 Port Input/Output Data Register	P8 port input/output data
	0x4411	P8_IOC	P8 Port I/O Control Register	Selects the P8 port I/O direction.
	0x4412	P9_DAT	P9 Port Input/Output Data Register	P9 port input/output data
	0x4413	P9_IOC	P9 Port I/O Control Register	Selects the P9 port I/O direction.
	0x4414	PA_DAT	PA Port Input/Output Data Register	PA port input/output data
	0x4415	PA_IOC	PA Port I/O Control Register	Selects the PA port I/O direction.
	0x4416	PB_DAT	PB Port Input/Output Data Register	PB port input/output data
	0x4417	PB_IOC	PB Port I/O Control Register	Selects the PB port I/O direction.
	0x4418	PC_DAT	PC Port Input/Output Data Register	PC port input/output data
	0x4419	PC_IOC	PC Port I/O Control Register	Selects the PC port I/O direction.
	0x441a–0x441f	–	–	Reserved
	0x4420	P0_03_CFP	P00–P03 Port Function Select Register	Selects the P00–P03 port functions.
	0x4421	P0_47_CFP	P04–P07 Port Function Select Register	Selects the P04–P07 port functions.
	0x4422	P1_03_CFP	P10–P13 Port Function Select Register	Selects the P10–P13 port functions.
	0x4423	P1_46_CFP	P14–P16 Port Function Select Register	Selects the P14–P16 port functions.
	0x4424	P2_03_CFP	P20–P23 Port Function Select Register	Selects the P20–P23 port functions.
	0x4425	P2_47_CFP	P24–P27 Port Function Select Register	Selects the P24–P27 port functions.
	0x4426	P3_03_CFP	P30–P33 Port Function Select Register	Selects the P30–P33 port functions.
	0x4427	P3_47_CFP	P34–P37 Port Function Select Register	Selects the P34–P37 port functions.
	0x4428	P4_03_CFP	P40–P43 Port Function Select Register	Selects the P40–P43 port functions.
	0x4429	P4_45_CFP	P44–P45 Port Function Select Register	Selects the P44–P45 port functions.
	0x442a	P5_03_CFP	P50–P53 Port Function Select Register	Selects the P50–P53 port functions.
	0x442b	P5_47_CFP	P54–P57 Port Function Select Register	Selects the P54–P57 port functions.
	0x442c	P6_03_CFP	P60–P63 Port Function Select Register	Selects the P60–P63 port functions.
	0x442d	P6_47_CFP	P64–P67 Port Function Select Register	Selects the P64–P67 port functions.
	0x442e	P7_03_CFP	P70–P73 Port Function Select Register	Selects the P70–P73 port functions.
	0x442f	P7_47_CFP	P74–P77 Port Function Select Register	Selects the P74–P77 port functions.
	0x4430	P8_03_CFP	P80–P83 Port Function Select Register	Selects the P80–P83 port functions.
	0x4431	P8_46_CFP	P84–P86 Port Function Select Register	Selects the P84–P86 port functions.
	0x4432	P9_03_CFP	P90–P93 Port Function Select Register	Selects the P90–P93 port functions.
	0x4433	P9_47_CFP	P94–P97 Port Function Select Register	Selects the P94–P97 port functions.
	0x4434	PA_03_CFP	PA0–PA3 Port Function Select Register	Selects the PA0–PA3 port functions.
	0x4435	PA_46_CFP	PA4–PA6 Port Function Select Register	Selects the PA4–PA6 port functions.
	0x4436	PB_03_CFP	PB0–PB3 Port Function Select Register	Selects the PB0–PB3 port functions.
	0x4437	PB_47_CFP	PB4–PB7 Port Function Select Register	Selects the PB4–PB7 port functions.
	0x4438	PC_03_CFP	PC0–PC3 Port Function Select Register	Selects the PC0–PC3 port functions.
	0x4439	PC_47_CFP	PC4–PC7 Port Function Select Register	Selects the PC4–PC7 port functions.
	0x443a–0x443f	–	–	Reserved
	0x4440	PINTSEL0	Port Input Interrupt 0 Select Register	Selects a P*x*0 port for input interrupt.
	0x4441	PINTSEL1	Port Input Interrupt 1 Select Register	Selects a P*x*1 port for input interrupt.
	0x4442	PINTSEL2	Port Input Interrupt 2 Select Register	Selects a P*x*2 port for input interrupt.
	0x4443	PINTSEL3	Port Input Interrupt 3 Select Register	Selects a P*x*3 port for input interrupt.
	0x4444	PINTSEL4	Port Input Interrupt 4 Select Register	Selects a P*x*4 port for input interrupt.
	0x4445	PINTSEL5	Port Input Interrupt 5 Select Register	Selects a P*x*5 port for input interrupt.
	0x4446	PINTSEL6	Port Input Interrupt 6 Select Register	Selects a P*x*6 port for input interrupt.
	0x4447	PINTSEL7	Port Input Interrupt 7 Select Register	Selects a P*x*7 port for input interrupt.
	0x4448–0x44ff	–	–	Reserved
USB function controller (8-bit device)	0x4500	MainIntStat	Main Interrupt Status Register	Indicates main interrupt status.
	0x4501	SIE_IntStat	SIE Interrupt Status Register	Indicates SIE interrupt status.
	0x4502	EPrIntStat	EPr Interrupt Status Register	Indicates EPr interrupt status.
	0x4503	–	–	Reserved
	0x4504	FIFO_IntStat	FIFO Interrupt Status Register	Indicates FIFO interrupt status.

Peripheral	Address	Register name		Function
USB function controller (8-bit device)	0x4505–0x4506	–	–	Reserved
	0x4507	EP0IntStat	EP0 Interrupt Status Register	Indicates EP0 interrupt status.
	0x4508	EPaIntStat	EPa Interrupt Status Register	Indicates EPa interrupt status.
	0x4509	EPbIntStat	EPb Interrupt Status Register	Indicates EPb interrupt status.
	0x450a	EPcIntStat	EPc Interrupt Status Register	Indicates EPc interrupt status.
	0x450b	EPdIntStat	EPd Interrupt Status Register	Indicates EPd interrupt status.
	0x450c–0x450f	–	–	Reserved
	0x4510	MainIntEnb	Main Interrupt Enable Register	Enables main interrupts.
	0x4511	SIE_IntEnb	SIE Interrupt Enable Register	Enables SIE interrupts.
	0x4512	EPrIntEnb	EPr Interrupt Enable Register	Enables EPr interrupts.
	0x4513	–	–	Reserved
	0x4514	FIFOIntEnb	FIFO Interrupt Enable	Enables FIFO interrupts.
	0x4515–0x4516	–	–	Reserved
	0x4517	EP0IntEnb	EP0 Interrupt Enable Register	Enables EP0 interrupts.
	0x4518	EPaIntEnb	EPa Interrupt Enable Register	Enables EPa interrupts.
	0x4519	EPbIntEnb	EPb Interrupt Enable Register	Enables EPb interrupts.
	0x451a	EPcIntEnb	EPc Interrupt Enable Register	Enables EPc interrupts.
	0x451b	EPdIntEnb	EPd Interrupt Enable Register	Enables EPd interrupts.
	0x451c–0x451f	–	–	Reserved
	0x4520	RevisionNum	Revision Number Register	Indicates the revision number of the USB controller.
	0x4521	USB_Control	USB Control Register	Controls USB conditions.
	0x4522	USB_Status	USB Status Register	Indicates the USB status.
	0x4523	XcvrControl	Xcvr Control Register	Controls the transceiver macro.
	0x4524	USB_Test	USB Test Register	Sets up the USB test mode.
	0x4525	EPnControl	Endpoint Control Register	Clears all FIFOs and sets NAK/STALL bits.
	0x4526	EPrFIFO_Clr	EPr FIFO Clear Register	Clears each FIFO.
	0x4527–0x452d	–	–	Reserved
	0x452e	FrameNumber_H	Frame Number High Register	Frame number
	0x452f	FrameNumber_L	Frame Number Low Register	
	0x4530	EP0Setup_0	EP0 Setup 0 Register	EP0 setup data (BmRequestType)
	0x4531	EP0Setup_1	EP0 Setup 1 Register	EP0 setup data (BRequest)
	0x4532	EP0Setup_2	EP0 Setup 2 Register	EP0 setup data (low-order Wvalue bits)
	0x4533	EP0Setup_3	EP0 Setup 3 Register	EP0 setup data (high-order Wvalue bits)
	0x4534	EP0Setup_4	EP0 Setup 4 Register	EP0 setup data (low-order WIndex bits)
	0x4535	EP0Setup_5	EP0 Setup 5 Register	EP0 setup data (high-order WIndex bits)
	0x4536	EP0Setup_6	EP0 Setup 6 Register	EP0 setup data (low-order WLength bits)
	0x4537	EP0Setup_7	EP0 Setup 7 Register	EP0 setup data (high-order WLength bits)
	0x4538	USB_Address	USB Address Register	Sets a USB address.
	0x4539	EP0Control	EP0 Control Register	Sets up EP0.
	0x453a	EP0ControlIN	EP0 Control In Register	Sets EP0 IN transaction conditions.
	0x453b	EP0ControlOUT	EP0 Control Out Register	Sets EP0 OUT transaction conditions.
	0x453c–0x453e	–	–	Reserved
	0x453f	EP0MaxSize	EP0 Max Packet Size Register	Sets the EP0 max packet size.
	0x4540	EPaControl	EPa Control Register	Sets up EPa.
	0x4541	EPbControl	EPb Control Register	Sets up EPb.
	0x4542	EPcControl	EPc Control Register	Sets up EPc.
	0x4543	EPdControl	EPd Control Register	Sets up EPd.
	0x4544–0x454f	–	–	Reserved
	0x4550	EPaMaxSize_H	EPa Max Packet Size High Register	Sets the EPa max packet size.
	0x4551	EPaMaxSize_L	EPa Max Packet Size Low Register	
	0x4552	EPaConfig_0	EPa Configuration 0 Register	Configures EPa.
	0x4553	EPaConfig_1	EPa Configuration 1 Register	
	0x4554	EPbMaxSize_H	EPb Max Packet Size High Register	Sets the EPb max packet size.
	0x4555	EPbMaxSize_L	EPb Max Packet Size Low Register	
	0x4556	EPbConfig_0	EPb Configuration 0 Register	Configures EPb.
	0x4557	EPbConfig_1	EPb Configuration 1 Register	
	0x4558	EPcMaxSize_H	EPc Max Packet Size High Register	Sets the EPc max packet size.
	0x4559	EPcMaxSize_L	EPc Max Packet Size Low Register	
	0x455a	EPcConfig_0	EPc Configuration 0 Register	Configures EPc.
	0x455b	EPcConfig_1	EPc Configuration 1 Register	
	0x455c	EPdMaxSize_H	EPd Max Packet Size High Register	Sets the EPd max packet size.
	0x455d	EPdMaxSize_L	EPd Max Packet Size Low Register	
	0x455e	EPdConfig_0	EPd Configuration 0 Register	Configures EPd.
	0x455f	EPdConfig_1	EPd Configuration 1 Register	
	0x4560–0x456f	–	–	Reserved
	0x4570	EPaStartAdrs_H	EPa FIFO Start Address High Register	Sets the FIFO start address for EPa.
	0x4571	EPaStartAdrs_L	EPa FIFO Start Address Low Register	

Peripheral	Address	Register name		Function
USB function controller (8-bit device)	0x4572	EPbStartAdrs_H	EPb FIFO Start Address High Register	Sets the FIFO start address for EPb.
	0x4573	EPbStartAdrs_L	EPb FIFO Start Address Low Register	
	0x4574	EPcStartAdrs_H	EPc FIFO Start Address High Register	Sets the FIFO start address for EPc.
	0x4575	EPcStartAdrs_L	EPc FIFO Start Address Low Register	
	0x4576	EPdStartAdrs_H	Epd FIFO Start Address High Register	Sets the FIFO start address for EPd.
	0x4577	EPdStartAdrs_L	EPd FIFO Start Address Low Register	
	0x4578–0x457f	–	–	Reserved
	0x4580	CPU_JoinRd	CPU Join FIFO Read Register	Sets up the FIFO data read conditions.
	0x4581	CPU_JoinWr	CPU Join FIFO Write Register	Sets up the FIFO data write conditions.
	0x4582	EnEPnFIFO _Access	EPn FIFO Access Enable Register	Enables the CPU_JoinRd and CPU_JoinWr registers.
	0x4583	EPnFIFOforCPU	EPn FIFO for CPU Register	EPn FIFO for accessing by the CPU.
	0x4584	EPnRdRemain_H	EPn FIFO Read Remain High Register	Indicates the remained data quantity in the FIFO.
	0x4585	EPnRdRemain_L	EPn FIFO Read Remain Low Register	
	0x4586	EPnWrRemain_H	EPn FIFO Write High Register	Indicates the free space capacity in the FIFO.
	0x4587	EPnWrRemain_L	EPn FIFO Write Low Register	
	0x4588	DescAdrs_H	Descriptor Address High Register	Specifies the FIFO start address for the descriptor reply function.
	0x4589	DescAdrs_L	Descriptor Address Low Register	
	0x458a	DescSize_H	Descriptor Size High Register	Specifies the number of data for the descriptor reply function.
	0x458b	DescSize_L	Descriptor Size Low Register	
	0x458c–0x458e	–	–	Reserved
	0x458f	DescDoor	Descriptor Door Register	Reads/writes descriptors.
	0x4590–0x45ff	–	–	Reserved
Real-time clock (8-bit device)	0x4600	RTC_INTSTAT	RTC Interrupt Status Register	Indicates RTC interrupt status.
	0x4601	RTC_INTMODE	RTC Interrupt Mode Register	Sets up RTC interrupt modes.
	0x4602	RTC_CNTL0	RTC Control 0 Register	Controls the RTC.
	0x4603	RTC_CNTL1	RTC Control 1 Register	
	0x460f	RTC_WAKEUP	RTC Wakeup Configuration Register	Sets up RTC wakeup conditions.
	0x4614	RTC_SEC	RTC Second Register	Second counter data
	0x4615	RTC_MIN	RTC Minute Register	Minute counter data
	0x4616	RTC_HOUR	RTC Hour Register	Hour counter data
	0x4617	RTC_DAY	RTC Day Register	Day counter data
	0x4628	RTC_MONTH	RTC Month Register	Month counter data
	0x4629	RTC_YEAR	RTC Year Register	Year counter data
	0x462a	RTC_WEEK	RTC Days of Week Register	Days of week counter data
	0x462b–0x46ff	–	–	Reserved
8-bit programmable timer CH.0 (8-bit device)	0x4800	PT8_CLK0	PT8 CH.0 Input Clock Select Register	Selects the count clock.
	0x4801	PT8_RLD0	PT8 CH.0 Reload Data Register	Sets reload data.
	0x4802	PT8_PTD0	PT8 CH.0 Counter Data Register	Counter data
	0x4803	PT8_CTL0	PT8 CH.0 Control Register	Sets the timer mode and starts/stops the timer.
8-bit programmable timer CH.1 (8-bit device)	0x4804	PT8_CLK1	PT8 CH.1 Input Clock Select Register	Selects the count clock.
	0x4805	PT8_RLD1	PT8 CH.1 Reload Data Register	Sets reload data.
	0x4806	PT8_PTD1	PT8 CH.1 Counter Data Register	Counter data
	0x4807	PT8_CTL1	PT8 CH.1 Control Register	Sets the timer mode and starts/stops the timer.
8-bit programmable timer CH.2 (8-bit device)	0x4808	PT8_CLK2	PT8 CH.2 Input Clock Select Register	Selects the count clock.
	0x4809	PT8_RLD2	PT8 CH.2 Reload Data Register	Sets reload data.
	0x480a	PT8_PTD2	PT8 CH.2 Counter Data Register	Counter data
	0x480b	PT8_CTL2	PT8 CH.2 Control Register	Sets the timer mode and starts/stops the timer.
8-bit programmable timer CH.3 (8-bit device)	0x480c	PT8_CLK3	PT8 CH.3 Input Clock Select Register	Selects the count clock.
	0x480d	PT8_RLD3	PT8 CH.3 Reload Data Register	Sets reload data.
	0x480e	PT8_PTD3	PT8 CH.3 Counter Data Register	Counter data
	0x480f	PT8_CTL3	PT8 CH.3 Control Register	Sets the timer mode and starts/stops the timer.
	0x4810–0x48ff	–	–	Reserved
Clock management unit (8-bit device)	0x4900	CMU_SYSCLKCTL	System Clock Control Register	Controls the system clock.
	0x4901	CMU_OSC3_WCNT	OSC3 Wait Timer Register	Sets the OSC3 wait timer for system wake-up.
	0x4902	CMU_NF	Noise Filter Control Register	Enables noise filters.
	0x4903	CMU_OSC3DIV	OSC3 Clock Divider Register	Selects a OSC3 system clock frequency.
	0x4904	–	–	Reserved
	0x4905	CMU_CMUCLK	CMU_CLK Select Register	Selects the output CMU_CLK frequency.
	0x4906	CMU_GATEDCLK0	Gated Clock Control 0 Register	Controls clock supply to peripheral modules.
	0x4907	CMU_GATEDCLK1	Gated Clock Control 1 Register	
	0x4908	CMU_GATEDCLK2	Gated Clock Control 2 Register	
	0x4909	CMU_USBWT	USB Wait Control Register	Sets the wait cycles for accessing the USB registers.
	0x490a–0x491f	–	–	Reserved
	0x4920	CMU_PROTECT	CMU Write Protect Register	Enables writing to the CMU registers (0x4900–0x4909).
	0x4921–0x49ff	–	–	Reserved

Peripheral	Address	Register name		Function
SRAM controller (16-bit device)	0x5000	EXTMEM_SWAIT	External Memory Static Wait Control Register	Sets up static wait cycles.
	0x5002	–	–	Reserved
	0x5004	EXTMEM_SIZE	External Memory Device Size Setup Register	Selects the device size (8/16 bits).
	0x5006	–	–	Reserved
	0x5008	EXTMEM_A0_BSL	External Memory Device Type Setup Register	Selects the device type (A0/BSL).
	0x500a–0x500e	–	–	Reserved
	0x5010	IRAM2_SWAIT	Internal Memory Static Wait Control Register	Sets up IRAM2 read cycle.
	0x5012	–	–	Reserved
	0x5014	MAC_WAIT	MAC Wait Control Register	Sets up the MAC wait cycle.
	0x5016	–	–	Reserved
	0x5018	RTC_WAIT	RTC Wait Control Register	Sets up RTC access cycle.
	0x501a–0x50ff	–	–	Reserved
Multi-function timer (16-bit device)	0x5200	MFT_TC	MFT Counter Data Register	Counter data
	0x5202	MFT_PRD	MFT Period Data Register	Sets period data.
	0x5204	MFT_CMP	MFT Compare Data Register	Sets compare data.
	0x5206	MFT_CTL	MFT Control Register	Sets the timer mode and starts/stops the timer.
	0x5208–0x521d	–	–	Reserved
	0x521e	MFT_IOCTL	MFT Input/Output Control Register	Controls the clock input/output.
	0x5230	MFT_IE	MFT Interrupt Enable Register	Enables the MFT interrupt.
	0x5238	MFT_IF	MFT Interrupt Flag Register	Indicates the MFT interrupt status.
	0x523a–0x527d	–	–	Reserved
	0x527e	MFT_TST	MFT Test Register	Controls the MFT test.
	0x5280–0x52ff	–	–	Reserved
I²S (16-bit device)	0x5300	I2S_CTL_OUT	I²S CH.0 Control Register	Sets output conditions.
	0x5302	I2S_CTL_IN	I²S CH.1 Control Register	Sets input conditions.
	0x5304	I2S_DV_MCLK	I²S MCLK Divide Ratio Register	Configures MCLK.
	0x5306	I2S_DV_AUDIO_CLK	I²S Audio Clock Divide Ratio Register	Configures the audio clock.
	0x5308	I2S_START	I²S Start/Stop Register	Controls/indicates I²S start/stop status.
	0x530a	I2S_FIFO_STAT	I²S FIFO Status Register	Indicates the FIFO status.
	0x530c	I2S_INT_MOD	I²S Interrupt Mode Select Register	Sets the I²S interrupt conditions.
	0x5310	I2S_FIFO_OUT	I²S CH.0 FIFO Register	L & R channel output data (16- or 24-bit access)
	0x5314	I2S_FIFO_IN	I²S CH.1 FIFO Register	L & R channel input data (16-bit access)
	0x5318–0x53ff	–	–	Reserved
Remote controller (16-bit device)	0x5400	REMC_PSC	REMC Prescaler Control Register	Sets up the REMC prescaler.
	0x5404	REMC_CFG	REMC Configuration Register	Sets the REMC modes and controls the REMC interrupt.
	0x5406	–	–	Reserved
	0x5408	REMC_CTL	REMC Control Register	Starts/stops transmission.
	0x540a	–	–	Reserved
	0x540c	REMC_CARL	REMC Carrier Load Register	Configures the carrier signal.
	0x540e	REMC_ENVL	REMC Envelope Load Register	Configures the envelope pulse width.
	0x5410	REMC_ENVC	REMC Envelope Capture Register	Input envelope pulse width
	0x5412–0x54ff	–	–	Reserved
A/D converter (16-bit device)	0x5500–0x551f	–	–	Reserved
	0x5520	AD_CLKCTL	A/D Clock Control Register	Controls A/D converter clock.
	0x5522–0x553f	–	–	Reserved
	0x5540	AD_DAT	A/D Conversion Result Register	A/D converted data
	0x5542	AD_TRIG_CH	A/D Trigger/Channel Select Register	Sets start/end channels and conversion mode.
	0x5544	AD_CTL	A/D Control/Status Register	Controls A/D converter and indicates conversion status.
	0x5546	AD_CH_STAT	A/D Channel Status Flag Register	Indicates overwrite error and conversion complete status.
	0x5548	AD_CH0_BUF	A/D CH.0 Conversion Result Buffer Register	A/D CH.0 converted data
	0x554a	AD_CH1_BUF	A/D CH.1 Conversion Result Buffer Register	A/D CH.1 converted data
	0x554c	AD_CH2_BUF	A/D CH.2 Conversion Result Buffer Register	A/D CH.2 converted data
	0x554e	AD_CH3_BUF	A/D CH.3 Conversion Result Buffer Register	A/D CH.3 converted data
	0x5550	AD_CH4_BUF	A/D CH.4 Conversion Result Buffer Register	A/D CH.4 converted data
	0x5552	AD_CH5_BUF	A/D CH.5 Conversion Result Buffer Register	A/D CH.5 converted data
	0x5554	AD_CH6_BUF	A/D CH.6 Conversion Result Buffer Register	A/D CH.6 converted data
	0x5556	AD_CH7_BUF	A/D CH.7 Conversion Result Buffer Register	A/D CH.7 converted data
	0x5558	AD_UPPER	A/D Upper Limit Value Register	Specifies A/D conversion upper limit value.
	0x555a	AD_LOWER	A/D Lower Limit Value Register	Specifies A/D conversion lower limit value.
	0x555c	AD_INTMASK	A/D Conversion Complete Interrupt Mask Register	Masks A/D conversion complete interrupt.
	0x555e	AD_ADVMODE	A/D Converter Mode Select/Internal Status Register	Selects A/D operating mode and indicates internal status and internal counter value.

Peripheral	Address	Register name		Function
Watchdog timer (16-bit device)	0x5600–0x565f	–	–	Reserved
	0x5660	WD_WP	WDT Write Protect Register	Enables WDT control registers for writing.
	0x5662	WD_EN	WDT Enable and Setup Register	Configures and starts watchdog timer.
	0x5664	WD_CMP_L	WDT Comparison Data L Register	Comparison data
	0x5666	WD_CMP_H	WDT Comparison Data H Register	
	0x5668	WD_CNT_L	WDT Count Data L Register	Watchdog timer counter data
	0x566a	WD_CNT_H	WDT Count Data H Register	
	0x566c	WD_CTL	WDT Control Register	Resets watchdog timer.
	0x566e–0x56ff	–	–	Reserved
Extended SPI (16-bit device)	0x5700	SPI_ST1	SPI CH.1 Status Register	Indicates transfer and buffer statuses.
	0x5702	SPI_TXD1	SPI CH.1 Transmit Data Register	Transmit data
	0x5704	SPI_RXD1	SPI CH.1 Receive Data Register	Receive data
	0x5706	SPI_CTL1	SPI CH.1 Control Register	Sets the SPI CH.1 mode and enables data transfer.
	0x5708	SPI_CLK1	SPI CH.1 Clock Control Register	Sets up the SPI clock.
	0x570a–0x57ff	–	–	Reserved
Flash controller (16-bit device)	0x5800	FLASH_CTL	FLASHC Control Register	Controls Flash erase/program operations.
	0x5802	FLASH_ADDR	FLASHC Sector Address Register	Sets the Flash address for erasing a sector.
	0x5804	FLASH_WAIT	FLASHC Wait Register	Sets the wait cycle for Flash read.
	0x5806–0x580f	–	–	Reserved
	0x5810	FLASH_PROT	FLASHC Protect Register	Enables Flash control registers for writing.
	0x5812	–	–	Reserved
	0x5814	TTBR_LOW	Trap Table Base Register 0	Sets the vector table address.
	0x5816	TTBR_HIGH	Trap Table Base Register 1	
	0x5818–0x58ff	–	–	Reserved
S1C17 Core I/O	0xffff90	DBRAM	Debug RAM Base Register	Indicates the debug RAM base address.

Note: Do not access the “Reserved” address in the table above and unused areas in the peripheral area that are not described in the table from the application program.

0x4020 Prescaler

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Prescaler Control Register (PSC_CTL)	**0x4020** (8 bits)	D7–2	–	reserved	–				–	–	0 when being read.
		D1	**PRUND**	Prescaler run/stop in debug mode	1	Run	0	Stop	0	R/W	
		D0	**PRUN**	Prescaler run/stop control	1	Run	0	Stop	0	R/W	

0x4100–0x4105 UART (with IrDA)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Status Register (UART_ST)	**0x4100** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6	**FER**	Framing error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D5	**PER**	Parity error flag	1 Error 0 Normal	0	R/W	
		D4	**OER**	Overrun error flag	1 Error 0 Normal	0	R/W	
		D3	**RD2B**	Second byte receive flag	1 Ready 0 Empty	0	R	
		D2	**TRBS**	Transmit busy flag	1 Busy 0 Idle	0	R	Shift register status
		D1	**RDRY**	Receive data ready flag	1 Ready 0 Empty	0	R	
		D0	**TDBE**	Transmit data buffer empty flag	1 Empty 0 Not empty	1	R	
UART Transmit Data Register (UART_TXD)	**0x4101** (8 bits)	D7–0	**TXD[7:0]**	Transmit data TXD7(6) = MSB TXD0 = LSB	0x0 to 0xff (0x7f)	0x0	R/W	
UART Receive Data Register (UART_RXD)	**0x4102** (8 bits)	D7–0	**RXD[7:0]**	Receive data in the receive data buffer RXD7(6) = MSB RXD0 = LSB	0x0 to 0xff (0x7f)	0x0	R	Older data in the buffer is read out first.
UART Mode Register (UART_MOD)	**0x4103** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**CHLN**	Character length	1 8 bits 0 7 bits	0	R/W	
		D3	**PREN**	Parity enable	1 With parity 0 No parity	0	R/W	
		D2	**PMD**	Parity mode select	1 Odd 0 Even	0	R/W	
		D1	**STPB**	Stop bit select	1 2 bits 0 1 bit	0	R/W	
		D0	**SSCK**	Input clock select	1 External 0 Internal	0	R/W	
UART Control Register (UART_CTL)	**0x4104** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6	**REIEN**	Receive error int. enable	1 Enable 0 Disable	0	R/W	
		D5	**RIEN**	Receive buffer full int. enable	1 Enable 0 Disable	0	R/W	
		D4	**TIEN**	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**RBFI**	Receive buffer full int. condition	1 2 bytes 0 1 byte	0	R/W	
		D0	**RXEN**	UART enable	1 Enable 0 Disable	0	R/W	
UART Expansion Register (UART_EXP)	**0x4105** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**IRCLK[2:0]**	IrDA receive detection clock select	IRCLK[2:0] Clock 0x7 PCLK•1/128 0x6 PCLK•1/64 0x5 PCLK•1/32 0x4 PCLK•1/16 0x3 PCLK•1/8 0x2 PCLK•1/4 0x1 PCLK•1/2 0x0 PCLK•1/1	0x0	R/W	
		D3–1	**–**	reserved	–	–	–	0 when being read.
		D0	**IRMD**	IrDA mode select	1 On 0 Off	0	R/W	

0x4200–0x4206 CLG_T16U0 Timer

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T16U0 Input Clock Select Register (CLG_T16U0 _CLK)	**0x4200** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	Timer input clock select (Prescaler output clock)	DF[3:0] / Clock 0xf reserved 0xe PCLK•1/16384 0xd PCLK•1/8192 0xc PCLK•1/4096 0xb PCLK•1/2048 0xa PCLK•1/1024 0x9 PCLK•1/512 0x8 PCLK•1/256 0x7 PCLK•1/128 0x6 PCLK•1/64 0x5 PCLK•1/32 0x4 PCLK•1/16 0x3 PCLK•1/8 0x2 PCLK•1/4 0x1 PCLK•1/2 0x0 PCLK•1/1	0x0	R/W	
CLG_T16U0 Reload Data Register (CLG_T16U0 _TR)	**0x4202** (16 bits)	D15–0	**TR[15:0]**	16-bit timer reload data TR15 = MSB TR0 = LSB	0x0 to 0xffff	0x0	R/W	
CLG_T16U0 Counter Data Register (CLG_T16U0 _TC)	**0x4204** (16 bits)	D15–0	**TC[15:0]**	16-bit timer counter data TC15 = MSB TC0 = LSB	0x0 to 0xffff	0xffff	R	
CLG_T16U0 Control Register (CLG_T16U0 _CTL)	**0x4206** (16 bits)	D15–5	**–**	reserved	–	–	–	0 when being read.
		D4	**TRMD**	Count mode select	1 One shot 0 Repeat	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PRESER**	Timer reset	1 Reset 0 Ignored	0	W	
		D0	**PRUN**	Timer run/stop control	1 Run 0 Stop	0	R/W	

0x4240–0x4246 CLG_T8S Timer

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T8S Input Clock Select Register (CLG_T8S_CLK)	**0x4240** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	Timer input clock select (Prescaler output clock)	DF[3:0] / Clock 0xf reserved 0xe PCLK•1/16384 0xd PCLK•1/8192 0xc PCLK•1/4096 0xb PCLK•1/2048 0xa PCLK•1/1024 0x9 PCLK•1/512 0x8 PCLK•1/256 0x7 PCLK•1/128 0x6 PCLK•1/64 0x5 PCLK•1/32 0x4 PCLK•1/16 0x3 PCLK•1/8 0x2 PCLK•1/4 0x1 PCLK•1/2 0x0 PCLK•1/1	0x0	R/W	
CLG_T8S Reload Data Register (CLG_T8S_TR)	**0x4242** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	8-bit timer reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
CLG_T8S Counter Data Register (CLG_T8S_TC)	**0x4244** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	8-bit timer counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0xff	R	
CLG_T8S Control Register (CLG_T8S_CTL)	**0x4246** (16 bits)	D15–5	**–**	reserved	–	–	–	0 when being read.
		D4	**TRMD**	Count mode select	1 One shot / 0 Repeat	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PRESER**	Timer reset	1 Reset / 0 Ignored	0	W	
		D0	**PRUN**	Timer run/stop control	1 Run / 0 Stop	0	R/W	

0x4260–0x4266 CLG_T8I Timer

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CLG_T8I Input Clock Select Register (CLG_T8I_CLK)	**0x4260** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	Timer input clock select (Prescaler output clock)	DF[3:0] / Clock 0xf reserved 0xe PCLK•1/16384 0xd PCLK•1/8192 0xc PCLK•1/4096 0xb PCLK•1/2048 0xa PCLK•1/1024 0x9 PCLK•1/512 0x8 PCLK•1/256 0x7 PCLK•1/128 0x6 PCLK•1/64 0x5 PCLK•1/32 0x4 PCLK•1/16 0x3 PCLK•1/8 0x2 PCLK•1/4 0x1 PCLK•1/2 0x0 PCLK•1/1	0x0	R/W	
CLG_T8I Reload Data Register (CLG_T8I_TR)	**0x4262** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	8-bit timer reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
CLG_T8I Counter Data Register (CLG_T8I_TC)	**0x4264** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	8-bit timer counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0xff	R	
CLG_T8I Control Register (CLG_T8I_CTL)	**0x4266** (16 bits)	D15–5	**–**	reserved	–	–	–	0 when being read.
		D4	**TRMD**	Count mode select	1 One shot 0 Repeat	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PRESER**	Timer reset	1 Reset 0 Ignored	0	W	
		D0	**PRUN**	Timer run/stop control	1 Run 0 Stop	0	R/W	

0x42e0–0x42f4 Interrupt Controller

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Additional Interrupt Flag Register (ITC_AIFLG)	**0x42e0** (16 bits)	D15	**–**	reserved	–	–	–	0 when being read.
		D14	**AIFT14**	REMC interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D13	**AIFT13**	I²S receive interrupt flag		0	R/W	
		D12	**AIFT12**	I²S transmit interrupt flag		0	R/W	
		D11	**AIFT11**	USB interrupt flag		0	R/W	
		D10	**AIFT10**	SPI CH.1 interrupt flag		0	R/W	
		D9	**–**	reserved	–	–	–	0 when being read.
		D8	**AIFT8**	PT8 CH.3 interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D7	**AIFT7**	PT8 CH.2 interrupt flag		0	R/W	
		D6	**AIFT6**	PT8 CH.1 interrupt flag		0	R/W	
		D5	**AIFT5**	PT8 CH.0 interrupt flag		0	R/W	
		D4	**AIFT4**	RTC interrupt flag		0	R/W	
		D3	**AIFT3**	ADC end-of-conversion interrupt flag		0	R/W	
		D2	**AIFT2**	ADC out-of-range interrupt flag		0	R/W	
		D1	**–**	reserved	–	–	–	0 when being read.
		D0	**AIFT0**	MFT interrupt flag	1 Occurred / 0 Not occurred	0	R/W	Reset by writing 1.
Additional Interrupt Enable Register (ITC_AEN)	**0x42e2** (16 bits)	D15	**–**	reserved	–	–	–	0 when being read.
		D14	**AIEN14**	REMC interrupt enable	1 Enable / 0 Disable	0	R/W	
		D13	**AIEN13**	I²S receive interrupt enable		0	R/W	
		D12	**AIEN12**	I²S transmit interrupt enable		0	R/W	
		D11	**AIEN11**	USB interrupt enable		0	R/W	
		D10	**AIEN10**	SPI CH.1 interrupt enable		0	R/W	
		D9	**–**	reserved	–	–	–	0 when being read.
		D8	**AIEN8**	PT8 CH.3 interrupt enable	1 Enable / 0 Disable	0	R/W	
		D7	**AIEN7**	PT8 CH.2 interrupt enable		0	R/W	
		D6	**AIEN6**	PT8 CH.1 interrupt enable		0	R/W	
		D5	**AIEN5**	PT8 CH.0 interrupt enable		0	R/W	
		D4	**AIEN4**	RTC interrupt enable		0	R/W	
		D3	**AIEN3**	ADC end-of-conversion interrupt enable		0	R/W	
		D2	**AIEN2**	ADC out-of-range interrupt enable		0	R/W	
		D1	**–**	reserved	–	–	–	0 when being read.
		D0	**AIEN0**	MFT interrupt enable	1 Enable / 0 Disable	0	R/W	
Additional Interrupt Level Setup Register 0 (ITC_AILV0)	**0x42e6** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV0[2:0]**	MFT interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 1 (ITC_AILV1)	**0x42e8** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV3[2:0]**	ADC end-of-conversion interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV2[2:0]**	ADC out-of-range interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 2 (ITC_AILV2)	**0x42ea** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV5[2:0]**	PT8 CH.0 interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV4[2:0]**	RTC interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 3 (ITC_AILV3)	**0x42ec** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV7[2:0]**	PT8 CH.2 interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV6[2:0]**	PT8 CH.1 interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 4 (ITC_AILV4)	**0x42ee** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV8[2:0]**	PT8 CH.3 interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 5 (ITC_AILV5)	**0x42f0** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV11[2:0]**	USB interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV10[2:0]**	SPI CH.1 interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 6 (ITC_AILV6)	**0x42f2** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**AILV13[2:0]**	I²S receive interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV12[2:0]**	I²S transmit interrupt level	0 to 7	0x0	R/W	
Additional Interrupt Level Setup Register 7 (ITC_AILV7)	**0x42f4** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**AILV14[2:0]**	REMC interrupt level	0 to 7	0x0	R/W	

0x4300–0x430a Interrupt Controller

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Interrupt Flag Register (ITC_IFLG)	**0x4300** (16 bits)	D15	**IIFT7**	I²C interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D14	**IIFT6**	SPI CH.0 interrupt flag		0	R/W	
		D13	**–**	reserved	–	–	–	0 when being read.
		D12	**IIFT4**	UART interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D11	**IIFT3**	CLG_T8I timer interrupt flag		0	R/W	
		D10	**IIFT2**	CLG_T8S timer interrupt flag		0	R/W	
		D9	**–**	reserved	–	–	–	0 when being read.
		D8	**IIFT0**	CLG_T16U0 timer interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D7	**EIFT7**	Port interrupt 7 flag		0	R/W	Reset by writing 1 in pulse trigger mode. Cannot be reset by software in level trigger mode.
		D6	**EIFT6**	Port interrupt 6 flag		0	R/W	
		D5	**EIFT5**	Port interrupt 5 flag		0	R/W	
		D4	**EIFT4**	Port interrupt 4 flag		0	R/W	
		D3	**EIFT3**	Port interrupt 3 flag		0	R/W	
		D2	**EIFT2**	Port interrupt 2 flag		0	R/W	
		D1	**EIFT1**	Port interrupt 1 flag		0	R/W	
		D0	**EIFT0**	Port interrupt 0 flag		0	R/W	
Interrupt Enable Register (ITC_EN)	**0x4302** (16 bits)	D15	**IIEN7**	I²C interrupt enable	1 Enable / 0 Disable	0	R/W	
		D14	**IIEN6**	SPI CH.0 interrupt enable		0	R/W	
		D13	**–**	reserved	–	–	–	0 when being read.
		D12	**IIEN4**	UART interrupt enable	1 Enable / 0 Disable	0	R/W	
		D11	**IIEN3**	CLG_T8I timer interrupt enable		0	R/W	
		D10	**IIEN2**	CLG_T8S timer interrupt enable		0	R/W	
		D9	**–**	reserved	–	–	–	0 when being read.
		D8	**IIEN0**	CLG_T16U0 timer interrupt enable	1 Enable / 0 Disable	0	R/W	
		D7	**EIEN7**	Port interrupt 7 enable		0	R/W	
		D6	**EIEN6**	Port interrupt 6 enable		0	R/W	
		D5	**EIEN5**	Port interrupt 5 enable		0	R/W	
		D4	**EIEN4**	Port interrupt 4 enable		0	R/W	
		D3	**EIEN3**	Port interrupt 3 enable		0	R/W	
		D2	**EIEN2**	Port interrupt 2 enable		0	R/W	
		D1	**EIEN1**	Port interrupt 1 enable		0	R/W	
		D0	**EIEN0**	Port interrupt 0 enable		0	R/W	
ITC Control Register (ITC_CTL)	**0x4304** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**ITEN**	ITC enable	1 Enable / 0 Disable	0	R/W	
External Interrupt Level Setup Register 0 (ITC_ELV0)	**0x4306** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13	**EITP1**	Port interrupt 1 trigger polarity	1 Positive / 0 Negative	0	R/W	
		D12	**EITG1**	Port interrupt 1 trigger mode	1 Level / 0 Pulse	0	R/W	
		D11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**EILV1[2:0]**	Port interrupt 1 level	0 to 7	0x0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**EITP0**	Port interrupt 0 trigger polarity	1 Positive / 0 Negative	0	R/W	
		D4	**EITG0**	Port interrupt 0 trigger mode	1 Level / 0 Pulse	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**EILV0[2:0]**	Port interrupt 0 level	0 to 7	0x0	R/W	
External Interrupt Level Setup Register 1 (ITC_ELV1)	**0x4308** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13	**EITP3**	Port interrupt 3 trigger polarity	1 Positive / 0 Negative	0	R/W	
		D12	**EITG3**	Port interrupt 3 trigger mode	1 Level / 0 Pulse	0	R/W	
		D11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**EILV3[2:0]**	Port interrupt 3 level	0 to 7	0x0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**EITP2**	Port interrupt 2 trigger polarity	1 Positive / 0 Negative	0	R/W	
		D4	**EITG2**	Port interrupt 2 trigger mode	1 Level / 0 Pulse	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**EILV2[2:0]**	Port interrupt 2 level	0 to 7	0x0	R/W	
External Interrupt Level Setup Register 2 (ITC_ELV2)	**0x430a** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13	**EITP5**	Port interrupt 5 trigger polarity	1 Positive / 0 Negative	0	R/W	
		D12	**EITG5**	Port interrupt 5 trigger mode	1 Level / 0 Pulse	0	R/W	
		D11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**EILV5[2:0]**	Port interrupt 5 level	0 to 7	0x0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**EITP4**	Port interrupt 4 trigger polarity	1 Positive / 0 Negative	0	R/W	
		D4	**EITG4**	Port interrupt 4 trigger mode	1 Level / 0 Pulse	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**EILV4[2:0]**	Port interrupt 4 level	0 to 7	0x0	R/W	

0x430c–0x4314 Interrupt Controller

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Interrupt Level Setup Register 3 (ITC_ELV3)	**0x430c** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13	**EITP7**	Port interrupt 7 trigger polarity	1 Positive 0 Negative	0	R/W	
		D12	**EITG7**	Port interrupt 7 trigger mode	1 Level 0 Pulse	0	R/W	
		D11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**EILV7[2:0]**	Port interrupt 7 level	0 to 7	0x0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**EITP6**	Port interrupt 6 trigger polarity	1 Positive 0 Negative	0	R/W	
		D4	**EITG6**	Port interrupt 6 trigger mode	1 Level 0 Pulse	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**EILV6[2:0]**	Port interrupt 6 level	0 to 7	0x0	R/W	
Internal Interrupt Level Setup Register 0 (ITC_ILV0)	**0x430e** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**IILV0[2:0]**	CLG_T16U0 timer interrupt level	0 to 7	0x0	R/W	
Internal Interrupt Level Setup Register 1 (ITC_ILV1)	**0x4310** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**IILV3[2:0]**	CLG_T8I timer interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**IILV2[2:0]**	CLG_T8S timer interrupt level	0 to 7	0x0	R/W	
Internal Interrupt Level Setup Register 2 (ITC_ILV2)	**0x4312** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**IILV4[2:0]**	UART interrupt level	0 to 7	0x0	R/W	
Internal Interrupt Level Setup Register 3 (ITC_ILV3)	**0x4314** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10–8	**IILV7[2:0]**	I^2C interrupt level	0 to 7	0x0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**IILV6[2:0]**	SPI CH.0 interrupt level	0 to 7	0x0	R/W	

0x4320–0x4326 SPI

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SPI CH.0 Status Register (SPI_ST0)	**0x4320** (16 bits)	D15–3	**–**	reserved	–				–	–	0 when being read.
		D2	**SPBSY**	Transfer busy flag (master)	1	Busy	0	Idle	0	R	
				ss signal low flag (slave)	1	ss = L	0	ss = H			
		D1	**SPRBF**	Receive data buffer full flag	1	Full	0	Not full	0	R	
		D0	**SPTBE**	Transmit data buffer empty flag	1	Empty	0	Not empty	1	R	
SPI CH.0 Transmit Data Register (SPI_TXD0)	**0x4322** (16 bits)	D15–8	**–**	reserved	–				–	–	0 when being read.
		D7–0	**SPTDB[7:0]**	SPI CH.0 transmit data buffer SPTDB7 = MSB SPTDB0 = LSB	0x0 to 0xff				0x0	R/W	
SPI CH.0 Receive Data Register (SPI_RXD0)	**0x4324** (16 bits)	D15–8	**–**	reserved	–				–	–	0 when being read.
		D7–0	**SPRDB[7:0]**	SPI CH.0 receive data buffer SPRDB7 = MSB SPRDB0 = LSB	0x0 to 0xff				0x0	R	
SPI CH.0 Control Register (SPI_CTL0)	**0x4326** (16 bits)	D15–6	**–**	reserved	–				–	–	0 when being read.
		D5	**SPRIE**	Receive data buffer full int. enable	1	Enable	0	Disable	0	R/W	
		D4	**SPTIE**	Transmit data buffer empty int. enable	1	Enable	0	Disable	0	R/W	
		D3	**CPHA**	Clock phase select	1	Data out	0	Data in	0	R/W	These bits must be set before setting SPEN to 1.
		D2	**CPOL**	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	**MSSL**	Master/slave mode select	1	Master	0	Slave	0	R/W	
		D0	**SPEN**	SPI CH.0 enable	1	Enable	0	Disable	0	R/W	

0x4340–0x4346 I²C

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²C Enable Register (I2C_EN)	**0x4340** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**I2CEN**	I²C enable	1 Enable 0 Disable	0	R/W	
I²C Control Register (I2C_CTL)	**0x4342** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9	**RBUSY**	Receive busy flag	1 Busy 0 Idle	0	R	
		D8	**TBUSY**	Transmit busy flag	1 Busy 0 Idle	0	R	
		D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**NSERM**	Noise remove on/off	1 On 0 Off	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**STP**	Stop control	1 Stop 0 Ignored	0	R/W	
		D0	**STRT**	Start control	1 Start 0 Ignored	0	R/W	
I²C Data Register (I2C_DAT)	**0x4344** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11	**RBRDY**	Receive buffer ready	1 Ready 0 Empty	0	R	
		D10	**RXE**	Receive execution	1 Receive 0 Ignored	0	R/W	
		D9	**TXE**	Transmit execution	1 Transmit 0 Ignored	0	R/W	
		D8	**RTACK**	Receive/transmit ACK	1 Error 0 ACK	0	R/W	
		D7–0	**RTDT[7:0]**	Receive/transmit data RTDT7 = MSB RTDT0 = LSB	0x0 to 0xff	0x0	R/W	
I²C Interrupt Control Register (I2C_ICTL)	**0x4346** (16 bits)	D15–2	**–**	reserved	–	–	–	0 when being read.
		D1	**RINTE**	Receive interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**TINTE**	Transmit interrupt enable	1 Enable 0 Disable	0	R/W	

0x4400–0x4413 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
P0 Port Input Data Register (P0_DAT)	**0x4400** (8 bits)	D7–0	**P0D[7:0]**	P0[7:0] port input data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P1 Port Input /Output Data Register (P1_DAT)	**0x4402** (8 bits)	D7	**–**	reserved	–				–	–	0 when being read.
		D6–0	**P1D[6:0]**	P1[6:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P1 I/O Control Register (P1_IOC)	**0x4403** (8 bits)	D7	**–**	reserved	–				–	–	0 when being read.
		D6–0	**IOC1[6:0]**	P1[6:0] I/O control	1	Output	0	Input	0	R/W	
P2 Port Input /Output Data Register (P2_DAT)	**0x4404** (8 bits)	D7–0	**P2D[7:0]**	P2[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P2 I/O Control Register (P2_IOC)	**0x4405** (8 bits)	D7–0	**IOC2[7:0]**	P2[7:0] I/O control	1	Output	0	Input	0	R/W	
P3 Port Input /Output Data Register (P3_DAT)	**0x4406** (8 bits)	D7–0	**P3D[7:0]**	P3[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P3 I/O Control Register (P3_IOC)	**0x4407** (8 bits)	D7–0	**IOC3[7:0]**	P3[7:0] I/O control	1	Output	0	Input	0	R/W	
P4 Port Input /Output Data Register (P4_DAT)	**0x4408** (8 bits)	D7–6	**–**	reserved	–				–	–	0 when being read.
		D5–0	**P4D[5:0]**	P4[5:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P4 I/O Control Register (P4_IOC)	**0x4409** (8 bits)	D7–6	**–**	reserved	–				–	–	0 when being read.
		D5–0	**IOC4[5:0]**	P4[5:0] I/O control	1	Output	0	Input	0	R/W	
P5 Port Input /Output Data Register (P5_DAT)	**0x440a** (8 bits)	D7–0	**P5D[7:0]**	P5[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P5 I/O Control Register (P5_IOC)	**0x440b** (8 bits)	D7–0	**IOC5[7:0]**	P5[7:0] I/O control	1	Output	0	Input	0	R/W	
P6 Port Input /Output Data Register (P6_DAT)	**0x440c** (8 bits)	D7–0	**P6D[7:0]**	P6[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P6 I/O Control Register (P6_IOC)	**0x440d** (8 bits)	D7–0	**IOC6[7:0]**	P6[7:0] I/O control	1	Output	0	Input	0	R/W	
P7 Port Input /Output Data Register (P7_DAT)	**0x440e** (8 bits)	D7–0	**P7D[7:0]**	P7[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P7 I/O Control Register (P7_IOC)	**0x440f** (8 bits)	D7–0	**IOC7[7:0]**	P7[7:0] I/O control	1	Output	0	Input	0	R/W	
P8 Port Input /Output Data Register (P8_DAT)	**0x4410** (8 bits)	D7	**–**	reserved	–				–	–	0 when being read.
		D6–0	**P8D[6:0]**	P8[6:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P8 I/O Control Register (P8_IOC)	**0x4411** (8 bits)	D7	**–**	reserved	–				–	–	0 when being read.
		D6–0	**IOC8[6:0]**	P8[6:0] I/O control	1	Output	0	Input	0	R/W	
P9 Port Input /Output Data Register (P9_DAT)	**0x4412** (8 bits)	D7–0	**P9D[7:0]**	P9[7:0] port input/output data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
P9 I/O Control Register (P9_IOC)	**0x4413** (8 bits)	D7–0	**IOC9[7:0]**	P9[7:0] I/O control	1	Output	0	Input	0	R/W	

0x4414–0x4422 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PA Port Input /Output Data Register (PA_DAT)	0x4414 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–0	PAD[6:0]	PA[6:0] port input/output data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
PA I/O Control Register (PA_IOC)	0x4415 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–0	IOCA[6:0]	PA[6:0] I/O control	1 Output 0 Input	0	R/W	
PB Port Input /Output Data Register (PB_DAT)	0x4416 (8 bits)	D7–0	PBD[7:0]	PB[7:0] port input/output data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
PB I/O Control Register (PB_IOC)	0x4417 (8 bits)	D7–0	IOCB[7:0]	PB[7:0] I/O control	1 Output 0 Input	0	R/W	
PC Port Input /Output Data Register (PC_DAT)	0x4418 (8 bits)	D7–0	PCD[7:0]	PC[7:0] port input/output data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: The initial value depends on the external pin status.
PC I/O Control Register (PC_IOC)	0x4419 (8 bits)	D7–0	IOCC[7:0]	PC[7:0] I/O control	1 Output 0 Input	0	R/W	
P00–P03 Port Function Select Register (P0_03_CFP)	0x4420 (8 bits)	D7–6	CFP03[1:0]	P03 port function select	CFP03[1:0] Function 0x3–0x2 reserved 0x1 AIN3 0x0 P03	0x0	R/W	
		D5–4	CFP02[1:0]	P02 port function select	CFP02[1:0] Function 0x3–0x2 reserved 0x1 AIN2 0x0 P02	0x0	R/W	
		D3–2	CFP01[1:0]	P01 port function select	CFP01[1:0] Function 0x3–0x2 reserved 0x1 AIN1 0x0 P01	0x0	R/W	
		D1–0	CFP00[1:0]	P00 port function select	CFP00[1:0] Function 0x3–0x2 reserved 0x1 AIN0 0x0 P00	0x0	R/W	
P04–P07 Port Function Select Register (P0_47_CFP)	0x4421 (8 bits)	D7–6	CFP07[1:0]	P07 port function select	CFP07[1:0] Function 0x3–0x2 reserved 0x1 AIN7 0x0 P07	0x0	R/W	
		D5–4	CFP06[1:0]	P06 port function select	CFP06[1:0] Function 0x3–0x2 reserved 0x1 AIN6 0x0 P06	0x0	R/W	
		D3–2	CFP05[1:0]	P05 port function select	CFP05[1:0] Function 0x3–0x2 reserved 0x1 AIN5 0x0 P05	0x0	R/W	
		D1–0	CFP04[1:0]	P04 port function select	CFP04[1:0] Function 0x3–0x2 reserved 0x1 AIN4 0x0 P04	0x0	R/W	
P10–P13 Port Function Select Register (P1_03_CFP)	0x4422 (8 bits)	D7–6	CFP13[1:0]	P13 port function select	CFP13[1:0] Function 0x3 SPI_SDI0 0x2 #SPI_SSI1 0x1 #SPI_SSI0 0x0 P13	0x0	R/W	
		D5–4	CFP12[1:0]	P12 port function select	CFP12[1:0] Function 0x3–0x2 reserved 0x1 SPI_SCK0 0x0 P12	0x0	R/W	
		D3–2	CFP11[1:0]	P11 port function select	CFP11[1:0] Function 0x3–0x2 reserved 0x1 SPI_SDO0 0x0 P11	0x0	R/W	
		D1–0	CFP10[1:0]	P10 port function select	CFP10[1:0] Function 0x3–0x2 reserved 0x1 SPI_SDI0 0x0 P10	0x0	R/W	

0x4423–0x4426 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P14–P16 Port Function Select Register (P1_46_CFP)	**0x4423** (8 bits)	D7–6	–	reserved	–		–	–	0 when being read.
		D5–4	**CFP16[1:0]**	P16 port function select	CFP16[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SCK1			
					0x0	P16			
		D3–2	**CFP15[1:0]**	P15 port function select	CFP15[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SDO1			
					0x0	P15			
		D1–0	**CFP14[1:0]**	P14 port function select	CFP14[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	SPI_SDI1			
					0x0	P14			
P20–P23 Port Function Select Register (P2_03_CFP)	**0x4424** (8 bits)	D7–6	**CFP23[1:0]**	P23 port function select	CFP23[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_MCLK0			
					0x0	P23			
		D5–4	**CFP22[1:0]**	P22 port function select	CFP22[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SCK0			
					0x0	P22			
		D3–2	**CFP21[1:0]**	P21 port function select	CFP21[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_WS0			
					0x0	P21			
		D1–0	**CFP20[1:0]**	P20 port function select	CFP20[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SDO0			
					0x0	P20			
P24–P27 Port Function Select Register (P2_47_CFP)	**0x4425** (8 bits)	D7–6	**CFP27[1:0]**	P27 port function select	CFP27[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_MCLK1			
					0x0	P27			
		D5–4	**CFP26[1:0]**	P26 port function select	CFP26[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SCK1			
					0x0	P26			
		D3–2	**CFP25[1:0]**	P25 port function select	CFP25[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_WS1			
					0x0	P25			
		D1–0	**CFP24[1:0]**	P24 port function select	CFP24[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2S_SDI1			
					0x0	P24			
P30–P33 Port Function Select Register (P3_03_CFP)	**0x4426** (8 bits)	D7–6	**CFP33[1:0]**	P33 port function select	CFP33[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#TM0			
					0x1	#SMRD			
					0x0	P33			
		D5–4	**CFP32[1:0]**	P32 port function select	CFP32[1:0]	Function	0x0	R/W	
					0x3	CMU_CLK			
					0x2	#WDT_NMI			
					0x1	WDT_CLK			
					0x0	P32			
		D3–2	**CFP31[1:0]**	P31 port function select	CFP31[1:0]	Function	0x0	R/W	
					0x3	#ADTRG			
					0x2	#TM0			
					0x1	reserved			
					0x0	P31			
		D1–0	**CFP30[1:0]**	P30 port function select	CFP30[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	TM0			
					0x0	P30			

0x4427–0x442a GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P34–P37 Port Function Select Register (P3_47_CFP)	**0x4427** (8 bits)	D7–6	**CFP37[1:0]**	P37 port function select	CFP37[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	P37			
					0x0	DST2			
		D5–4	**CFP36[1:0]**	P36 port function select	CFP36[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	P36			
					0x0	DSIO			
		D3–2	**CFP35[1:0]**	P35 port function select	CFP35[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	P35			
					0x0	DCLK			
		D1–0	**CFP34[1:0]**	P34 port function select	CFP34[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#SMWR			
					0x0	P34			
P40–P43 Port Function Select Register (P4_03_CFP)	**0x4428** (8 bits)	D7–6	**CFP43[1:0]**	P43 port function select	CFP43[1:0]	Function	0x0	R/W	
					0x3	PWMPRT0			
					0x2	REMC_IN			
					0x1	reserved			
					0x0	P43			
		D5–4	**CFP42[1:0]**	P42 port function select	CFP42[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	#SCLK0			
					0x0	P42			
		D3–2	**CFP41[1:0]**	P41 port function select	CFP41[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#SMWR			
					0x1	SOUT0			
					0x0	P41			
		D1–0	**CFP40[1:0]**	P40 port function select	CFP40[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#SMRD			
					0x1	SIN0			
					0x0	P40			
P44–P45 Port Function Select Register (P4_45_CFP)	**0x4429** (8 bits)	D7–4	**–**	reserved	–		–	–	0 when being read.
		D3–2	**CFP45[1:0]**	P45 port function select	CFP45[1:0]	Function	0x0	R/W	
					0x3	#WAIT			
					0x2–0x1	reserved			
					0x0	P45			
		D1–0	**CFP44[1:0]**	P44 port function select	CFP44[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	REMC_OUT			
					0x1	reserved			
					0x0	P44			
P50–P53 Port Function Select Register (P5_03_CFP)	**0x442a** (8 bits)	D7–6	**CFP53[1:0]**	P53 port function select	CFP53[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	REMC_OUT			
					0x0	P53			
		D5–4	**CFP52[1:0]**	P52 port function select	CFP52[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#TM0			
					0x1	REMC_IN			
					0x0	P52			
		D3–2	**CFP51[1:0]**	P51 port function select	CFP51[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	I2C_SCL			
					0x0	P51			
		D1–0	**CFP50[1:0]**	P50 port function select	CFP50[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	I2C_SDA			
					0x0	P50			

0x442b–0x442e GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P54–P57 Port Function Select Register (P5_47_CFP)	**0x442b** (8 bits)	D7–6	**CFP57[1:0]**	P57 port function select	CFP57[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPDRDY			
					0x0	P57			
		D5–4	**CFP56[1:0]**	P56 port function select	CFP56[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#TM0			
					0x1	FPSHIFT			
					0x0	P56			
		D3–2	**CFP55[1:0]**	P55 port function select	CFP55[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPLINE			
					0x0	P55			
		D1–0	**CFP54[1:0]**	P54 port function select	CFP54[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	FPFRAME			
					0x0	P54			
P60–P63 Port Function Select Register (P6_03_CFP)	**0x442c** (8 bits)	D7–6	**CFP63[1:0]**	P63 port function select	CFP63[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A3			
					0x0	P63			
		D5–4	**CFP62[1:0]**	P62 port function select	CFP62[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A2			
					0x0	P62			
		D3–2	**CFP61[1:0]**	P61 port function select	CFP61[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A1			
					0x0	P61			
		D1–0	**CFP60[1:0]**	P60 port function select	CFP60[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A0/#BSL			
					0x0	P60			
P64–P67 Port Function Select Register (P6_47_CFP)	**0x442d** (8 bits)	D7–6	**CFP67[1:0]**	P67 port function select	CFP67[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A7			
					0x0	P67			
		D5–4	**CFP66[1:0]**	P66 port function select	CFP66[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A6			
					0x0	P66			
		D3–2	**CFP65[1:0]**	P65 port function select	CFP65[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A5			
					0x0	P65			
		D1–0	**CFP64[1:0]**	P64 port function select	CFP64[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A4			
					0x0	P64			
P70–P73 Port Function Select Register (P7_03_CFP)	**0x442e** (8 bits)	D7–6	**CFP73[1:0]**	P73 port function select	CFP73[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A11			
					0x0	P73			
		D5–4	**CFP72[1:0]**	P72 port function select	CFP72[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A10			
					0x0	P72			
		D3–2	**CFP71[1:0]**	P71 port function select	CFP71[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A9			
					0x0	P71			
		D1–0	**CFP70[1:0]**	P70 port function select	CFP70[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A8			
					0x0	P70			

0x442f–0x4432 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P74–P77 Port Function Select Register (P7_47_CFP)	**0x442f** (8 bits)	D7–6	**CFP77[1:0]**	P77 port function select	CFP77[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A15			
					0x0	P77			
		D5–4	**CFP76[1:0]**	P76 port function select	CFP76[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A14			
					0x0	P76			
		D3–2	**CFP75[1:0]**	P75 port function select	CFP75[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A13			
					0x0	P75			
		D1–0	**CFP74[1:0]**	P74 port function select	CFP74[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A12			
					0x0	P74			
P80–P83 Port Function Select Register (P8_03_CFP)	**0x4430** (8 bits)	D7–6	**CFP83[1:0]**	P83 port function select	CFP83[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A19			
					0x0	P83			
		D5–4	**CFP82[1:0]**	P82 port function select	CFP82[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A18			
					0x0	P82			
		D3–2	**CFP81[1:0]**	P81 port function select	CFP81[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A17			
					0x0	P81			
		D1–0	**CFP80[1:0]**	P80 port function select	CFP80[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A16			
					0x0	P80			
P84–P86 Port Function Select Register (P8_46_CFP)	**0x4431** (8 bits)	D7–6	**–**	reserved	–		–	–	0 when being read.
		D5–4	**CFP86[1:0]**	P86 port function select	CFP86[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	CMU_CLK			
					0x1	A22			
					0x0	P86			
		D3–2	**CFP85[1:0]**	P85 port function select	CFP85[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A21			
					0x0	P85			
		D1–0	**CFP84[1:0]**	P84 port function select	CFP84[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	A20			
					0x0	P84			
P90–P93 Port Function Select Register (P9_03_CFP)	**0x4432** (8 bits)	D7–6	**CFP93[1:0]**	P93 port function select	CFP93[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D3			
					0x0	P93			
		D5–4	**CFP92[1:0]**	P92 port function select	CFP92[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D2			
					0x0	P92			
		D3–2	**CFP91[1:0]**	P91 port function select	CFP91[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D1			
					0x0	P91			
		D1–0	**CFP90[1:0]**	P90 port function select	CFP90[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D0			
					0x0	P90			

0x4433–0x4436 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P94–P97 Port Function Select Register (P9_47_CFP)	**0x4433** (8 bits)	D7–6	**CFP97[1:0]**	P97 port function select	CFP97[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D7			
					0x0	P97			
		D5–4	**CFP96[1:0]**	P96 port function select	CFP96[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D6			
					0x0	P96			
		D3–2	**CFP95[1:0]**	P95 port function select	CFP95[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D5			
					0x0	P95			
		D1–0	**CFP94[1:0]**	P94 port function select	CFP94[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D4			
					0x0	P94			
PA0–PA3 Port Function Select Register (PA_03_CFP)	**0x4434** (8 bits)	D7–6	**CFPA3[1:0]**	PA3 port function select	CFPA3[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE3			
					0x0	PA3			
		D5–4	**CFPA2[1:0]**	PA2 port function select	CFPA2[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE2			
					0x0	PA2			
		D3–2	**CFPA1[1:0]**	PA1 port function select	CFPA1[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE1			
					0x0	PA1			
		D1–0	**CFPA0[1:0]**	PA0 port function select	CFPA0[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#CE0			
					0x0	PA0			
PA4–PA6 Port Function Select Register (PA_46_CFP)	**0x4435** (8 bits)	D7–6	**–**	reserved	–		–	–	0 when being read.
		D5–4	**CFPA6[1:0]**	PA6 port function select	CFPA6[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#WRH/#BSH			
					0x0	PA6			
		D3–2	**CFPA5[1:0]**	PA5 port function select	CFPA5[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#WRL			
					0x0	PA5			
		D1–0	**CFPA4[1:0]**	PA4 port function select	CFPA4[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	#RD			
					0x0	PA4			
PB0–PB3 Port Function Select Register (PB_03_CFP)	**0x4436** (8 bits)	D7–6	**CFPB3[1:0]**	PB3 port function select	CFPB3[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D11			
					0x0	PB3			
		D5–4	**CFPB2[1:0]**	PB2 port function select	CFPB2[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D10			
					0x0	PB2			
		D3–2	**CFPB1[1:0]**	PB1 port function select	CFPB1[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D9			
					0x0	PB1			
		D1–0	**CFPB0[1:0]**	PB0 port function select	CFPB0[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D8			
					0x0	PB0			

0x4437–0x4440 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PB4–PB7 Port Function Select Register (PB_47_CFP)	**0x4437** (8 bits)	D7–6	**CFPB7[1:0]**	PB7 port function select	CFPB7[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D15			
					0x0	PB7			
		D5–4	**CFPB6[1:0]**	PB6 port function select	CFPB6[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D14			
					0x0	PB6			
		D3–2	**CFPB5[1:0]**	PB5 port function select	CFPB5[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D13			
					0x0	PB5			
		D1–0	**CFPB4[1:0]**	PB4 port function select	CFPB4[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	D12			
					0x0	PB4			
PC0–PC3 Port Function Select Register (PC_03_CFP)	**0x4438** (8 bits)	D7–6	**CFPC3[1:0]**	PC3 port function select	CFPC3[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	PWMPRT0			
					0x1	FPDAT3			
					0x0	PC3			
		D5–4	**CFPC2[1:0]**	PC2 port function select	CFPC2[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#ADTRG			
					0x1	FPDAT2			
					0x0	PC2			
		D3–2	**CFPC1[1:0]**	PC1 port function select	CFPC1[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#WDT_NMI			
					0x1	FPDAT1			
					0x0	PC1			
		D1–0	**CFPC0[1:0]**	PC0 port function select	CFPC0[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	CMU_CLK			
					0x1	FPDAT0			
					0x0	PC0			
PC4–PC7 Port Function Select Register (PC_47_CFP)	**0x4439** (8 bits)	D7–6	**CFPC7[1:0]**	PC7 port function select	CFPC7[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPDAT7			
					0x0	PC7			
		D5–4	**CFPC6[1:0]**	PC6 port function select	CFPC6[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPDAT6			
					0x0	PC6			
		D3–2	**CFPC5[1:0]**	PC5 port function select	CFPC5[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPDAT5			
					0x0	PC5			
		D1–0	**CFPC4[1:0]**	PC4 port function select	CFPC4[1:0]	Function	0x0	R/W	
					0x3–0x2	reserved			
					0x1	FPDAT4			
					0x0	PC4			
Port Input Interrupt 0 Select Register (PINTSEL0)	**0x4440** (8 bits)	D7–4	**PINT0[3:0]**	Interrupt port select	PINT0[3:0]	Function	0xf	R/W	
					0xc	PC0			
					0xb	PB0			
					0xa	PA0			
					0x9	P90			
					0x8	P80			
					0x7	P70			
					0x6	P60			
					0x5	P50			
					0x4	P40			
					0x3	P30			
					0x2	P20			
					0x1	P10			
					0x0	P00			
		D3–0	–	reserved	–		X	R	

0x4441–0x4445 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Port Input Interrupt 1 Select Register (PINTSEL1)	**0x4441** (8 bits)	D7–4	**PINT1[3:0]**	Interrupt port select	PINT1[3:0]	Function	0xf	R/W	
					0xc	PC1			
					0xb	PB1			
					0xa	PA1			
					0x9	P91			
					0x8	P81			
					0x7	P71			
					0x6	P61			
					0x5	P51			
					0x4	P41			
					0x3	P31			
					0x2	P21			
					0x1	P11			
					0x0	P01			
		D3–0	–	reserved	–		X	R	
Port Input Interrupt 2 Select Register (PINTSEL2)	**0x4442** (8 bits)	D7–4	**PINT2[3:0]**	Interrupt port select	PINT2[3:0]	Function	0xf	R/W	
					0xc	PC2			
					0xb	PB2			
					0xa	PA2			
					0x9	P92			
					0x8	P82			
					0x7	P72			
					0x6	P62			
					0x5	P52			
					0x4	P42			
					0x3	P32			
					0x2	P22			
					0x1	P12			
					0x0	P02			
		D3–0	–	reserved	–		X	R	
Port Input Interrupt 3 Select Register (PINTSEL3)	**0x4443** (8 bits)	D7–4	**PINT3[3:0]**	Interrupt port select	PINT3[3:0]	Function	0xf	R/W	
					0xc	PC3			
					0xb	PB3			
					0xa	PA3			
					0x9	P93			
					0x8	P83			
					0x7	P73			
					0x6	P63			
					0x5	P53			
					0x4	P43			
					0x3	P33			
					0x2	P23			
					0x1	P13			
					0x0	P03			
		D3–0	–	reserved	–		X	R	
Port Input Interrupt 4 Select Register (PINTSEL4)	**0x4444** (8 bits)	D7–4	**PINT4[3:0]**	Interrupt port select	PINT4[3:0]	Function	0xf	R/W	
					0xc	PC4			
					0xb	PB4			
					0xa	PA4			
					0x9	P94			
					0x8	P84			
					0x7	P74			
					0x6	P64			
					0x5	P54			
					0x4	P44			
					0x3	P34			
					0x2	P24			
					0x1	P14			
					0x0	P04			
		D3–0	–	reserved	–		X	R	
Port Input Interrupt 5 Select Register (PINTSEL5)	**0x4445** (8 bits)	D7–4	**PINT5[3:0]**	Interrupt port select	PINT5[3:0]	Function	0xf	R/W	
					0xc	PC5			
					0xb	PB5			
					0xa	PA5			
					0x9	P95			
					0x8	P85			
					0x7	P75			
					0x6	P65			
					0x5	P55			
					0x4	P45			
					0x3	P35			
					0x2	P25			
					0x1	P15			
					0x0	P05			
		D3–0	–	reserved	–		X	R	

0x4446–0x4447 GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Port Input Interrupt 6 Select Register (PINTSEL6)	**0x4446** (8 bits)	D7–4	**PINT6[3:0]**	Interrupt port select	PINT6[3:0]	Function	0xf	R/W	
					0xc	PC6			
					0xb	PB6			
					0xa	PA6			
					0x9	P96			
					0x8	P86			
					0x7	P76			
					0x6	P66			
					0x5	P56			
					0x4	–			
					0x3	P36			
					0x2	P26			
					0x1	P16			
					0x0	P06			
		D3–0	–	reserved	–		X	R	
Port Input Interrupt 7 Select Register (PINTSEL7)	**0x4447** (8 bits)	D7–4	**PINT7[3:0]**	Interrupt port select	PINT7[3:0]	Function	0xf	R/W	
					0xc	PC7			
					0xb	PB7			
					0xa	–			
					0x9	P97			
					0x8	–			
					0x7	P77			
					0x6	P67			
					0x5	P57			
					0x4	–			
					0x3	P37			
					0x2	P27			
					0x1	–			
					0x0	P07			
		D3–0	–	reserved	–		X	R	

0x4500–0x450b USB Function Controller

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
MainIntStat	**0x4500**	D7	**SIE_IntStat**	1	SIE interrupts	0	None	0	R	
(Main interrupt	(8 bits)	D6	**EPrIntStat**	1	EPr interrupts	0	None	0	R	
status)		D5	**–**	–				–	–	0 when being read.
		D4	**FIFO_IntStat**	1	FIFO interrupts	0	None	0	R	
		D3–2	**–**	–				–	–	0 when being read.
		D1	**EP0IntStat**	1	EP0 interrupts	0	None	0	R	
		D0	**RcvEP0SETUP**	1	Receive EP0 SETUP	0	None	0	R(W)	
SIE_IntStat	**0x4501**	D7	**VBUS_Changed**	1	VBUS is changed	0	None	0	R(W)	
(SIE interrupt	(8 bits)	D6	**NonJ**	1	Detect non J state	0	None	0	R(W)	
status)		D5	**DetectReset**	1	Detect USB reset	0	None	0	R(W)	
		D4	**DetectSuspend**	1	Detect USB suspend	0	None	0	R(W)	
		D3	**RcvSOF**	1	Receive SOF token	0	None	0	R(W)	
		D2	**DetectJ**	1	Detect J state	0	None	0	R(W)	
		D1	**–**	–				–	–	0 when being read.
		D0	**SetAddressCmp**	1	AutoSetAddress complete	0	None	0	R(W)	
EPrIntStat	**0x4502**	D7–4	**–**	–				–	–	0 when being read.
(EPr interrupt	(8 bits)	D3	**EPdIntStat**	1	EPd interrupt	0	None	0	R	
status)		D2	**EPcIntStat**	1	EPc interrupt	0	None	0	R	
		D1	**EPbIntStat**	1	EPb interrupt	0	None	0	R	
		D0	**EPaIntStat**	1	EPa interrupt	0	None	0	R	
FIFO_IntStat	**0x4504**	D7	**DescriptorCmp**	1	Descriptor complete	0	None	0	R(W)	
(FIFO interrupt	(8 bits)	D6–0	**–**	–				–	–	0 when being read.
status)										
EP0IntStat	**0x4507**	D7–6	**–**	–				–	–	0 when being read.
(EP0 interrupt	(8 bits)	D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
status)		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	
EPaIntStat	**0x4508**	D7	**–**	–				–	–	0 when being read.
(EPa interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	
EPbIntStat	**0x4509**	D7	**–**	–				–	–	0 when being read.
(EPb interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	
EPcIntStat	**0x450a**	D7	**–**	–				–	–	0 when being read.
(EPc interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	
EPdIntStat	**0x450b**	D7	**–**	–				–	–	0 when being read.
(EPd interrupt	(8 bits)	D6	**OUT_ShortACK**	1	Out short packet ACK	0	None	0	R(W)	
status)		D5	**IN_TranACK**	1	In transaction ACK	0	None	0	R(W)	
		D4	**OUT_TranACK**	1	Out transaction ACK	0	None	0	R(W)	
		D3	**IN_TranNAK**	1	In transaction NAK	0	None	0	R(W)	
		D2	**OUT_TranNAK**	1	Out transaction NAK	0	None	0	R(W)	
		D1	**IN_TranErr**	1	In transaction error	0	None	0	R(W)	
		D0	**OUT_TranErr**	1	Out transaction error	0	None	0	R(W)	

0x4510–0x451b USB Function Controller

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
MainIntEnb (Main interrupt enable)	0x4510 (8 bits)	D7	EnSIE_IntStat	1	Enabled	0	Disabled	0	R/W	
		D6	EnEPrIntStat					0	R/W	
		D5	–	–				–	–	0 when being read.
		D4	EnFIFO_IntStat	1	Enabled	0	Disabled	0	R/W	
		D3–2	–	–				–	–	0 when being read.
		D1	EnEP0IntStat	1	Enabled	0	Disabled	0	R/W	
		D0	EnRcvEP0SETUP					0	R/W	
SIE_IntEnb (SIE interrupt enable)	0x4511 (8 bits)	D7	EnVBUS_Changed	1	Enabled	0	Disabled	0	R/W	
		D6	EnNonJ					0	R/W	
		D5	EnDetectReset					0	R/W	
		D4	EnDetectSuspend					0	R/W	
		D3	EnRcvSOF					0	R/W	
		D2	EnDetectJ					0	R/W	
		D1	–	–				–	–	0 when being read.
		D0	EnSetAddressCmp	1	Enabled	0	Disabled	0	R/W	
EPrIntEnb (EPr interrupt enable)	0x4512 (8 bits)	D7–4	–	–				–	–	0 when being read.
		D3	EnEPdIntStat	1	Enabled	0	Disabled	0	R/W	
		D2	EnEPcIntStat					0	R/W	
		D1	EnEPbIntStat					0	R/W	
		D0	EnEPaIntStat					0	R/W	
FIFO_IntEnb (FIFO interrupt enable)	0x4514 (8 bits)	D7	EnDescriptorCmp	1	Enabled	0	Disabled	0	R/W	
		D6–0	–	–				–	–	0 when being read.
EP0IntEnb (EP0 interrupt enable)	0x4517 (8 bits)	D7–6	–	–				–	–	0 when being read.
		D5	EnIN_TranACK	1	Enabled	0	Disabled	0	R/W	
		D4	EnOUT_TranACK					0	R/W	
		D3	EnIN_TranNAK					0	R/W	
		D2	EnOUT_TranNAK					0	R/W	
		D1	EnIN_TranErr					0	R/W	
		D0	EnOUT_TranErr					0	R/W	
EPaIntEnb (EPa interrupt enable)	0x4518 (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enabled	0	Disabled	0	R/W	
		D5	EnIN_TranACK					0	R/W	
		D4	EnOUT_TranACK					0	R/W	
		D3	EnIN_TranNAK					0	R/W	
		D2	EnOUT_TranNAK					0	R/W	
		D1	EnIN_TranErr					0	R/W	
		D0	EnOUT_TranErr					0	R/W	
EPbIntEnb (EPb interrupt enable)	0x4519 (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enabled	0	Disabled	0	R/W	
		D5	EnIN_TranACK					0	R/W	
		D4	EnOUT_TranACK					0	R/W	
		D3	EnIN_TranNAK					0	R/W	
		D2	EnOUT_TranNAK					0	R/W	
		D1	EnIN_TranErr					0	R/W	
		D0	EnOUT_TranErr					0	R/W	
EPcIntEnb (EPc interrupt enable)	0x451a (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enabled	0	Disabled	0	R/W	
		D5	EnIN_TranACK					0	R/W	
		D4	EnOUT_TranACK					0	R/W	
		D3	EnIN_TranNAK					0	R/W	
		D2	EnOUT_TranNAK					0	R/W	
		D1	EnIN_TranErr					0	R/W	
		D0	EnOUT_TranErr					0	R/W	
EPdIntEnb (EPd interrupt enable)	0x451b (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enabled	0	Disabled	0	R/W	
		D5	EnIN_TranACK					0	R/W	
		D4	EnOUT_TranACK					0	R/W	
		D3	EnIN_TranNAK					0	R/W	
		D2	EnOUT_TranNAK					0	R/W	
		D1	EnIN_TranErr					0	R/W	
		D0	EnOUT_TranErr					0	R/W	

0x4520–0x4537 USB Function Controller

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
RevisionNum (Revision number)	**0x4520** (8 bits)	D7	**RevisionNum[7]**	Revision number (0x12)	0	R	
		D6	**RevisionNum[6]**		0		
		D5	**RevisionNum[5]**		0		
		D4	**RevisionNum[4]**		1		
		D3	**RevisionNum[3]**		0		
		D2	**RevisionNum[2]**		0		
		D1	**RevisionNum[1]**		1		
		D0	**RevisionNum[0]**		0		
USB_Control (USB control register)	**0x4521** (8 bits)	D7	**DisBusDetect**	1 Disable bus detect / 0 Enable bus detect	0	R/W	
		D6	**EnAutoNego**	1 Enable auto negotiation / 0 Disable auto negotiation	0	R/W	
		D5	**InSUSPEND**	1 Monitor NonJ / 0 Do nothing	0	R/W	
		D4	**StartDetectJ**	1 Start J-state detection / 0 Do nothing	0	R/W	
		D3	**SendWakeup**	1 Send remote wakeup signal / 0 Do nothing	0	R/W	
		D2–1	**–**	–	–	–	0 when being read.
		D0	**ActiveUSB**	1 Activate USB / 0 Disactivate USB	0	R/W	
USB_Status (USB status register)	**0x4522** (8 bits)	D7	**VBUS**	1 VBUS=High / 0 VBUS=Low	X	R	
		D6	**FS**	1 FS mode (fixed) / 0 –	1	R	
		D5–2	**–**	–	–	–	0 when being read.
		D1	**LineState[1]**	LineState[1:0]: 1 1 = SE1; 1 0 = K; 0 1 = J; 0 0 = SE0	X	R	
		D0	**LineState[0]**		X		
XcvrControl (Xcvr control register)	**0x4523** (8 bits)	D7	**RpuEnb**	1 Enable pull-up / 0 Disable pull-up	1	R/W	
		D6–2	**–**	–	–	–	0 when being read.
		D1	**OpMode[1]**	OpMode[1:0]: 1 1 = reserved; 1 0 = Disable bit stuffing and NRZI encoding; 0 1 = Non-driving; 0 0 = Normal operation	0	R/W	
		D0	**OpMode[0]**		1		
USB_Test (USB test)	**0x4524** (8 bits)	D7	**EnUSB_Test**	1 Enable USB test / 0 Do nothing	0	R/W	
		D6–4	**–**	–	–	–	0 when being read.
		D3	**Test_SE0_NAK**	1 Test_SE0_NAK / 0 Do nothing	0	R/W	
		D2	**Test_J**	1 Test_J / 0 Do nothing	0	R/W	
		D1	**Test_K**	1 Test_K / 0 Do nothing	0	R/W	
		D0	**Test_Packet**	1 Test_Packet / 0 Do nothing	0	R/W	
EPnControl (Endpoint control)	**0x4525** (8 bits)	D7	**AllForceNAK**	1 Set all ForceNAK / 0 Do nothing	0	W	0 when being read.
		D6	**EPrForceSTALL**	1 Set EP's ForceSTALL / 0 Do nothing	0	W	
		D5	**AllFIFO_Clr**	1 Clear all FIFO / 0 Do nothing	0	W	
		D4–1	**–**	–	–	–	
		D0	**EP0FIFO_Clr**	1 Clear EP0 FIFO / 0 Do nothing	0	W	
EPrFIFO_Clr (EPr FIFO clear)	**0x4526** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**EPdFIFO_Clr**	1 Clear EPd FIFO / 0 Do nothing	0	W	
		D2	**EPcFIFO_Clr**	1 Clear EPc FIFO / 0 Do nothing	0	W	
		D1	**EPbFIFO_Clr**	1 Clear EPb FIFO / 0 Do nothing	0	W	
		D0	**EPaFIFO_Clr**	1 Clear EPa FIFO / 0 Do nothing	0	W	
FrameNumber_H (Frame number high)	**0x452e** (8 bits)	D7	**FnInvalid**	1 Invalid frame number / 0 Valid frame number	1	R	
		D6–3	**–**	–	–	–	0 when being read.
		D2	**FrameNumber[10]**	Frame number high	0	R	
		D1	**FrameNumber[9]**		0		
		D0	**FrameNumber[8]**		0		
FrameNumber_L (Frame number low)	**0x452f** (8 bits)	D7	**FrameNumber[7]**	Frame number low	0	R	
		D6	**FrameNumber[6]**		0		
		D5	**FrameNumber[5]**		0		
		D4	**FrameNumber[4]**		0		
		D3	**FrameNumber[3]**		0		
		D2	**FrameNumber[2]**		0		
		D1	**FrameNumber[1]**		0		
		D0	**FrameNumber[0]**		0		
EP0Setup_0 (EP0 set-up 0) | EP0Setup_7 (EP0 set-up 7)	**0x4530 | 0x4537** (8 bits)	D7	**EP0Setup_n[7]**	Endpoint 0 set-up data 0 | Endpoint 0 set-up data 7	0	R	
		D6	**EP0Setup_n[6]**		0		
		D5	**EP0Setup_n[5]**		0		
		D4	**EP0Setup_n[4]**		0		
		D3	**EP0Setup_n[3]**		0		
		D2	**EP0Setup_n[2]**		0		
		D1	**EP0Setup_n[1]**		0		
		D0	**EP0Setup_n[0]**		0		

0x4538–0x4543 USB Function Controller

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
USB_Address	**0x4538**	D7	**AutoSetAddress**	1 Auto set address 0 Do nothing	0	R/W	
(USB address)	(8 bits)	D6	**USB_Address[6]**	USB address	0	R/W	
		D5	**USB_Address[5]**		0		
		D4	**USB_Address[4]**		0		
		D3	**USB_Address[3]**		0		
		D2	**USB_Address[2]**		0		
		D1	**USB_Address[1]**		0		
		D0	**USB_Address[0]**		0		
EP0Control	**0x4539**	D7	**INxOUT**	1 IN 0 OUT	0	R/W	
(EP0 control)	(8 bits)	D6–1	**–**	–	–	–	0 when being read.
		D0	**ReplyDescriptor**	1 Reply descriptor 0 Do nothing	0	W	
EP0ControlIN	**0x453a**	D7	**–**	–	–	–	0 when being read.
(EP0 control	(8 bits)	D6	**EnShortPkt**	1 Enable short packet 0 Do nothing	0	R/W	
IN)		D5	**–**	–	–	–	0 when being read.
		D4	**ToggleStat**	Toggle sequence bit	0	R	
		D3	**ToggleSet**	1 Set toggle sequence bit 0 Do nothing	0	R/W	
		D2	**ToggleClr**	1 Clear toggle sequence bit 0 Do nothing	0	R/W	
		D1	**ForceNAK**	1 Force NAK 0 Do nothing	0	R/W	
		D0	**ForceSTALL**	1 Force STALL 0 Do nothing	0	R/W	
EP0ControlOUT	**0x453b**	D7	**AutoForceNAK**	1 Auto force NAK 0 Do nothing	0	R/W	
(EP0 control	(8 bits)	D6–5	**–**	–	–	–	0 when being read.
OUT)		D4	**ToggleStat**	Toggle sequence bit	0	R	
		D3	**ToggleSet**	1 Set toggle sequence bit 0 Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1 Clear toggle sequence bit 0 Do nothing	0	W	
		D1	**ForceNAK**	1 Force NAK 0 Do nothing	0	R/W	
		D0	**ForceSTALL**	1 Force STALL 0 Do nothing	0	R/W	
EP0MaxSize	**0x453f**	D7	**–**	–	–	–	0 when being read.
(EP0 max	(8 bits)	D6	**EP0MaxSize[6]**	Endpoint EP0 max packet size	0	R/W	
packet size)		D5	**EP0MaxSize[5]**		0		
		D4	**EP0MaxSize[4]**		0		
		D3	**EP0MaxSize[3]**		1		
		D2–0	**–**	–	–	–	0 when being read.
EPaControl	**0x4540**	D7	**AutoForceNAK**	1 Auto force NAK 0 Do nothing	0	R/W	
(EPa control)	(8 bits)	D6	**EnShortPkt**	1 Enable short packet 0 Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1 Disable auto force 0 Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit	0	R	
		D3	**ToggleSet**	1 Set toggle sequence bit 0 Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1 Clear toggle sequence bit 0 Do nothing	0	W	
		D1	**ForceNAK**	1 Force NAK 0 Do nothing	0	R/W	
		D0	**ForceSTALL**	1 Force STALL 0 Do nothing	0	R/W	
EPbControl	**0x4541**	D7	**AutoForceNAK**	1 Auto force NAK 0 Do nothing	0	R/W	
(EPb control)	(8 bits)	D6	**EnShortPkt**	1 Enable short packet 0 Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1 Disable auto force 0 Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit	0	R	
		D3	**ToggleSet**	1 Set toggle sequence bit 0 Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1 Clear toggle sequence bit 0 Do nothing	0	W	
		D1	**ForceNAK**	1 Force NAK 0 Do nothing	0	R/W	
		D0	**ForceSTALL**	1 Force STALL 0 Do nothing	0	R/W	
EPcControl	**0x4542**	D7	**AutoForceNAK**	1 Auto force NAK 0 Do nothing	0	R/W	
(EPc control)	(8 bits)	D6	**EnShortPkt**	1 Enable short packet 0 Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1 Disable auto force 0 Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit	0	R	
		D3	**ToggleSet**	1 Set toggle sequence bit 0 Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1 Clear toggle sequence bit 0 Do nothing	0	W	
		D1	**ForceNAK**	1 Force NAK 0 Do nothing	0	R/W	
		D0	**ForceSTALL**	1 Force STALL 0 Do nothing	0	R/W	
EPdControl	**0x4543**	D7	**AutoForceNAK**	1 Auto force NAK 0 Do nothing	0	R/W	
(EPd control)	(8 bits)	D6	**EnShortPkt**	1 Enable short packet 0 Do nothing	0	R/W	
		D5	**DisAF_NAK_Short**	1 Disable auto force 0 Auto force NAK short	0	R/W	
		D4	**ToggleStat**	Toggle sequence bit	0	R	
		D3	**ToggleSet**	1 Set toggle sequence bit 0 Do nothing	0	W	0 when being read.
		D2	**ToggleClr**	1 Clear toggle sequence bit 0 Do nothing	0	W	
		D1	**ForceNAK**	1 Force NAK 0 Do nothing	0	R/W	
		D0	**ForceSTALL**	1 Force STALL 0 Do nothing	0	R/W	

0x4550–0x455a USB Function Controller

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPaMaxSize_H (EPa max packet size high)	**0x4550** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1	**EPaMaxSize[9]**	Endpoint EPa max packet size	0	R/W	
		D0	**EPaMaxSize[8]**		0		
EPaMaxSize_L (EPa max packet size low)	**0x4551** (8 bits)	D7	**EPaMaxSize[7]**	Endpoint EPa max packet size	0	R/W	
		D6	**EPaMaxSize[6]**		0		
		D5	**EPaMaxSize[5]**		0		
		D4	**EPaMaxSize[4]**		0		
		D3	**EPaMaxSize[3]**		0		
		D2	**EPaMaxSize[2]**		0		
		D1	**EPaMaxSize[1]**		0		
		D0	**EPaMaxSize[0]**		0		
EPaConfig_0 (EPa configuration 0)	**0x4552** (8 bits)	D7	**INxOUT**	1 In / 0 Out	0	R/W	
		D6	**ToggleMode**	1 Always toggle / 0 Normal toggle	0	R/W	
		D5	**EnEndPoint**	1 Enable endpoint / 0 Disable endpoint	0	R/W	
		D4	**–**	–	–	–	0 when being read.
		D3	**EndPointNumber[3]**	Endpoint number (0x1 to 0xf)	0	R/W	
		D2	**EndPointNumber[2]**		0		
		D1	**EndPointNumber[1]**		0		
		D0	**EndPointNumber[0]**		0		
EPaConfig_1 (EPa configuration 1)	**0x4553** (8 bits)	D7	**ISO**	1 ISO / 0 Non-ISO	0	R/W	
		D6–0	**–**	–	–	–	0 when being read.
EPbMaxSize_H (EPb max packet size high)	**0x4554** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1	**EPbMaxSize[9]**	Endpoint EPb max packet size	0	R/W	
		D0	**EPbMaxSize[8]**		0		
EPbMaxSize_L (EPb max packet size low)	**0x4555** (8 bits)	D7	**EPbMaxSize[7]**	Endpoint EPb max packet size	0	R/W	
		D6	**EPbMaxSize[6]**		0		
		D5	**EPbMaxSize[5]**		0		
		D4	**EPbMaxSize[4]**		0		
		D3	**EPbMaxSize[3]**		0		
		D2	**EPbMaxSize[2]**		0		
		D1	**EPbMaxSize[1]**		0		
		D0	**EPbMaxSize[0]**		0		
EPbConfig_0 (EPb configuration 0)	**0x4556** (8 bits)	D7	**INxOUT**	1 In / 0 Out	0	R/W	
		D6	**ToggleMode**	1 Always toggle / 0 Normal toggle	0	R/W	
		D5	**EnEndPoint**	1 Enable endpoint / 0 Disable endpoint	0	R/W	
		D4	**–**	–	–	–	0 when being read.
		D3	**EndPointNumber[3]**	Endpoint number (0x1 to 0xf)	0	R/W	
		D2	**EndPointNumber[2]**		0		
		D1	**EndPointNumber[1]**		0		
		D0	**EndPointNumber[0]**		0		
EPbConfig_1 (EPb configuration 1)	**0x4557** (8 bits)	D7	**ISO**	1 ISO / 0 Non-ISO	0	R/W	
		D6–0	**–**	–	–	–	0 when being read.
EPcMaxSize_H (EPc max packet size high)	**0x4558** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1	**EPcMaxSize[9]**	Endpoint EPc max packet size	0	R/W	
		D0	**EPcMaxSize[8]**		0		
EPcMaxSize_L (EPc max packet size low)	**0x4559** (8 bits)	D7	**EPcMaxSize[7]**	Endpoint EPc max packet size	0	R/W	
		D6	**EPcMaxSize[6]**		0		
		D5	**EPcMaxSize[5]**		0		
		D4	**EPcMaxSize[4]**		0		
		D3	**EPcMaxSize[3]**		0		
		D2	**EPcMaxSize[2]**		0		
		D1	**EPcMaxSize[1]**		0		
		D0	**EPcMaxSize[0]**		0		
EPcConfig_0 (EPc configuration 0)	**0x455a** (8 bits)	D7	**INxOUT**	1 In / 0 Out	0	R/W	
		D6	**ToggleMode**	1 Always toggle / 0 Normal toggle	0	R/W	
		D5	**EnEndPoint**	1 Enable endpoint / 0 Disable endpoint	0	R/W	
		D4	**–**	–	–	–	0 when being read.
		D3	**EndPointNumber[3]**	Endpoint number (0x1 to 0xf)	0	R/W	
		D2	**EndPointNumber[2]**		0		
		D1	**EndPointNumber[1]**		0		
		D0	**EndPointNumber[0]**		0		

0x455b–0x4575 USB Function Controller

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcConfig_1	**0x455b**	D7	**ISO**	1 ISO 0 Non-ISO	0	R/W	
(EPc	(8 bits)	D6–0	**–**	–	–	–	0 when being read.
configuration 1)							
EPdMaxSize_H	**0x455c**	D7–2	**–**	–	–	–	0 when being read.
(EPd max	(8 bits)	D1	**EPdMaxSize[9]**	Endpoint EPd max packet size	0	R/W	
packet size		D0	**EPdMaxSize[8]**		0		
high)							
EPdMaxSize_L	**0x455d**	D7	**EPdMaxSize[7]**	Endpoint EPd max packet size	0	R/W	
(EPd max	(8 bits)	D6	**EPdMaxSize[6]**		0		
packet size		D5	**EPdMaxSize[5]**		0		
low)		D4	**EPdMaxSize[4]**		0		
		D3	**EPdMaxSize[3]**		0		
		D2	**EPdMaxSize[2]**		0		
		D1	**EPdMaxSize[1]**		0		
		D0	**EPdMaxSize[0]**		0		
EPdConfig_0	**0x455e**	D7	**INxOUT**	1 In 0 Out	0	R/W	
(EPd	(8 bits)	D6	**ToggleMode**	1 Always toggle 0 Normal toggle	0	R/W	
configuration 0)		D5	**EnEndPoint**	1 Enable endpoint 0 Disable endpoint	0	R/W	
		D4	**–**	–	–	–	0 when being read.
		D3	**EndPointNumber[3]**	Endpoint number	0	R/W	
		D2	**EndPointNumber[2]**	(0x1 to 0xf)	0		
		D1	**EndPointNumber[1]**		0		
		D0	**EndPointNumber[0]**		0		
EPdConfig_1	**0x455f**	D7	**ISO**	1 ISO 0 Non-ISO	0	R/W	
(EPd	(8 bits)	D6–0	**–**	–	–	–	0 when being read.
configuration 1)							
EPaStartAdrs_H	**0x4570**	D7–4	**–**	–	–	–	0 when being read.
(EPa FIFO start	(8 bits)	D3	**EPaStartAdrs[11]**	Endpoint EPa start address	0	R/W	
address high)		D2	**EPaStartAdrs[10]**		0		
		D1	**EPaStartAdrs[9]**		0		
		D0	**EPaStartAdrs[8]**		0		
EPaStartAdrs_L	**0x4571**	D7	**EPaStartAdrs[7]**	Endpoint EPa start address	0	R/W	
(EPa FIFO start	(8 bits)	D6	**EPaStartAdrs[6]**		0		
address low)		D5	**EPaStartAdrs[5]**		0		
		D4	**EPaStartAdrs[4]**		0		
		D3	**EPaStartAdrs[3]**		0		
		D2	**EPaStartAdrs[2]**		0		
		D1–0	**–**	–	–	–	0 when being read.
EPbStartAdrs_H	**0x4572**	D7–4	**–**	–	–	–	0 when being read.
(EPb FIFO start	(8 bits)	D3	**EPbStartAdrs[11]**	Endpoint EPb start address	0	R/W	
address high)		D2	**EPbStartAdrs[10]**		0		
		D1	**EPbStartAdrs[9]**		0		
		D0	**EPbStartAdrs[8]**		0		
EPbStartAdrs_L	**0x4573**	D7	**EPbStartAdrs[7]**	Endpoint EPb start address	0	R/W	
(EPb FIFO start	(8 bits)	D6	**EPbStartAdrs[6]**		0		
address low)		D5	**EPbStartAdrs[5]**		0		
		D4	**EPbStartAdrs[4]**		0		
		D3	**EPbStartAdrs[3]**		0		
		D2	**EPbStartAdrs[2]**		0		
		D1–0	**–**	–	–	–	0 when being read.
EPcStartAdrs_H	**0x4574**	D7–4	**–**	–	–	–	0 when being read.
(EPc FIFO start	(8 bits)	D3	**EPcStartAdrs[11]**	Endpoint EPc start address	0	R/W	
address high)		D2	**EPcStartAdrs[10]**		0		
		D1	**EPcStartAdrs[9]**		0		
		D0	**EPcStartAdrs[8]**		0		
EPcStartAdrs_L	**0x4575**	D7	**EPcStartAdrs[7]**	Endpoint EPc start address	0	R/W	
(EPc FIFO start	(8 bits)	D6	**EPcStartAdrs[6]**		0		
address low)		D5	**EPcStartAdrs[5]**		0		
		D4	**EPcStartAdrs[4]**		0		
		D3	**EPcStartAdrs[3]**		0		
		D2	**EPcStartAdrs[2]**		0		
		D1–0	**–**	–	–	–	0 when being read.

0x4576–0x4587 USB Function Controller

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdStartAdrs_H (EPd FIFO start address high)	**0x4576** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**EPdStartAdrs[11]**	Endpoint EPd start address	0	R/W	
		D2	**EPdStartAdrs[10]**		0		
		D1	**EPdStartAdrs[9]**		0		
		D0	**EPdStartAdrs[8]**		0		
EPdStartAdrs_L (EPd FIFO start address low)	**0x4577** (8 bits)	D7	**EPdStartAdrs[7]**	Endpoint EPd start address	0	R/W	
		D6	**EPdStartAdrs[6]**		0		
		D5	**EPdStartAdrs[5]**		0		
		D4	**EPdStartAdrs[4]**		0		
		D3	**EPdStartAdrs[3]**		0		
		D2	**EPdStartAdrs[2]**		0		
		D1–0	**–**	–	–	–	0 when being read.
CPU_JoinRd (CPU join FIFO read)	**0x4580** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**JoinEPdRd**	1 Join EPd FIFO read / 0 Do nothing	0	R/W	
		D2	**JoinEPcRd**	1 Join EPc FIFO read / 0 Do nothing	0	R/W	
		D1	**JoinEPbRd**	1 Join EPb FIFO read / 0 Do nothing	0	R/W	
		D0	**JoinEPaRd**	1 Join EPa FIFO read / 0 Do nothing	0	R/W	
CPU_JoinWr (CPU join FIFO write)	**0x4581** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**JoinEPdWr**	1 Join EPd FIFO write / 0 Do nothing	0	R/W	
		D2	**JoinEPcWr**	1 Join EPc FIFO write / 0 Do nothing	0	R/W	
		D1	**JoinEPbWr**	1 Join EPb FIFO write / 0 Do nothing	0	R/W	
		D0	**JoinEPaWr**	1 Join EPa FIFO write / 0 Do nothing	0	R/W	
EnEPnFIFO_Access (Enable EPn FIFO access)	**0x4582** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1	**EnEPnFIFO_Wr**	1 Enable join EPn FIFO write / 0 Do nothing	0	R/W	
		D0	**EnEPnFIFO_Rd**	1 Enable join EPn FIFO read / 0 Do nothing	0	R/W	
EPnFIFOforCPU (EPn FIFO for CPU)	**0x4583** (8 bits)	D7	**EPnFIFOData[7]**	Endpoint EP0 FIFO access from CPU	X	R/W	
		D6	**EPnFIFOData[6]**		X		
		D5	**EPnFIFOData[5]**		X		
		D4	**EPnFIFOData[4]**		X		
		D3	**EPnFIFOData[3]**		X		
		D2	**EPnFIFOData[2]**		X		
		D1	**EPnFIFOData[1]**		X		
		D0	**EPnFIFOData[0]**		X		
EPnRdRemain_H (EPn FIFO read remain high)	**0x4584** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**EPnRdRemain[11]**	Endpoint n FIFO read remain high	0	R	
		D2	**EPnRdRemain[10]**		0		
		D1	**EPnRdRemain[9]**		0		
		D0	**EPnRdRemain[8]**		0		
EPnRdRemain_L (EPn FIFO read remain low)	**0x4585** (8 bits)	D7	**EPnRdRemain[7]**	Endpoint n FIFO read remain low	0	R	
		D6	**EPnRdRemain[6]**		0		
		D5	**EPnRdRemain[5]**		0		
		D4	**EPnRdRemain[4]**		0		
		D3	**EPnRdRemain[3]**		0		
		D2	**EPnRdRemain[2]**		0		
		D1	**EPnRdRemain[1]**		0		
		D0	**EPnRdRemain[0]**		0		
EPnWrRemain_H (EPn FIFO write remain high)	**0x4586** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**EPnWrRemain[11]**	Endpoint n FIFO write remain high	0	R	
		D2	**EPnWrRemain[10]**		0		
		D1	**EPnWrRemain[9]**		0		
		D0	**EPnWrRemain[8]**		0		
EPnWrRemain_L (EPn FIFO write remain low)	**0x4587** (8 bits)	D7	**EPnWrRemain[7]**	Endpoint n FIFO write remain low	0	R	
		D6	**EPnWrRemain[6]**		0		
		D5	**EPnWrRemain[5]**		0		
		D4	**EPnWrRemain[4]**		0		
		D3	**EPnWrRemain[3]**		0		
		D2	**EPnWrRemain[2]**		0		
		D1	**EPnWrRemain[1]**		0		
		D0	**EPnWrRemain[0]**		0		

0x4588–0x458f USB Function Controller

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescAdrs_H	**0x4588**	D7–4	**–**	–	–	–	0 when being read.
(Descriptor	(8 bits)	D3	**DescAdrs[11]**	Descriptor address	0	R/W	
address high)		D2	**DescAdrs[10]**		0		
		D1	**DescAdrs[9]**		0		
		D0	**DescAdrs[8]**		0		
DescAdrs_L	**0x4589**	D7	**DescAdrs[7]**	Descriptor address	0	R/W	
(Descriptor	(8 bits)	D6	**DescAdrs[6]**		0		
address low)		D5	**DescAdrs[5]**		0		
		D4	**DescAdrs[4]**		0		
		D3	**DescAdrs[3]**		0		
		D2	**DescAdrs[2]**		0		
		D1	**DescAdrs[1]**		0		
		D0	**DescAdrs[0]**		0		
DescSize_H	**0x458a**	D7–2	**–**	–	–	–	0 when being read.
(Descriptor	(8 bits)	D1	**DescSize[9]**	Descriptor size	0	R/W	
size high)		D0	**DescSize[8]**		0		
DescSize_L	**0x458b**	D7	**DescSize[7]**	Descriptor size	0	R/W	
(Descriptor	(8 bits)	D6	**DescSize[6]**		0		
size low)		D5	**DescSize[5]**		0		
		D4	**DescSize[4]**		0		
		D3	**DescSize[3]**		0		
		D2	**DescSize[2]**		0		
		D1	**DescSize[1]**		0		
		D0	**DescSize[0]**		0		
DescDoor	**0x458f**	D7	**DescMode[7]**	Descriptor door	0	R/W	
(Descriptor	(8 bits)	D6	**DescMode[6]**		0		
door)		D5	**DescMode[5]**		0		
		D4	**DescMode[4]**		0		
		D3	**DescMode[3]**		0		
		D2	**DescMode[2]**		0		
		D1	**DescMode[1]**		0		
		D0	**DescMode[0]**		0		

0x4600–0x462a Real-time Clock

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Interrupt Status Register (RTC_INTSTAT)	0x4600 (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	RTCIRQ	Interrupt status	1 Occurred 0 Not occurred	X	R/W	Reset by writing 1.
RTC Interrupt Mode Register (RTC_INTMODE)	0x4601 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3–2	RTCT[1:0]	RTC interrupt cycle setup	RTCT[1:0] Cycle 0x3 1 hour 0x2 1 minute 0x1 1 second 0x0 1/64 second	X	R/W	
		D1	RTCIMD	RTC interrupt mode select	1 Level sense 0 Edge trigger	X	R/W	
		D0	RTCIEN	RTC interrupt enable	1 Enable 0 Disable	X	R/W	
RTC Control 0 Register (RTC_CNTL0)	0x4602 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	RTC24H	24H/12H mode select	1 24H 0 12H	X	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2	RTCADJ	30-second adjustment	1 Adjust 0 –	X	R/W	
		D1	RTCSTP	Counter run/stop control	1 Stop 0 Run	X	R/W	
		D0	RTCRST	Software reset	1 Reset 0 –	X	R/W	
RTC Control 1 Register (RTC_CNTL1)	0x4603 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	RTCBSY	Counter busy flag	1 Busy 0 R/W possible	X	R	
		D0	RTCHLD	Counter hold control	1 Hold 0 Running	X	R/W	
RTC Wakeup Configuration Register (RTC_WAKEUP)	0x460f (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	WUP_CTL	WAKEUP control	1 Active 0 Inactive	X	R/W	
		D0	WUP_POL	WAKEUP polarity selection	1 Active low 0 Active high	X	R/W	
RTC Second Register (RTC_SEC)	0x4614 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–4	RTCSH[2:0]	RTC 10-second counter	0 to 5	X	R/W	
		D3–0	RTCSL[3:0]	RTC 1-second counter	0 to 9	X	R/W	
RTC Minute Register (RTC_MIN)	0x4615 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–4	RTCMIH[2:0]	RTC 10-minute counter	0 to 5	X	R/W	
		D3–0	RTCMIL[3:0]	RTC 1-minute counter	0 to 9	X	R/W	
RTC Hour Register (RTC_HOUR)	0x4616 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	RTCAP	AM/PM indicator	1 PM 0 AM	X	R/W	
		D5–4	RTCHH[1:0]	RTC 10-hour counter	0 to 2 or 0 to 1	X	R/W	
		D3–0	RTCHL[3:0]	RTC 1-hour counter	0–9	X	R/W	
RTC Day Register (RTC_DAY)	0x4617 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5–4	RTCDH[1:0]	RTC 10-day counter	0 to 3	X	R/W	
		D3–0	RTCDL[3:0]	RTC 1-day counter	0 to 9	X	R/W	
RTC Month Register (RTC_MONTH)	0x4628 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	RTCMOH	RTC 10-month counter	0 to 1	X	R/W	
		D3–0	RTCMOL[3:0]	RTC 1-month counter	0 to 9	X	R/W	
RTC Year Register (RTC_YEAR)	0x4629 (8 bits)	D7–4	RTCYH[3:0]	RTC 10-year counter	0 to 9	X	R/W	
		D3–0	RTCYL[3:0]	RTC 1-year counter	0 to 9	X	R/W	
RTC Days of Week Register (RTC_WEEK)	0x462a (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	RTCWK[2:0]	RTC days of week counter	RTCWK[2:0] Days of week 0x7 – 0x6 Saturday 0x5 Friday 0x4 Thursday 0x3 Wednesday 0x2 Tuesday 0x1 Monday 0x0 Sunday	X	R/W	

0x4800–0x4803 8-bit Programmable Timer CH.0

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.0 Input Clock Select Register (PT8_CLK0)	**0x4800** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**PT8_EN**	PT8 Clock Enable	1 Enable 0 Disable	0	R/W	
		D3–0	**PT8_CLK [3:0]**	PT8 clock division ratio selection (Prescaler output clock)	PT8_CLK[3:0] Clock 0xf–0xd reserved 0xc PT8_CLK•1/4096 0xb PT8_CLK•1/2048 0xa PT8_CLK•1/1024 0x9 PT8_CLK•1/512 0x8 PT8_CLK•1/256 0x7 PT8_CLK•1/128 0x6 PT8_CLK•1/64 0x5 PT8_CLK•1/32 0x4 PT8_CLK•1/16 0x3 PT8_CLK•1/8 0x2 PT8_CLK•1/4 0x1 PT8_CLK•1/2 0x0 PT8_CLK•1/1	0x0	R/W	
PT8 CH.0 Reload Data Register (PT8_RLD0)	**0x4801** (8 bits)	D7–0	**PT8_RLD [7:0]**	8-bit programmable timer reload data PT8_RLD7 = MSB PT8_RLD0 = LSB	0 to 255	X	R/W	
PT8 CH.0 Counter Data Register (PT8_PTD0)	**0x4802** (8 bits)	D7–0	**PT8_PTD [7:0]**	8-bit programmable timer counter data PT8_PTD7 = MSB PT8_PTD0 = LSB	0 to 255	X	R	
PT8 CH.0 Control Register (PT8_CTL0)	**0x4803** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PT8_PSET**	Timer reset	1 Reset 0 Ignored	0	W	
		D0	**PT8_RUN**	Timer run/stop control	1 Run 0 Stop	0	R/W	

0x4804–0x4807 8-bit Programmable Timer CH.1

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.1 Input Clock Select Register (PT8_CLK1)	**0x4804** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	**PT8_EN**	PT8 Clock Enable	1 Enable / 0 Disable	0	R/W	
		D3–0	**PT8_CLK [3:0]**	PT8 clock division ratio selection (Prescaler output clock)	PT8_CLK[3:0] / Clock: 0xf–0xd reserved 0xc PT8_CLK•1/4096 0xb PT8_CLK•1/2048 0xa PT8_CLK•1/1024 0x9 PT8_CLK•1/512 0x8 PT8_CLK•1/256 0x7 PT8_CLK•1/128 0x6 PT8_CLK•1/64 0x5 PT8_CLK•1/32 0x4 PT8_CLK•1/16 0x3 PT8_CLK•1/8 0x2 PT8_CLK•1/4 0x1 PT8_CLK•1/2 0x0 PT8_CLK•1/1	0x0	R/W	
PT8 CH.1 Reload Data Register (PT8_RLD1)	**0x4805** (8 bits)	D7–0	**PT8_RLD [7:0]**	8-bit programmable timer reload data PT8_RLD7 = MSB PT8_RLD0 = LSB	0 to 255	X	R/W	
PT8 CH.1 Counter Data Register (PT8_PTD1)	**0x4806** (8 bits)	D7–0	**PT8_PTD [7:0]**	8-bit programmable timer counter data PT8_PTD7 = MSB PT8_PTD0 = LSB	0 to 255	X	R	
PT8 CH.1 Control Register (PT8_CTL1)	**0x4807** (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	**PT8_PSET**	Timer reset	1 Reset / 0 Ignored	0	W	
		D0	**PT8_RUN**	Timer run/stop control	1 Run / 0 Stop	0	R/W	

0x4808–0x480b 8-bit Programmable Timer CH.2

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.2 Input Clock Select Register (PT8_CLK2)	**0x4808** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**PT8_EN**	PT8 Clock Enable	1 Enable / 0 Disable	0	R/W	
		D3–0	**PT8_CLK [3:0]**	PT8 clock division ratio selection (Prescaler output clock)	PT8_CLK[3:0] / Clock 0xf–0xd / reserved 0xc / PT8_CLK•1/4096 0xb / PT8_CLK•1/2048 0xa / PT8_CLK•1/1024 0x9 / PT8_CLK•1/512 0x8 / PT8_CLK•1/256 0x7 / PT8_CLK•1/128 0x6 / PT8_CLK•1/64 0x5 / PT8_CLK•1/32 0x4 / PT8_CLK•1/16 0x3 / PT8_CLK•1/8 0x2 / PT8_CLK•1/4 0x1 / PT8_CLK•1/2 0x0 / PT8_CLK•1/1	0x0	R/W	
PT8 CH.2 Reload Data Register (PT8_RLD2)	**0x4809** (8 bits)	D7–0	**PT8_RLD [7:0]**	8-bit programmable timer reload data PT8_RLD7 = MSB PT8_RLD0 = LSB	0 to 255	X	R/W	
PT8 CH.2 Counter Data Register (PT8_PTD2)	**0x480a** (8 bits)	D7–0	**PT8_PTD [7:0]**	8-bit programmable timer counter data PT8_PTD7 = MSB PT8_PTD0 = LSB	0 to 255	X	R	
PT8 CH.2 Control Register (PT8_CTL2)	**0x480b** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PT8_PSET**	Timer reset	1 Reset / 0 Ignored	0	W	
		D0	**PT8_RUN**	Timer run/stop control	1 Run / 0 Stop	0	R/W	

0x480c–0x480f 8-bit Programmable Timer CH.3

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PT8 CH.3 Input Clock Select Register (PT8_CLK3)	**0x480c** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	**PT8_EN**	PT8 Clock Enable	1 Enable / 0 Disable	0	R/W	
		D3–0	**PT8_CLK [3:0]**	PT8 clock division ratio selection (Prescaler output clock)	PT8_CLK[3:0] / Clock 0xf–0xd / reserved 0xc / PT8_CLK•1/4096 0xb / PT8_CLK•1/2048 0xa / PT8_CLK•1/1024 0x9 / PT8_CLK•1/512 0x8 / PT8_CLK•1/256 0x7 / PT8_CLK•1/128 0x6 / PT8_CLK•1/64 0x5 / PT8_CLK•1/32 0x4 / PT8_CLK•1/16 0x3 / PT8_CLK•1/8 0x2 / PT8_CLK•1/4 0x1 / PT8_CLK•1/2 0x0 / PT8_CLK•1/1	0x0	R/W	
PT8 CH.3 Reload Data Register (PT8_RLD3)	**0x480d** (8 bits)	D7–0	**PT8_RLD [7:0]**	8-bit programmable timer reload data PT8_RLD7 = MSB PT8_RLD0 = LSB	0 to 255	X	R/W	
PT8 CH.3 Counter Data Register (PT8_PTD3)	**0x480e** (8 bits)	D7–0	**PT8_PTD [7:0]**	8-bit programmable timer counter data PT8_PTD7 = MSB PT8_PTD0 = LSB	0 to 255	X	R	
PT8 CH.3 Control Register (PT8_CTL3)	**0x480f** (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	**PT8_PSET**	Timer reset	1 Reset / 0 Ignored	0	W	
		D0	**PT8_RUN**	Timer run/stop control	1 Run / 0 Stop	0	R/W	

0x4900–0x4906 Clock Management Unit

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
System Clock Control Register (CMU_SYSCLKCTL)	0x4900 (8 bits)	D7	OSC3OFF	OSC3 disable during SLEEP	1 Stop 0 Run	0	R/W	
		D6	TMHSP	Wait-timer high-speed mode	1 High speed 0 Normal	0	R/W	
		D5	–	reserved	–	–	–	0 when being read.
		D4	WAKEUPWT	Wakeup-wait function enable	1 Wait interrupt 0 No wait	0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2	OSCSEL	OSC clock selection	1 OSC1 0 OSC3	0	R/W	
		D1	SOSC3	OSC3 oscillator on/off	1 On 0 Off	1	R/W	
		D0	–	reserved	–	–	–	0 when being read.
OSC3 Wait Timer Register (CMU_OSC3_WCNT)	0x4901 (8 bits)	D7–0	OSCTM[7:0]	OSC oscillation stabilization-wait timer	0–255	0x0	R/W	
Noise Filter Control Register (CMU_NF)	0x4902 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	DSINNF	DSIO input noise filter enable	1 Enable 0 Disable	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	INPORTNF	Input port noise filter enable	1 Disable 0 Enable	1	R/W	
		D0	OSC3NF	OSC3 input noise filter enable	1 Disable 0 Enable	1	R/W	
OSC3 Clock Divider Register (CMU_OSC3DIV)	0x4903 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	OSC3DIV [2:0]	OSC3 clock divider selection	OSC3DIV[2:0] / Divider: 0x7 OSC3•1/1; 0x6 OSC3•1/1; 0x5 OSC3•1/32; 0x4 OSC3•1/16; 0x3 OSC3•1/8; 0x2 OSC3•1/4; 0x1 OSC3•1/2; 0x0 OSC3•1/1	0x0	R/W	
	0x4904 (8 bits)	D7–0	–	reserved	–	–	–	0 when being read.
CMU_CLK Select Register (CMU_CMUCLK)	0x4905 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3–0	CMU_CLKSEL[3:0]	CMU_CLK selection	CMU_CLKSEL[3:0] / Clock source: 0xf–0xa reserved; 0x9 OSC3•1/32; 0x8 OSC3•1/16; 0x7 OSC3•1/8; 0x6 OSC3•1/4; 0x5 OSC3•1/2; 0x4 OSC3•1/1; 0x3 reserved; 0x2 BCLK; 0x1 OSC1; 0x0 OSC3	0x0	R/W	
Gated Clock Control 0 Register (CMU_GATEDCLK0)	0x4906 (8 bits)	D7	FLASHC_CLK_EN	FLASHC clock control (in HALT mode)	1 On 0 Off	1	R/W	
		D6	–	reserved	–	–	–	0 when being read.
		D5	USB_SAPB_CLK_EN	USB SAPB I/F clock control	1 On 0 Off	0	R/W	
		D4	USB_CLK_EN	USB IP 48 MHz clock control		0	R/W	
		D3–1	–	reserved	–	–	–	0 when being read.
		D0	PCLK_EN	Core peripheral clock control	1 On 0 Off	1	R/W	

0x4907–0x4920 Clock Management Unit

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Gated Clock Control 1 Register (CMU_GATEDCLK1)	**0x4907** (8 bits)	D7	**SRAMC_CLK_EN**	SRAMC clock control (in HALT mode)	1 On 0 Off	1	R/W	
		D6–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PT8_CLK_EN**	8-bit programmable timer clock control	1 On 0 Off	1	R/W	
		D0	**MFT_CLK_EN**	Multi-function timer clock control		1	R/W	
Gated Clock Control 2 Register (CMU_GATEDCLK2)	**0x4908** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**SPI_CLK_EN**	SPI CH.1 module clock control	1 On 0 Off	1	R/W	
		D4	**REMC_CLK_EN**	REMC module clock control		1	R/W	
		D3	**ADC_CLK_EN**	ADC module clock control		1	R/W	
		D2	**WDT_CLK_EN**	WDT module clock control		1	R/W	
		D1	**PORT_CLK_EN**	I/O port module clock control		1	R/W	
		D0	**RTC_SAPB_CLK_EN**	RTC SAPB I/F clock control		1	R/W	
USB Wait Control Register (CMU_USBWT)	**0x4909** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**USBSNZ**	USB snooze control	1 Enable 0 Disable	0	R/W	
		D4–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**USBWT[2:0]**	USB register access wait control	USBWT[2:0] / Wait cycle 0x7 / 7 cycles : / : 0x0 / 0 cycles	0x7	R/W	
CMU Write Protect Register (CMU_PROTECT)	**0x4920** (8 bits)	D7–0	**CLGP[7:0]**	CMU register protect flag	Writing 10010110 (0x96) removes the write protection of the CMU registers (0x4900–0x4909). Writing another value set the write protection.	0x0	R/W	

0x5000–0x5018 SRAM Controller

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
External Memory Static Wait Control Register (EXTMEM_SWAIT)	**0x5000** (16 bits)	D15–12	**CE3WAIT [3:0]**	#CE3 static wait cycle setup	CE*x*WAIT[3:0] / Wait cycle 0xf / 15 cycles 0xe / 14 cycles 0xd / 13 cycles : / : 0x2 / 2 cycles 0x1 / 1 cycle 0x0 / 0 cycles	0xf	R/W	
		D11–8	**CE2WAIT [3:0]**	#CE2 static wait cycle setup		0xf	R/W	
		D7–4	**CE1WAIT [3:0]**	#CE1 static wait cycle setup		0xf	R/W	
		D3–0	**CE0WAIT [3:0]**	#CE0 static wait cycle setup		0xf	R/W	
External Memory Device Size Setup Register (EXTMEM_SIZE)	**0x5004** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3	**CE3SIZE**	#CE3 device size selection	1 16 bits / 0 8 bits	1	R/W	
		D2	**CE2SIZE**	#CE2 device size selection	1 16 bits / 0 8 bits	1	R/W	
		D1	**CE1SIZE**	#CE1 device size selection	1 16 bits / 0 8 bits	1	R/W	
		D0	**CE0SIZE**	#CE0 device size selection	1 16 bits / 0 8 bits	1	R/W	
External Memory Device Type Setup Register (EXTMEM_A0_BSL)	**0x5008** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3	**CE3TYPE**	#CE3 device type selection	1 BSL / 0 A0	0	R/W	
		D2	**CE2TYPE**	#CE2 device type selection	1 BSL / 0 A0	0	R/W	
		D1	**CE1TYPE**	#CE1 device type selection	1 BSL / 0 A0	0	R/W	
		D0	**CE0TYPE**	#CE0 device type selection	1 BSL / 0 A0	0	R/W	
External VRAM Select Register (EXTMEM_VRAM_SEL)	**0x500c** (16 bits)	D15–0	**–**	reserved	–	–	–	0 when being read.
Internal Memory Static Wait Control Register (IRAM2_SWAIT)	**0x5010** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**IRAM2_WAIT**	IRAM2 static wait cycle setup	1 1 cycle / 0 0 cycles	1	R/W	
MAC Wait Control Register (MAC_WAIT)	**0x5014** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**MAC_WAIT**	MAC wait cycle setup	1 1 cycle / 0 0 cycles	1	R/W	
RTC Wait Control Register (RTC_WAIT)	**0x5018** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**RTC_WAIT [2:0]**	RTC access wait cycle setup	RTC_WAIT[2:0] / Wait cycle 0x7 / 7 cycles : / : 0x0 / 0 cycles	0x7	R/W	

0x5200–0x527e Multi-Function Timer

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MFT Counter Data Register (MFT_TC)	**0x5200** (16 bits)	D15–0	**TC[15:0]**	Counter data TC15 = MSB TC0 = LSB	0x0 to 0xffff	0x0	R/W	
MFT Period Data Register (MFT_PRD)	**0x5202** (16 bits)	D15–0	**PRD[15:0]**	Period data PRD15 = MSB PRD0 = LSB	0x0 to 0xffff	0x0	R/W	
MFT Compare Data Register (MFT_CMP)	**0x5204** (16 bits)	D15–0	**CR[15:0]**	Compare data CR15 = MSB CR0 = LSB	0x0 to 0xffff	0x0	R/W	
MFT Control Register (MFT_CTL)	**0x5206** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11	**TPSON**	Clock control	1 On 0 Off	0	R/W	
		D10–8	**TPS[2:0]**	Clock division ratio selection (Prescaler output clock)	TPS[2:0] Count clock 0x7 MFT_CLK•1/128 0x6 MFT_CLK•1/64 0x5 MFT_CLK•1/32 0x4 MFT_CLK•1/16 0x3 MFT_CLK•1/8 0x2 MFT_CLK•1/4 0x1 MFT_CLK•1/2 0x0 MFT_CLK•1/1	0x0	R/W	
		D7	**BUFEN**	Comparison/period buffer enable	1 Enable 0 Disable	0	R/W	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5	**TCKS**	Input clock selection	1 External 0 Internal	0	R/W	
		D4–2	**–**	reserved	–	–	–	0 when being read.
		D1	**PRST**	Timer reset	1 Reset 0 Ignored	0	W	
		D0	**TMEN**	Timer run/stop control	1 Run 0 Stop	0	R/W	
MFT Input/Output Control Register (MFT_IOCTL)	**0x521e** (16 bits)	D15–7	**–**	reserved	–	–	–	0 when being read.
		D6	**APIL**	ADC protection input selection	1 Lower limit 0 Upper limit	0	R/W	
		D5	**PPIL**	Port protection input level selection	1 Low level 0 High level	0	R/W	
		D4	**IQA**	Port protection input noise filter	1 12 clocks 0 6 clocks	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2	**PTM**	Clock output enable	1 Enable 0 Disable	0	R/W	
		D1	**INITOL**	Initial output level	1 High 0 Low	0	R/W	
		D0	**OINV**	Inverse output	1 Invert 0 Normal	0	R/W	
MFT Interrupt Enable Register (MFT_IE)	**0x5230** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3	**CMPIE**	Compare-match interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**PRDIE**	Period-match interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**APIE**	ADC protection interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**PPIE**	Port protection interrupt enable	1 Enable 0 Disable	0	R/W	
MFT Interrupt Flag Register (MFT_IF)	**0x5238** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3	**CMPIF**	Compare-match interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D2	**PRDIF**	Period-match interrupt flag		0	R/W	
		D1	**APIF**	ADC protection interrupt flag		0	R/W	
		D0	**PPIF**	Port protection interrupt flag		0	R/W	
MFT Test Register (MFT_TST)	**0x527e** (16 bits)	D15–7	**–**	reserved	–	–	–	0 when being read.
		D6	**DBGMD**	MFT operation in debug mode	1 Stop 0 Run	0	R/W	
		D5–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**–**	reserved	0x7	0x7	–	Fix at 0x7.

0x5300–0x5308 I²S

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S CH.0 Control Register (I2S_CTL_OUT)	**0x5300** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10	**DTSIGN**	I²S CH.0 signed/unsigned data format select	1 Signed / 0 Unsigned	0	R/W	
		D9	**DATRES0**	I²S CH.0 output data resolution select	1 24 bits / 0 16 bits	0	R/W	
		D8	**I2SEN0**	I²S CH.0 enable	1 Enable / 0 Disable	0	R/W	
		D7	**WCLKMD0**	I²S CH.0 output word clock mode select	1 L: high R: low / 0 L: low R: high	0	R/W	
		D6	**BCLKPOL0**	I²S CH.0 output bit clock polarity select	1 Negative / 0 Positive	0	R/W	
		D5	**DTFORM**	I²S CH.0 output data format select	1 LSB first / 0 MSB first	0	R/W	
		D4	**I2SOUTEN**	I²S CH.0 output enable	1 Enable / 0 Disable	0	R/W	
		D3–2	**DTTMG0 [1:0]**	I²S CH.0 output data timing select	0x3 reserved 0x2 Right justified 0x1 Left justified 0x0 I²S	0x0	R/W	
		D1–0	**CHMD[1:0]**	I²S CH.0 output channel mode select	0x3 Mute 0x2 Mono left 0x1 Mono right 0x0 Stereo	0x0	R/W	
I²S CH.1 Control Register (I2S_CTL_IN)	**0x5302** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**WCLKMD1**	I²S CH.1 input word clock mode select	1 L: high R: low / 0 L: low R: high	0	R/W	
		D4	**BCLKPOL1**	I²S CH.1 input bit clock polarity select	1 Negative / 0 Positive	0	R/W	
		D3–2	**DTTMG1 [1:0]**	I²S CH.1 input data timing select	0x3 reserved 0x2 Right justified 0x1 Left justified 0x0 I²S	0x0	R/W	
		D1	**I2SBYPASS**	I²S bypass mode select	1 Bypass / 0 Normal	0	R/W	
		D0	**I2SEN1**	I²S CH.1 enable	1 Enable / 0 Disable	0	R/W	
I²S MCLK Divide Ratio Register (I2S_DV_MCLK)	**0x5304** (16 bits)	D15	**MCLKSEL**	I2S_MCLK source clock select	1 I2S_MCLK0 input clock / 0 System clock	0	R/W	
		D14–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**MCLKDIV [5:0]**	I2S_MCLK divide ratio select	0x3f PCLK•1/64 0x3e PCLK•1/63 0x3d PCLK•1/62 : 0x2 PCLK•1/3 0x1 PCLK•1/2 0x0 PCLK•1/1	0x0	R/W	
I²S Audio Clock Divide Ratio Register (I2S_DV_AUDIO_CLK)	**0x5306** (16 bits)	D15–12	**WSCLKCYC1 [3:0]**	I²S CH.1 WS clock cycle setup	0xf 31 clocks 0xe 30 clocks 0xd 29 clocks 0xc 28 clocks 0xb 27 clocks 0xa 26 clocks 0x9 25 clocks 0x8 24 clocks	0x0	R/W	
		D11–8	**WSCLKCYC0 [3:0]**	I²S CH.0 WS clock cycle setup	0x7 23 clocks 0x6 22 clocks 0x5 21 clocks 0x4 20 clocks 0x3 19 clocks 0x2 18 clocks 0x1 17 clocks 0x0 16 clocks	0x0	R/W	
		D7–0	**BCLKDIV [7:0]**	I²S CH.0 bit clock divide ratio select	0xff SRC_CLK•1/512 0xfe SRC_CLK•1/510 0xfd SRC_CLK•1/508 : 0x2 SRC_CLK•1/6 0x1 SRC_CLK•1/4 0x0 SRC_CLK•1/2	0x0	R/W	SRC_CLK: PCLK or I2S_MCLK0 input clock
I²S Start/Stop Register (I2S_START)	**0x5308** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**I2SSTART1**	I²S CH.1 start/stop control	1 Start / 0 Stop	0	R/W	
		D7	**I2SBUSY0**	I²S CH.0 busy flag	1 Busy / 0 Idle	0	R	
		D6–1	**–**	reserved	–	–	–	0 when being read.
		D0	**I2SSTART0**	I²S CH.0 start/stop control	1 Start (run) / 0 Stop	0	R/W	

0x530a–0x5314 I²S

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S FIFO Status Register (I2S_FIFO_STAT)	**0x530a** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9	**I2SFIFOFF1**	I²S CH.1 FIFO full flag	1 Full / 0 Not full	0	R	
		D8	**I2SFIFOEF1**	I²S CH.1 FIFO empty flag	1 Empty / 0 Not empty	1	R	
		D7–5	**–**	reserved	–	–	–	0 when being read.
		D4–2	**FIFOSTAT0 [2:0]**	I²S CH.0 FIFO state machine	FIFOSTAT0[2:0] / State 0x7–0x6 / reserved 0x5 / FLUSH 0x4 / EMPTY 0x3 / LACK 0x2 / FULL 0x1 / INIT 0x0 / STOP	0x0	R	
		D1	**I2SFIFOFF0**	I²S CH.0 FIFO full flag	1 Full / 0 Not full	0	R	
		D0	**I2SFIFOEF0**	I²S CH.0 FIFO empty flag	1 Empty / 0 Not empty	1	R	
I²S Interrupt Mode Select Register (I2S_INT_MOD)	**0x530c** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**I2SINTMD1 [1:0]**	I²S CH.1 interrupt mode select	I2SINTMD1[1:0] / State 0x3 / reserved 0x2 / One data 0x1 / Whole full 0x0 / Half full	0x0	R/W	
		D3–2	**I2SINTMD0 [1:0]**	I²S CH.0 interrupt mode select	I2SINTMD0[1:0] / State 0x3 / reserved 0x2 / One empty 0x1 / Whole empty 0x0 / Half empty	0x0	R/W	
		D1	**I2SINTEN1**	I²S CH.1 interrupt enable	1 Enable / 0 Disable	0	R/W	
		D0	**I2SINTEN0**	I²S CH.0 interrupt enable	1 Enable / 0 Disable	0	R/W	
I²S CH.0 FIFO Register (I2S_FIFO_OUT)	**0x5310** (24 bits)	D23–0	**I2SFIFO0 [23:0]**	I²S CH.0 FIFO (output data)	0 to 0xffffff (24 bits)	0x0	W	for 24-bit data 0 when being read.
	0x5310 (16 bits)	D15–0	**I2SFIFO0 [15:0]**		0 to 0xffff (16 bits)			for 16-bit data 0 when being read.
I²S CH.1 FIFO Register (I2S_FIFO_IN)	**0x5314** (16 bits)	D15–0	**I2SFIFO1 [15:0]**	I²S CH.1 FIFO (input data)	0 to 0xffff	0x0	R	

0x5400–0x5410 Remote Controller

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Prescaler Control Register (REMC_PSC)	**0x5400** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2	**REMPSON**	REMC prescaler control	1 On 0 Off	0	R/W	
		D1–0	**REMPSDIV [1:0]**	REMC prescaler division ratio select (Prescaler output clock)	REMPSDIV[1:0] Clock 0x3 PCLK•1/32 0x2 PCLK•1/24 0x1 PCLK•1/16 0x0 PCLK•1/12	0x0	R/W	
REMC Configuration Register (REMC_CFG)	**0x5404** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7	**MODE**	REMC mode select	1 Receive 0 Transmit	0	R/W	
		D6	**RIF**	Rising edge interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D5	**FIF**	Falling edge interrupt flag		0	R/W	
		D4	**UIF**	Underflow interrupt flag		0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2	**RIE**	Rising edge interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**FIE**	Falling edge interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**UIE**	Underflow interrupt enable	1 Enable 0 Disable	0	R/W	
REMC Control Register (REMC_CTL)	**0x5408** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**REMST**	REMC start/stop control	1 Start 0 Stop	0	R/W	
REMC Carrier Load Register (REMC_CARL)	**0x540c** (16 bits)	D15–8	**CLDH[7:0]**	REMC carrier high width setup	0x0 to 0xff	0x0	R/W	
		D7–0	**CLDL[7:0]**	REMC carrier low width setup	0x0 to 0xff	0x0	R/W	
REMC Envelope Load Register (REMC_ENVL)	**0x540e** (16 bits)	D15–0	**ELD[15:0]**	Envelop counter preset data	0x0 to 0xffff	0x0	R/W	
REMC Envelope Capture Register (REMC_ENVC)	**0x5410** (16 bits)	D15–0	**ECP[15:0]**	Receive envelop pulse width	0x0 to 0xffff	0x0	R	

0x5520–0x5558 A/D Converter

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Clock Control Register (AD_CLKCTL)	**0x5520** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3	**PSONAD**	A/D converter clock control	1 On / 0 Off	0	R/W	
		D2–0	**PSAD[2:0]**	A/D converter clock division ratio selection	PSAD[2:0] = A/D clock: 0x7 PCLK•1/256; 0x6 PCLK•1/128; 0x5 PCLK•1/64; 0x4 PCLK•1/32; 0x3 PCLK•1/16; 0x2 PCLK•1/8; 0x1 PCLK•1/4; 0x0 PCLK•1/2	0x0	R/W	
A/D Conversion Result Register (AD_DAT)	**0x5540** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**ADD[9:0]**	A/D converted data ADD9 = MSB ADD0 = LSB	0x0 to 0x3ff	0x0	R	
A/D Trigger/ Channel Select Register (AD_TRIG_CH)	**0x5542** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13–11	**CE[2:0]**	A/D converter end channel selection	0 to 7	0x0	R/W	
		D10–8	**CS[2:0]**	A/D converter start channel selection	0 to 7	0x0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**MS**	A/D conversion mode selection	1 Continuous / 0 Normal	0	R/W	
		D4–3	**TS[1:0]**	A/D conversion trigger selection	TS[1:0] = Trigger: 0x3 #ADTRG pin; 0x2 PT8 CH.0; 0x1 MFT; 0x0 Software	0x0	R/W	
		D2–0	**CH[2:0]**	A/D conversion channel status	0 to 7	0x0	R	
A/D Control/ Status Register (AD_CTL)	**0x5544** (16 bits)	D15	**ADCMPE**	Upper/lower-limit comparison enable	1 Enable / 0 Disable	0	R/W	Can be used when ADCADV = 1.
		D14–12	**ADCMP[2:0]**	Upper/lower-limit comparison channel selection	0 to 7	0x0	R/W	
		D11	**ADUPRST**	Upper-limit comparison status	1 Out of range / 0 Within range	0	R	
		D10	**ADLWRST**	Lower-limit comparison status	1 Out of range / 0 Within range	0	R	
		D9–8	**ST[1:0]**	Input signal sampling time setup	ST[1:0] = Sampling time: 0x3 9 clocks; 0x2 7 clocks; 0x1 5 clocks; 0x0 3 clocks	0x3	R/W	Use with 9 clocks.
		D7	**–**	reserved	–	–	–	0 when being read.
		D6	**INTMODE**	Interrupt signal mode	1 Complete only / 0 OR	0	R/W	Can be used when ADCADV = 1.
		D5	**CMPINTEN**	Out-of-range int. enable	1 Enable / 0 Disable	0	R/W	
		D4	**CNVINTEN**	Conversion-complete int. enable	1 Enable / 0 Disable	1	R/W	Can be changed when ADCADV = 1.
		D3	**ADF**	Conversion-complete flag	1 Completed / 0 Run/Stand-by	0	R	Reset when ADD is read.
		D2	**ADE**	A/D enable	1 Enable / 0 Disable	0	R/W	
		D1	**ADST**	A/D conversion control/status	1 Start/Run / 0 Stop	0	R/W	
		D0	**OWE**	Overwrite error flag	1 Error / 0 Normal	0	R/W	Reset by writing 0.
A/D Channel Status Flag Register (AD_CH_STAT)	**0x5546** (16 bits)	D15	**OWE7**	CH.7 overwrite error flag	1 Error / 0 Normal	0	R/W	Can be used when ADCADV = 1. Reset by writing 0.
		D14	**OWE6**	CH.6 overwrite error flag		0	R/W	
		D13	**OWE5**	CH.5 overwrite error flag		0	R/W	
		D12	**OWE4**	CH.4 overwrite error flag		0	R/W	
		D11	**OWE3**	CH.3 overwrite error flag		0	R/W	
		D10	**OWE2**	CH.2 overwrite error flag		0	R/W	
		D9	**OWE1**	CH.1 overwrite error flag		0	R/W	
		D8	**OWE0**	CH.0 overwrite error flag		0	R/W	
		D7	**ADF7**	CH.7 conversion-complete flag	1 Completed / 0 Run/Stand-by	0	R	Can be used when ADCADV = 1. Reset when AD*x*BUF [9:0] is read.
		D6	**ADF6**	CH.6 conversion-complete flag		0	R	
		D5	**ADF5**	CH.5 conversion-complete flag		0	R	
		D4	**ADF4**	CH.4 conversion-complete flag		0	R	
		D3	**ADF3**	CH.3 conversion-complete flag		0	R	
		D2	**ADF2**	CH.2 conversion-complete flag		0	R	
		D1	**ADF1**	CH.1 conversion-complete flag		0	R	
		D0	**ADF0**	CH.0 conversion-complete flag		0	R	
A/D CH.*x* Conversion Result Buffer Register (AD_CH*x*_BUF)	**0x5548** | **0x5556** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**AD*x*BUF [9:0]**	A/D CH.*x* converted data AD*x*BUF9 = MSB AD*x*BUF0 = LSB	0x0 to 0x3ff	0x0	R	Can be used when ADCADV = 1.
A/D Upper Limit Value Register (AD_UPPER)	**0x5558** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**ADUPR[9:0]**	A/D conversion upper limit value ADUPR9 = MSB ADUPR0 = LSB	0x0 to 0x3ff	0x0	R/W	Can be used when ADCADV = 1.

0x555a–0x555e A/D Converter

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Lower Limit Value Register (AD_LOWER)	**0x555a** (16 bits)	D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**ADLWR [9:0]**	A/D conversion lower limit value ADLWR9 = MSB ADLWR0 = LSB	0x0 to 0x3ff	0x0	R/W	Can be used when ADCADV = 1.
A/D Conversion Complete Interrupt Mask Register (AD_INTMASK)	**0x555c** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7	**INTMASK7**	CH.7 conversion-complete int. mask	1 Interrupt enabled / 0 Interrupt mask	1	R/W	Can be used when ADCADV = 1.
		D6	**INTMASK6**	CH.6 conversion-complete int. mask		1	R/W	
		D5	**INTMASK5**	CH.5 conversion-complete int. mask		1	R/W	
		D4	**INTMASK4**	CH.4 conversion-complete int. mask		1	R/W	
		D3	**INTMASK3**	CH.3 conversion-complete int. mask		1	R/W	
		D2	**INTMASK2**	CH.2 conversion-complete int. mask		1	R/W	
		D1	**INTMASK1**	CH.1 conversion-complete int. mask		1	R/W	
		D0	**INTMASK0**	CH.0 conversion-complete int. mask		1	R/W	
A/D Converter Mode Select/ Internal Status Register (AD_ADVMODE)	**0x555e** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**ADCADV**	Standard/advanced mode selection	1 Advanced / 0 Standard	0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**ISTATE[1:0]**	Internal status	ISTATE[1:0] / Status: 0x3 Converting 0x2 reserved 0x1 Sampling 0x0 Idle	0x0	R	
		D3–0	**ICOUNTER [3:0]**	Internal counter value	0 to 15	0x0	R	

0x5660–0x566c Watchdog Timer

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Write Protect Register (WD_WP)	**0x5660** (16 bits)	D15–0	**WDPTC [15:0]**	WDT register write protect flag	Writing 0x96 removes the write protection of the WD_EN, WD_CMP_L, and WD_CMP_H registers (0x5662–0x5666). Writing another value set the write protection.	X	W	0 when being read.
WDT Enable and Setup Register (WD_EN)	**0x5662** (16 bits)	D15–7	**–**	reserved	–	–	–	0 when being read.
		D6	**CLKSEL**	WDT input clock select	1 External clk 0 Internal clk	0	R/W	
		D5	**CLKEN**	WDT clock output control	1 On 0 Off	0	R/W	
		D4	**RUNSTP**	WDT Run/Stop control	1 Run 0 Stop	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**NMIEN**	WDT NMI enable	1 Enable 0 Disable	0	R/W	
		D0	**RESEN**	WDT RESET enable	1 Enable 0 Disable	0	R/W	
WDT Comparison Data L Register (WD_CMP_L)	**0x5664** (16 bits)	D15–0	**CMPDT [15:0]**	WDT comparison data CMPDT0 = LSB	0x0 to 0x3fffffff (low-order 16 bits)	0x0	R/W	
WDT Comparison Data H Register (WD_CMP_H)	**0x5666** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13-0	**CMPDT [29:16]**	WDT comparison data CMPDT29 = MSB	0x0 to 0x3fffffff (high-order 14 bits)	0x0	R/W	
WDT Count Data L Register (WD_CNT_L)	**0x5668** (16 bits)	D15–0	**CTRDT [15:0]**	WDT counter data CTRDT0 = LSB	0x0 to 0x3fffffff (low-order 16 bits)	X	R	
WDT Count Data H Register (WD_CNT_H)	**0x566a** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13-0	**CTRDT [29:16]**	WDT counter data CTRDT29 = MSB	0x0 to 0x3fffffff (high-order 14 bits)	X	R	
WDT Control Register (WD_CTL)	**0x566c** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**WDRESEN**	WDT reset	1 Reset 0 ignored	0	W	

0x5700–0x5708 Extended SPI

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SPI CH.1 Status Register (SPI_ST1)	**0x5700** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2	**SPBSY**	Transfer busy flag (master) ss signal low flag (slave)	1 Busy 0 Idle 1 ss = L 0 ss = H	0	R	
		D1	**SPRBF**	Receive data buffer full flag	1 Full 0 Not full	0	R	
		D0	**SPTBE**	Transmit data buffer empty flag	1 Empty 0 Not empty	1	R	
SPI CH.1 Transmit Data Register (SPI_TXD1)	**0x5702** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**SPTDB[7:0]**	SPI CH.1 transmit data buffer SPTDB7 = MSB SPTDB0 = LSB	0x0 to 0xff	0x0	R/W	
SPI CH.1 Receive Data Register (SPI_RXD1)	**0x5704** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**SPRDB[7:0]**	SPI CH.1 receive data buffer SPRDB7 = MSB SPRDB0 = LSB	0x0 to 0xff	0x0	R	
SPI CH.1 Control Register (SPI_CTL1)	**0x5706** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**SPRIE**	Receive data buffer full int. enable	1 Enable 0 Disable	0	R/W	
		D4	**SPTIE**	Transmit data buffer empty int. enable	1 Enable 0 Disable	0	R/W	
		D3	**CPHA**	Clock phase select	1 Data out 0 Data in	0	R/W	These bits must be set before setting SPEN to 1.
		D2	**CPOL**	Clock polarity select	1 Active L 0 Active H	0	R/W	
		D1	**MSSL**	Master/slave mode select	1 Master 0 Slave	0	R/W	
		D0	**SPEN**	SPI CH.1 enable	1 Enable 0 Disable	0	R/W	
SPI CH.1 Clock Control Register (SPI_CLK1)	**0x5708** (16 bits)	D15–5	**–**	reserved	–	–	–	0 when being read.
		D4	**SPI_CKE**	SPI CH.1 clock enable	1 Enable 0 Disable	0	R/W	
		D3–0	**SPI_CLK [3:0]**	SPI CH.1 clock division ratio selection (Prescaler output clock)	SPI_CLK[3:0] / Clock: 0xf–0xd: reserved 0xc: PCLK•1/4096 0xb: PCLK•1/2048 0xa: PCLK•1/1024 0x9: PCLK•1/512 0x8: PCLK•1/256 0x7: PCLK•1/128 0x6: PCLK•1/64 0x5: PCLK•1/32 0x4: PCLK•1/16 0x3: PCLK•1/8 0x2: PCLK•1/4 0x1: PCLK•1/2 0x0: PCLK•1/1	0x0	R/W	

0x5800–0x5816 Flash Controller

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FLASHC Control Register (FLASH_CTL)	**0x5800** (16 bits)	D15	**FLS_STAT**	Flash status flag	1 Busy 0 Idle	1	R	
		D14–11	**–**	reserved	–	–	–	0 when being read.
		D10	**STOP**	Flash erase/program stop	1 Stop 0 Ignored	0	W	
		D9	**START_HOLD**	Hold period start	1 Start 0 Ignored	0	W	
		D8	**START_ERASE**	Flash erasing start	1 Start 0 Ignored	0	W	
		D7–3	**–**	reserved	–	–	–	
		D2	**CHIP_ERS_EN**	Flash chip erase enable	1 Enable 0 Disable	0	R/W	
		D1	**SCT_ERS_EN**	Flash sector erase enable	1 Enable 0 Disable	0	R/W	
		D0	**WR_EN**	Flash programming enable	1 Enable 0 Disable	0	R/W	
FLASHC Sector Address Register (FLASH_ADDR)	**0x5802** (16 bits)	D15–7	**–**	reserved	–	–	–	0 when being read.
		D6–0	**FLS_ADDR [6:0]**	Erase sector address	Address[16:10]	0x0	R/W	
FLASHC Wait Register (FLASH_WAIT)	**0x5804** (16 bits)	D15–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**FLS_WAIT [2:0]**	Flash read access wait cycle setup	FLS_WAIT[2:0] Wait cycle 0x7 7 cycles : : 0x0 0 cycles	0x7	R/W	
FLASHC Protect Register (FLASH_PROT)	**0x5810** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**FLS_PROT [7:0]**	FLASHC register protect flag	Writing 10010110 (0x96) removes the write protection of the FLASHC Control Register (0x5800) and Trap Table Base Registers (0x5814–0x5816). Writing another value set the write protection.	0x0	R/W	
Trap Table Base Register 0 (TTBR_LOW)	**0x5814** (16 bits)	D15–8	**TTBR[15:8]**	Trap table base address A[15:8]	0x0–0xff	0x0	R/W	
		D7–0	**TTBR[7:0]**	Trap table base address A[7:0] (fixed at 0)	0x0	0x0	R	
Trap Table Base Register 1 (TTBR_HIGH)	**0x5816** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TTBR[23:16]**	Trap table base address A[23:16]	0x0–0xff	0x2	R/W	

0xffff90 S1C17 Core I/O

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Debug RAM Base Register (DBRAM)	**0xffff90** (32 bits)	D31–24	–	Unused (fixed at 0)	0x0	0x0	R	
		D23–0	**DBRAM[23:0]**	Debug RAM base address	0x0	0x0	R	

Appendix B Multiplier

B.1 Outline

The S1C17501 has an embedded coprocessor that provides a signed 16 × 16-bit multiplication function and a signed/unsigned 16 × 16-bit + 32-bit MAC (Multiplication and Accumulation) function with overflow detection. This section explains how to use these functions.

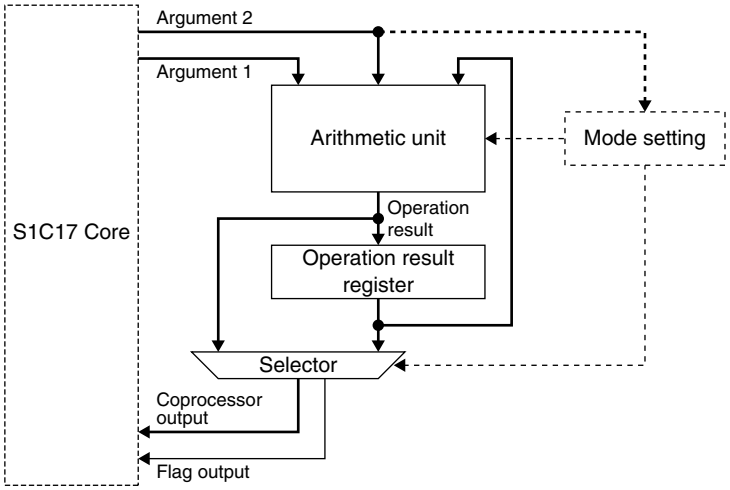


Figure B.1.1 Multiplier Block Diagram

The coprocessor provides a high-speed calculation ability as the table below.

Table B.1.1 Operating Frequency and Number of MAC Operation Cycles

Frequency	CCLK ≤ 24 MHz	24 MHz < CCLK ≤ 48 MHz
MAC_WAIT (D0/MAC_WAIT register)	0	1 (default)
Number of cycles for multiplication	1 cycle	2 cycles
Number of cycles for MAC	1 cycle	2 cycles

∗ **MAC_WAIT**: MAC Wait Cycle Setup Bit in the MAC Wait Control (MAC_WAIT) Register (D0/0x5014)

See Section III.2.7, "Setting Wait Cycles for Internal Devices," for MAC_WAIT.

B.2 Operation Mode and Output Mode

The multiplier operates according to the operation mode specified by the application program. As listed in Table B.2.1, the multiplier supports six operations.
The multiplication and MAC results are 32-bit data, therefore, the C17 Core cannot read them in one access. The output mode is provided to specify the high-order 16 bits or low-order 16 bits of the operation results to be read from the multiplier.
The operation and output modes can be specified with a 7-bit data by writing it to the mode setting register in the multiplier. Use a "`ld.cw`" instruction for this writing.

`ld.cw %rd,%rs` %rs[6:0] is written to the mode setting register. (%rd: not used)
`ld.cw %rd,`*`imm7`* *imm7*[6:0] is written to the mode setting register. (%rd: not used)

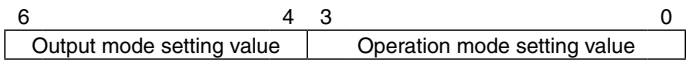


Figure B.2.1 Mode Setting Register

Table B.2.1 Mode Settings

Setting value (D[6:4])	Output mode	Setting value (D[3:0])	Operation mode
0x0	**16 low-order bits output mode** The low-order 16-bits of operation results can be read as the coprocessor output.	0x0	**Initialize mode 0** Clears the operation result register to 0x0.
0x1	**16 high-order bits output mode** The high-order 16-bits of operation results can be read as the coprocessor output.	0x1	**Initialize mode 1** Loads the 16-bit augend into the low-order 16 bits of the operation result register.
0x2–0x7	Reserved	0x2	**Initialize mode 2** Loads the 32-bit augend into the operation result register.
		0x3	**Operation result read mode** Outputs the data in the operation result register without computation.
		0x4	**Unsigned multiplication mode** Performs unsigned multiplication.
		0x5	**Signed multiplication mode** Performs signed multiplication.
		0x6	Reserved
		0x7	**Signed MAC mode** Performs signed MAC operation.
		0x8–0xf	Reserved

B.3 Multiplication

The multiplication function performs "A (32 bits) = B (16 bits) × C (16 bits)."
To perform a multiplication, set the operation mode to 0x4 (unsigned multiplication) or 0x5 (signed multiplication). Then send the 16-bit multiplicand (B) and 16-bit multiplier (C) to the multiplier using a "`ld.ca`" instruction. The one-half (16 bits according to the output mode) result (A[15:0] or A[31:16]) and the flag status will be returned to the CPU registers. Another one-half should be read by setting the multiplier into operation result read mode.

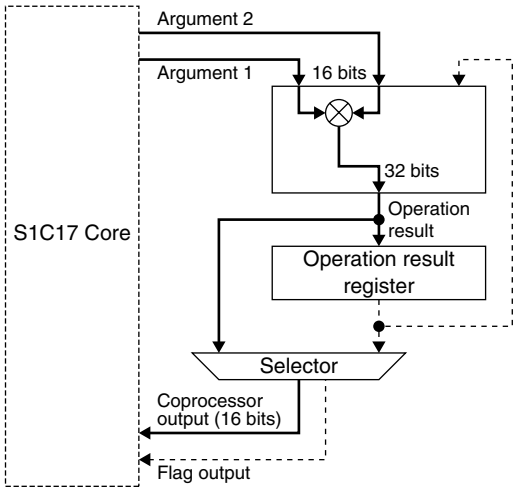


Figure B.3.1 Data Path in Multiplication Mode

Table B.3.1 Operation in Multiplication Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x04 or 0x05	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs %rd ← res[15:0]	psr (CVZN) ← 0b0000	The operation result register keeps the operation result until it is rewritten by other operation.
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* %rd ← res[15:0]		
0x14 or 0x15	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs %rd ← res[31:16]		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* %rd ← res[31:16]		

res: operation result register

Example:

```
ld.cw %r0,0x4   ; Sets the modes (unsigned multiplication mode and 16 low-order bits output mode).
ld.ca %r0,%r1   ; Performs "res = %r0 × %r1" and loads the 16 low-order bits of the result to %r0.
ld.cw %r0,0x13  ; Sets the modes (operation result read mode and 16 high-order bits output mode).
ld.ca %r1,%r0   ; Loads the 16 high-order bits of the result to %r1.
```

B.4 MAC

The MAC function performs "A (32 bits) = B (16 bits) × C (16 bits) + A (32 bits)."
Before performing a MAC operation, the initial value (A) must be set to the operation result register.
To clear the operation result register (A = 0), just set the operation mode to 0x0. It is not necessary to send 0x0 to the multiplier with another instruction.
To load a 16-bit value or a 32-bit value to the operation result register, set the operation mode to 0x1 or 0x2, respectively. Then send the initial value to the multiplier using a "`ld.cf`" instruction.

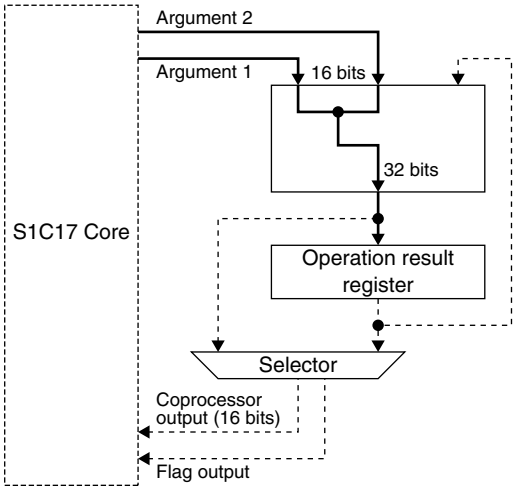


Figure B.4.1 Data Path in Initialize Mode

Table B.4.1 Initializing the Operation Result Register

Mode setting value	Instruction	Operations	Remarks
0x0	–	res[31:0] ← 0x0	Setting the operating mode executes the initialization without sending data.
0x1	`ld.cf %rd,%rs`	res[31:16] ← 0x0 res[15:0] ← %rs	
	(`ext` *`imm9`*) `ld.cf %rd,`*`imm7`*	res[31:16] ← 0x0 res[15:0] ← *imm7/16*	
0x2	`ld.cf %rd,%rs`	res[31:16] ← %rd res[15:0] ← %rs	
	(`ext` *`imm9`*) `ld.cf %rd,`*`imm7`*	res[31:16] ← %rd res[15:0] ← *imm7/16*	

res: operation result register

To perform a MAC operation, set the operation mode to 0x7 (signed MAC). Then send the 16-bit multiplicand (B) and 16-bit multiplier (C) to the multiplier using a "`ld.ca`" instruction. The one-half (16 bits according to the output mode) result (A[15:0] or A[31:16]) and the flag status will be returned to the CPU registers. Another one-half should be read by setting the multiplier into operation result read mode.
The overflow (V) flag in the PSR may be set to 1 according to the result. Other flags are set to 0.
When repeating the MAC operation without operation result read mode inserted, send multiplicand and multiplier data for number of required times. In this case it is not necessary to set the MAC mode every time.

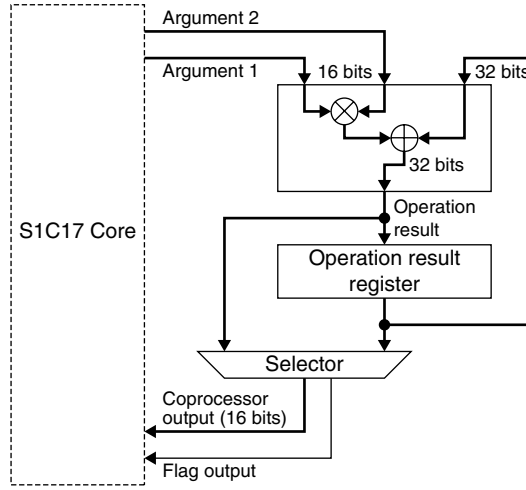


Figure B.4.2 Data Path in MAC Mode

Table B.4.2 Operation in MAC Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x07	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs + res[31:0] %rd ← res[15:0]	psr (CVZN) ← 0b0100 if an overflow has occurred Otherwise psr (CVZN) ← 0b0000	The operation result register keeps the operation result until it is rewritten by other operation.
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* + res[31:0] %rd ← res[15:0]		
0x17	`ld.ca %rd,%rs`	res[31:0] ← %rd × %rs + res[31:0] %rd ← res[31:16]		
	(`ext` *imm9*) `ld.ca %rd,`*imm7*	res[31:0] ← %rd × *imm7/16* + res[31:0] %rd ← res[31:16]		

res: operation result register

Example:

```
ld.cw %r0,0x7   ; Sets the modes (signed MAC mode and 16 low-order bits output mode).
ld.ca %r0,%r1   ; Performs "res = %r0 × %r1 + res" and loads the 16 low-order bits of the result to %r0.
ld.cw %r0,0x13  ; Sets the modes (operation result read mode and 16 high-order bits output mode).
ld.ca %r1,%r0   ; Loads the 16 high-order bits of the result to %r1.
```

Conditions to set the overflow (V) flag

An overflow occurs in a MAC operation and the overflow flag is set to 1 when the signs of the multiplication result, operation result register value, and multiplication & accumulation result match the following conditions:

Table B.4.3 Conditions to Set the Overflow (V) Flag

Mode setting value	Sign of multiplication result	Sign of operation result register value	Sign of multiplication & accumulation result
0x07	0 (positive)	0 (positive)	1 (negative)
0x07	1 (negative)	1 (negative)	0 (positive)

An overflow occurs when a MAC operation performs addition of positive values and a negative value results, or it performs addition of negative values and a positive value results. The coprocessor holds the operation result when the overflow (V) flag is cleared.

Conditions to clear the overflow (V) flag

The overflow (V) flag that has been set will be cleared when an overflow has not been occurred during execution of the "`ld.ca`" instruction for MAC operation or when the "`ld.ca`" or "`ld.cf`" instruction is executed in an operation mode other than operation result read mode.

B.5 Reading Results

The "ld.ca" instruction cannot load a 32-bit operation result to a CPU register, so a multiplication or MAC operation returns the one-half (16 bits according to the output mode) result (A[15:0] or A[31:16]) and the flag status to the CPU registers. Another one-half should be read by setting the multiplier into operation result read mode. The operation result register keeps the loaded operation result until it is rewritten by other operation.

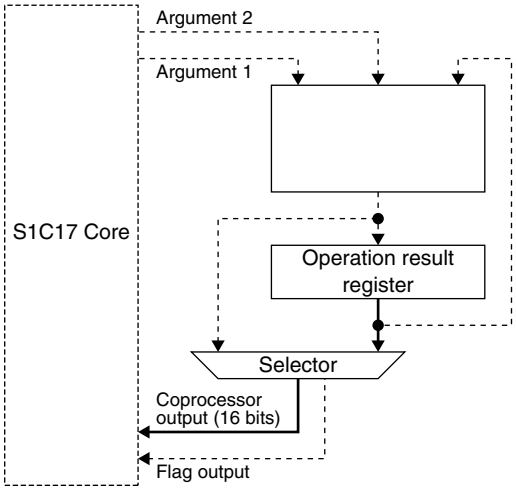


Figure B.5.1 Data Path in Operation Result Read Mode

Table B.5.1 Operation in Operation Result Read Mode

Mode setting value	Instruction	Operations	Flags	Remarks
0x03	ld.ca %rd,%rs	%rd ← res[15:0]	psr (CVZN) ← 0b0000	This operation mode does not affect the operation result register.
	ld.ca %rd,*imm7*	%rd ← res[15:0]		
0x13	ld.ca %rd,%rs	%rd ← res[31:16]		
	ld.ca %rd,*imm7*	%rd ← res[31:16]		

res: operation result register

Appendix C Power Saving

Current consumption depends, to a large degree, on the CPU operating mode, operating clock frequency, and the peripheral circuits to be activated. This chapter summarizes the control to save power.
Figure C.1 shows the S1C17501 clock system.

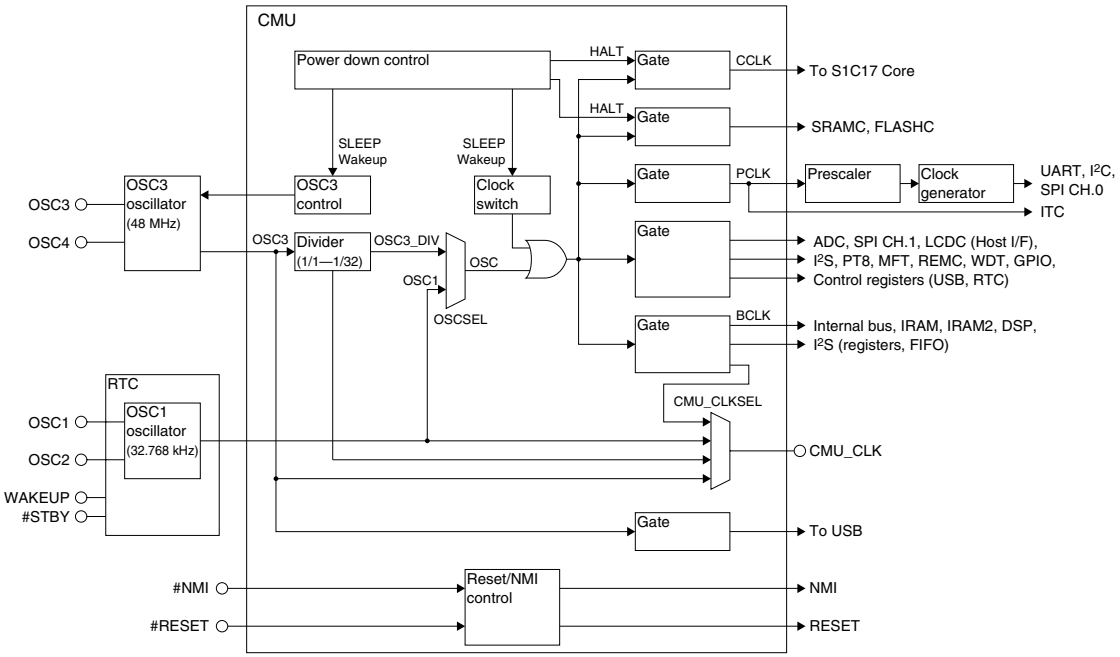


Figure C.1 Clock System

The following shows the clock systems that can be controlled with software and power saving control methods. For details of control registers and control methods, see the chapter for each module.

System sleep (disabling all clocks)

- Executing the `slp` instruction
  Execute the `slp` instruction if all of the system can be stopped. In SLEEP mode, the CPU stops operating and the CMU stops supplying a clock to each functional module (see Section II.2.6 for more information). Therefore, all peripheral circuits (except the OSC1 oscillator circuit and RTC) stop operating.
  The OSC3 oscillator stops oscillating in SLEEP mode if OSC3OFF (D7/CMU_SYSCLKCTL register) is set to 1.
  The CPU is reawaken from SLEEP mode (when WAKEUPWT (D4/CMU_SYSCLKCTL register) = 1) by initial reset, RTC interrupt (level triggered), #NMI signal, or other interrupt from an external device (port input interrupt with level triggered).

  * **OSC3OFF**: OSC3 Disable During SLEEP Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D7/0x4900)
  * **WAKEUPWT**: Wakeup-Wait Function Enable Bit in the System Clock Control (CMU_SYSCLKCTL) Register (D4/0x4900)

System clock

- Selecting the clock source (CMU module)
  Either OSC3 or OSC1 can be selected as the system clock source. If the application can process the task with a low-speed clock, select OSC1 as the system clock source to reduce current consumption.
- Disabling the OSC3 oscillator circuit (CMU module)
  Enable the oscillator configured as the system clock source and disable another oscillator if possible. Using OSC1 for the system clock and disabling the OSC3 oscillator circuit achieves more reduction of current consumed.
- Selecting a low clock gear (CMU module)
  The CMU module provides clock gears to set the system clock speed to 1/1 to 1/32 of the OSC3 clock. By running the S1C17501 with the lowest speed required for the application's task, current consumption can be reduced.

CPU clock (CCLK)

- Executing the `halt` instruction
  Execute the `halt` instruction if there is no task to be processed by the CPU such as when the display on the LCD is only required or when the CPU is waiting an interrupt. Although the CPU enters HALT mode and stops operating, the peripheral modules keep the status when the `halt` instruction is executed. So the peripheral modules used to generate an interrupt can be made to be run. Power saving effect will be enhanced by disabling the unnecessary oscillator and peripheral modules before executing the `halt` instruction. The CPU reactivates from HALT mode by an interrupt from the ports or peripheral modules that are being operated in HALT mode.

Peripheral clocks

- Disabling peripheral clocks (CMU, CLG, and PSC modules)
  The peripheral clock supply can be disabled if the peripheral modules listed below can be placed in standby state.

  - FLASHC clock in HALT mode (CMU)
  - SRAMC clock in HALT mode (CMU)
  - Core peripheral clocks for PSC, CLG, ITC (CMU)
  - USB clocks (CMU)
  - PT8 and I^2S clocks (CMU)
  - MFT clock (CMU)
  - SPI CH.1 clock (CMU)
  - REMC clock (CMU)

- ADC clock (CMU)
- WDT clock (CMU)
- I/O port clock (CMU)
- RTC register clock (CMU)
- CLG clocks (PSC)
- UART clocks (CLG)
- SPI CH.0 clock (CLG)
- I^2C clock (CLG)

Table C.1 lists the clock control conditions and how to suspend/resume the CPU operation.

Table C.1 List of Clock Control Conditions

Current consumption	OSC1	OSC3	CPU (CCLK)	Peripherals	CPU suspending method	CPU resuming method
↑ Low	Oscillating	Stop	Stop	Stop	`slp` instruction	1
	Oscillating	Stop	Stop	Stop (only RTC is running)	`slp` instruction	1, 2
	Oscillating (System clock)	Stop	Stop	Stop	`halt` instruction	1, 2
	Oscillating (System clock)	Stop	Stop	Run	`halt` instruction	1, 2, 3
	Oscillating (System clock)	Stop	Run	Run		
	Oscillating	Oscillating (System clock)	Stop	Run	`halt` instruction	1, 2, 3
	Oscillating	Oscillating (System clock)	Run (low gear)	Run		
High ↓	Oscillating	Oscillating (System clock)	Run (OSC3•1/1)	Run		

Clearing HALT and SLEEP modes (CPU resuming methods)

1. Resuming by a port input interrupt (level triggered), #RESET or #NMI
   The CPU resumes operating by occurrence of a cause of port input interrupt (level triggered), #RESET, #NMI or a debug interrupt (issuing an ICD forced break). See Section II.2.8 for details.
2. Resuming by the RTC
   The CPU resumes operating by occurrence of a cause of RTC interrupt (level triggered). See Sections II.2.8 and II.5.4 for details.
3. Resuming by a peripheral
   The CPU resumes operating by occurrence of a cause of interrupt in a peripheral whose interrupt is enabled by the interrupt controller. If the IE flag in the CPU has been set to 0, the CPU does not accept the interrupt request and starts executing the instructions that follow the `halt` instruction. If the IE flag has been set to 1, the CPU executes the interrupt handler.

Battery backup mode

- Turning the system power (V_{DD}, AV_{DD}) off
  If the system uses separated V_{DD} and $RTCV_{DD}$ power sources, it is possible to operate only the $RTCV_{DD}$ system circuits (RTC, OSC1, and IRAM2) with the system power (V_{DD}, AV_{DD}) turned off to reduce current consumption. Turning the system power off reduces leakage current that cannot be reduced in SLEEP mode.
  The #STBY and WAKEUP pins that have been provided in the RTC module are used for controlling this function. Refer to Section II.5.6, "WAKEUP and #STBY Pins," for more information on the control.

Note: The battery backup mode can help reduce current consumption when many parts are used in the external circuit or if the V_{DD}/AV_{DD} system circuits will be deactivated for a relatively long time. Depending on the system configuration, the SLEEP mode may be efficient for saving power. Take these conditions into consideration at the system design stage.

Revision History

Code No.	Page	Contents
411525601	All	New anactment
411525602	I-5-5	Descriptions modified. ipa.d→jpa.d
	VI-1-6	Descriptions modified. (2) RDRY = 1, RD2B = 0...Therefore, be sure to read the receive data buffer before an overrun error occurs.
	VI-1-7	Descriptions modified. When the RXEN bit is set to 0, the transmit data buffer is in empty status (data is cleared if any remains).
	VI-1-8	Descriptions modified. However, if the receive data...shift register will not be sent to the buffer and generate an overrun error.
	VI-1-14	Descriptions modified. FER is reset by writing 1....OER is reset by writing 1.
	VI-1-19	Descriptions modified. Writing 0 to RXEN also clears the transmit data buffer.
	VI-1-21	Descriptions modified. •When the RXEN bit is...the transmit data buffer is cleared (initialized). Therefore, make sure that the buffers do not contain any data waiting for transmission before writing 0 to the RXEN bit.
	VI-2-1	Descriptions modified. 7-bit slave addressing→7-bit/10-bit slave addressing
	VI-2-5	Descriptions modified. two parts→two parts or three parts
	VI-2-6	Figure VI.2.5.2 modified.
	VI-2-7	Descriptions modified. In the 9th clock cycle, ...during the response time, the correct communication can not be performed.
	VI-2-8	Descriptions modified. After the stop condition has been generated,...and transfer data at that point cannot be guaranteed.
	VI-2-9	Figure VI.2.5.5 modified.
	VI-2-10	Figure VI.2.5.7 and Figure VI.2.5.8 modified.
	VI-2-11	Figure VI.2.5.9 modified.
	VI-3-3	Descriptions deleted. Since the internal circuit operates with...clock is differentiated and used to sync with the PCLK clock. Descriptions modified. Note: The duty ratio of the clock input via the SPI_SCK0 pin must be 50%. Figure VI.3.3.2 deleted.
	VI-3-4	Descriptions added. Figure VI.3.4.2 added.
	VI-3-6, VI-3-7	Figure VI.3.5.1 and Figure VI.3.5.2 deleted. Figure VI.3.5.1 added.
	VI-3-7	Descriptions modified. After data transfer...transferred cannot be guaranteed if SPEN is set to 0 during transmitting/receiving.
	VI-3-8	Descriptions modified. When SPTBE = 0,→When SPTBE = 1,
	VI-3-12	Descriptions added. Note: Make sure that SPEN is set to 1...to the SPI_TXD register to start data transmission/reception.
	VI-4-3	Descriptions modified. Note: The duty ratio of the clock input via the SPI_SCK0 pin must be 50%. Figure VI.4.3.1 deleted.
	VI-4-8	Descriptions modified. When SPTBE = 0,→When SPTBE = 1,

EPSON

International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.
2580 Orchard Parkway,
San Jose, CA 95131, USA
Phone: +1-800-228-3964 Fax: +1-408-922-0238

EUROPE

EPSON EUROPE ELECTRONICS GmbH
Riesstrasse 15, 80992 Munich,
GERMANY
Phone: +49-89-14005-0 Fax: +49-89-14005-110

ASIA

EPSON (CHINA) CO., LTD.
7F, Jinbao Bldg., No.89 Jinbao St.,
Dongcheng District,
Beijing 100005, CHINA
Phone: +86-10-8522-1199 Fax: +86-10-8522-1125

SHANGHAI BRANCH
7F, Block B, Hi-Tech Bldg., 900 Yishan Road,
Shanghai 200233, CHINA
Phone: +86-21-5423-5577 Fax: +86-21-5423-4677

SHENZHEN BRANCH
12F, Dawning Mansion, Keji South 12th Road,
Hi-Tech Park, Shenzhen 518057, CHINA
Phone: +86-755-2699-3828 Fax: +86-755-2699-3838

EPSON HONG KONG LTD.
Unit 715-723, 7/F Trade Square, 681 Cheung Sha Wan Road,
Kowloon, Hong Kong.
Phone: +852-2585-4600 Fax: +852-2827-4346

EPSON TAIWAN TECHNOLOGY & TRADING LTD.
14F, No. 7, Song Ren Road,
Taipei 110, TAIWAN
Phone: +886-2-8786-6688 Fax: +886-2-8786-6660

EPSON SINGAPORE PTE., LTD.
1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 Fax: +65-6271-3182

SEIKO EPSON CORP.
KOREA OFFICE
5F, KLI 63 Bldg., 60 Yoido-dong,
Youngdeungpo-Ku, Seoul 150-763, KOREA
Phone: +82-2-784-6027 Fax: +82-2-767-3677

SEIKO EPSON CORP.
MICRODEVICES OPERATIONS DIVISION

Device Sales & Marketing Dept.
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 Fax: +81-42-587-5117

Document Code: 411525602
First Issue November 2008
Revised April 2011Ⓗ