EPSON
EXCEED YOUR VISION

CMOS 32-BIT SINGLE CHIP MICROCONTROLLER

S1C33L27
Technical Manual

SEIKO EPSON CORPORATION

Rev.1.1

NOTICE

Configuration of product number

Devices

S1 C 33209 F 00E1 00

Packing specifications
[00 : Besides tape & reel
0A : TCP BL 2 directions
0B : Tape & reel BACK
0C : TCP BR 2 directions
0D : TCP BT 2 directions
0E : TCP BD 2 directions
0F : Tape & reel FRONT
0G : TCP BT 4 directions
0H : TCP BD 4 directions
0J : TCP SL 2 directions
0K : TCP SR 2 directions
0L : Tape & reel LEFT
0M : TCP ST 2 directions
0N : TCP SD 2 directions
0P : TCP ST 4 directions
0Q : TCP SD 4 directions
0R : Tape & reel RIGHT
99 : Specs not fixed]

Specification

Package
[D: die form; F: QFP, B: BGA]

Model number

Model name
[C: microcomputer, digital products]

Product classification
[S1: semiconductor]

Development tools

S5U1 C 33000 H2 1 00

Packing specifications
[00: standard packing]

Version
[1: Version 1]

Tool type
[Hx : ICE
Dx : Evaluation board
Ex : ROM emulation board
Mx : Emulation memory for external ROM
Tx : A socket for mounting
Cx : Compiler package
Sx : Middleware package
Yx : Writer software]

Corresponding model number
[33L01: for S1C33L01]

Tool classification
[C: microcomputer use]

Product classification
[S5U1: development tool for semiconductor products]

– Contents –

1 Overview

The S1C33L27 is a 32-bit application specific RISC controller. It is suitable for applications that require an abundance of input/output ports and serial interfaces, USB, ADC, and a display panel, such as electronic dictionaries and control panels on OA/FA equipment.
The S1C33L27 incorporates an LCD controller and VRAM supporting QVGA display in single-chip. Adding an external SDRAM expands this capability into more higher resolution and displayable colors (e.g., VGA display, 16M-color display). It provides an interface to communicate with a built-in RAM type LCD driver.

1.1 Features

The features of the S1C33L27 are outlined below.

CPU

- EPSON original C33 PE 32-bit RISC CPU-Core
- Maximum operating frequency: 60 MHz
- Internal two-stage pipeline
- Instruction set: 128 instructions (16-bit fixed length)

Internal Memories

- A0RAM (general-purpose RAM)
  - 20K bytes (including 1K-byte instruction cache and 1K-byte data cache)
  - Usable as a general-purpose RAM when not used as cache RAM
- IVRAM (internal VRAM)
  - 32K bytes
  - Configurable as a general-purpose RAM in Area 0 or a RAM for the calculation module
- DSTRAM (DMA descriptor table RAM)
  - 2K bytes
  - Configurable as a RAM for the calculation module
- BBRAM (battery backup RAM)
  - 16 bytes
  - The RAM contents can be maintained while the system power is off using the separated power supply for RTC.

Input clock

- High-speed clock (OSC3)
  - Maximum input clock frequency: 48 MHz
  - Generated by the oscillator circuit (using an external crystal or ceramic resonator) or an external clock is input.
- Low-speed clock (OSC1)
  - 32.768 kHz (typ.) clock for RTC and low-power operations
  - Generated by the oscillator circuit (using an external crystal resonator) or an external clock is input.

Cache Controller (CCU)

- 1K-byte instruction cache and 1K-byte data cache that adopts a four-way associative method
- LRU replacement algorithm
- Automatic lock function during debug mode and the interrupt process of specified priority
- Write through function with a 1-word write buffer

DMA Controller (DMAC)

- Eight channels of table DMA
- Supports table reloading and low-priority channel pausing functions.
- 24 hardware trigger sources and 8 software trigger sources

SRAM Controller (SRAMC)

- Allows connection of SRAM, ROM, and Flash memories.
- 26-bit address bus and 8/16-bit selectable data bus
- Up to six chip enable signals are available to connect external devices.
- Up to 64M-byte (A[25:0]) address space can be accessible with each chip enable signal.
- Programmable bus access wait cycle (0 to 15 cycles)
- Little endian access
- Memory mapped I/O
- Supports both A0 and BS (Bus Strobe) type devices.
- Supports external wait request via the #WAIT pin.

SDRAM Controller (SDRAMC)

- Supports SDRAM interface.
- Supports only SDRAM devices with 8/16-bit data bus.
  - Minimum configuration: 16M bits (2MB), 16-bit SDRAM × 1
  - Maximum configuration: 512M bits (64MB), 16-bit SDRAM × 1
- CAS latency: one, two, or three programmable
- Supports two-burst read and single write operations.
- Equipped with a four-stage × 16-bit DQB (Data Queue Buffer).
- Supports up to four SDRAM banks and bank active mode.
- Built-in 12-bit auto-refresh counter
- Intelligent self-refresh function for low power operation
- Arbitrates external bus accesses by AHB-1 (CPU, DMAC, HIF) and AHB-2 (LCDC).

Host Processor Interface (HIF)

- 8- or 16-bit asynchronous parallel interface to control the S1C33L27 by an external host processor
- Provides semaphore registers.

Clock Management Unit/Oscillators/PLL (CMU)

- Selects the system clock source (OSC3, PLL, or OSC1).
- Turns the OSC3 and OSC1 oscillator circuits on and off.
- Controls frequency multiplication rate of the PLL (×1 to ×16).
- Controls clocks according to the standby mode (SLEEP and HALT).
- Controls the external clock.
- Controls clock supply to the core and peripheral modules.
- OSC3 oscillator circuit
  - Crystal oscillation: 5 MHz min. to 48 MHz max.
  - Ceramic oscillation: 5 MHz min. to 48 MHz max.
  - External clock input: 2 MHz min. to 48 MHz max.

  * A 48 MHz clock source with 0.25% of accuracy should be connected for using the USB function.
  * Before using a ceramic resonator, please be sure to contact Murata Manufacturing Co., Ltd. for further information on conditions of use for ceramic resonators.
- PLL
  - PLL input frequency: 5 MHz min. to 48 MHz max. (OSC3 ×1, ×1/2, ×1/3, ... ×1/9, ×1/10)
  - PLL output frequency: 20 MHz min. to 60 MHz max.
  - Multiplication rate: ×1, ×2, ×3, ... ×15, ×16
- OSC1 oscillator circuit
  - Crystal oscillation: 32.768 kHz typ.
  - External clock input: 32.768 kHz typ.

Interrupt Controller (ITC)

- Five non-maskable interrupts
- 34 maskable interrupts (including four software exceptions)

16-bit Audio PWM Timer (T16P)

- Two channels of 16-bit timer/counter with PWM output function
- Three bit division modes are provided. (10 bits + 6 bits, 9 bits + 7 bits, 8 bits + 8 bits)
- Can support 8, 16, 22.05, 32, 44.1, and 48 kHz sampling rates.
- PWM function that can handle 8-bit and 16-bit PCM data with 8 k to 48 kbps sampling rates
- Provides fine mode to improve the precision of the pulse width.
- Supports a digital volume control function.
- Can generate two types of compare-match interrupts.
- Supports DMA transfer.

Fine Mode 8-bit Timers (T8F)

- Six channels of 8-bit timer with fine mode (presettable down counter)
- Clocks generated with the counter underflow can be output to internal devices (USI, USIL, ADC, and UART).
- Each timer can generate underflow interrupts.

16-bit PWM Timer (T16A6)

- Four channels of 16-bit timer with a counter capture/comparison functions
- Each channel has built-in two comparison/capture data buffers.
- Can generate compare/capture interrupts.
- The counter clock can be selected from the system clock, OSC3 clock, and OSC1 clock.
- Supports DMA transfer.

Watchdog Timer (WDT)

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

Real Time Clock (RTC)

- Contains time counters (seconds, minutes, and hours) and calendar counters (days, days of the week, months, and year).
- 24-hour or 12-hour mode can be selected.
- Operates with an independent power supply ($RTCV_{DD}$) separated from system power (operable while the system power is off).
- Provides the WAKEUP output pin and #STBY input pin to control standby mode.
- Can generate clock interrupts.

Universal Serial Interface (USI)

- Three channels of multi-serial interface that can be used as a UART, SPI, or I^2C module
- Contains 1-byte receive data buffer and 1-byte transmit data buffer.
- UART mode
  - Character length: 7 or 8 bits
  - Parity mode: even, odd, or no parity
  - Stop bit: 1 or 2 bits (start bit: 1 bit fixed)
  - Supports both MSB first and LSB first modes.
  - Parity error, framing error, and overrun error detectable
  - Can generate receive buffer full, transmit buffer empty, and receive error interrupts.
  - Supports DMA transfer.
- SPI mode
  - Supports both master and slave modes.
  - Data length: 8 or 9 bits (master mode), 8 bits fixed (slave mode)
  - Supports both MSB first and LSB first modes.
  - Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
  - Can generate receive buffer full, transmit buffer empty, and overrun error interrupts.
  - Supports DMA transfer.

- I²C mode
  - Supports both master (single master only) and slave modes.
  - 7-bit addressing mode (10-bit addressing is possible by software control.)
  - Supports clock stretch/wait functions.
  - Can generate start/stop, data transfer, ACK/NAK transfer, and overrun error interrupts.

Universal Serial Interface with Built-in RAM LCD interface (USIL)

- Multi-serial interface that can be used as a UART, SPI, I²C, or built-in RAM LCD interface module
- Contains 1-byte receive data buffer and 1-byte transmit data buffer.
- UART mode
  - Same features as USI
- SPI mode
  - Data length: 8 bits fixed
  - Other features are the same as USI.
- I²C mode
  - Same features as USI
- LCD SPI mode
  - Data length is configurable for 8 bits, 16 bits, 18 bits (4 data format) and 24 bits + CMD bit.
  - CMD bit or A0 is selectable.
  - Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
  - Can generate transmit buffer empty interrupts.
  - Supports DMA transfer.
- LCD parallel interface mode
  - Provides 8-bit data bus, #CS, #RD, #WR and A0 control signals.
  - Supports byte read/write access mode only.
  - Can generate transmit buffer empty and receive buffer full interrupts.
  - Supports DMA transfer for both data transmission and reception.

UART

- One channel of UART is available.
- Conforms to IrDA 1.0.
- 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: 7 or 8 bits, parity mode: even, odd, or no parity, stop bit: 1 or 2 bits
- Parity error, framing error, and overrun error detectable
- Can generate receive buffer full, transmit buffer empty, and receive error interrupts.

I²S Bus Interface (I2S)

- General-purpose I²S audio bus interface with one input and one output channels
- Contains a 24-byte FIFO (24 bits × 2 channels (L & R) × 4).
- Resolution: 16 bits and 24 bits (PCM data output format)
- Clock polarity and data shift direction (MSB first/LSB first) are software configurable.
- Can generate FIFO empty interrupts for the output channel (half empty, whole empty, or one empty) and FIFO full interrupts for the input channel (whole full or one data).
- Supports DMA transfer.

Card Interface (CARD)

- Generates 8-bit SLC/MLC NAND Flash interface signals.
- Includes a Reed-Solomon codec to support an EDC (Error Detection Code) and an ECC (Error Correction Code) functions.
- A #CE area can be selected to connect a NAND Flash.

MMC/SD/SDHC Card Interface (SD_MMC)

- SD/SDHC card controller compatible with SD Memory Card Physical Layer Specification Version 2.00.
- MMC controller compatible with MultiMediaCard System Specification Version 2.2.

- Variable clock rate up to 30 MHz.
- Supports 4-bit (wide bus) and 1-bit SD bus interface.
- CRC7 and CRC16 generators
- Supports DMA transfer.

Note: Please join the SD Association (SDA) when handling SD and SDHC cards.

Infrared Remote Controller (REMC)

- Outputs a modulated carrier signal and inputs remote control pulses.
- Embedded carrier signal generator and data length counter
- Can generate counter underflow interrupts for data transmission and input rising/falling edge detection interrupts for data reception.

LCD Controller (LCDC)

- Supports STN LCD panels with 4/8-bit data lines or TFT LCD panels with up to 24-bit data lines.
- Supports various panel resolutions, such as 640 × 480 pixels (VGA) and 320 × 240 pixels (QVGA) (can be configured according to the panel used).
- Supports up to 16M-color (for color TFT), 4K-color (for color STN), and 16-level gray scale (for monochrome STN) display modes.
- Typical display configuration when the internal VRAM (20KB) is used
  - 320 × 240 pixels, 2 bpp (4-level gray scale display)
- Display configuration when an external memory is used
  - 320 × 240 pixels, 16 bpp (QVGA 64K-colors display)
  - 400 × 240 pixels, 16 bpp (WQVGA 64K-colors display)
  - 640 × 480 pixels, 16 bpp (VGA 64K-colors display)
- Two-image overlay display via the Picture-in-Picture Plus function
- Virtual display function to handle images with a different resolution from the LCD panel (any area in the virtual screen can be displayed on the LCD.)

A/D Converter (ADC10)

- 10-bit successive approximation type A/D converter
- Up to eight analog input channels (chip and PFBGA12U-180 package)
- Up to four analog input channels (TQFP24/QFP20-144pin and TQFP15-128pin packages)
- Conversion time: 10 µs min. (when 2 MHz input clock is selected)
  1,250 µs max. (when 16 kHz input clock is selected)
- Can generate conversion completion and data overwrite error interrupts.

USB Function Controller (USB)

- Supports USB2.0 full speed (12M bps) mode.
- Provides auto negotiation function.
- Supports control, bulk, isochronous and interrupt transfers.
- Supports four general-purpose endpoints and endpoint 0 (control).
- Embedded 1K-byte programmable FIFO
- Can generate USB interrupts.
- Supports DMA transfer.

General-purpose I/O Ports (GPIO)

- Maximum 91 I/O ports and eight input ports are available (chip and PFBGA12U-180 package).
- Maximum 72 I/O ports and four input ports are available (TQFP24/QFP20-144pin package).
- Maximum 56 I/O ports and four input ports are available (TQFP15-128pin package).
- Can generate maximum 8 port input interrupts from the 64 I/O ports selected and key input interrupts from the predefined 32 ports.

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

Calculation Module (CALC)

- Multiply and accumulation (MAC)
- Matrix computation
- Affine transformation
- Butterfly computation
- Supports signed/unsigned 32-bit integer operation mode and signed 16-bit fixed-point values operation mode with saturation processing.

Operating Voltage

- HV_{DD} (I/O power voltage)
  2.7 V to 3.6 V (3.3 V typ.)
  or 3.0 V to 3.6 V (3.3 V typ.) when the USB module is used.
- AV_{DD} (analog power voltage)
  2.7 V to 3.6 V (3.3 V typ.)
- LV_{DD} (internal logic/internal memory power voltage)
  1.65 V to 1.95 V (1.8 V typ.)
  or 1.7 V to 1.9 V (1.8 V typ.) when a ceramic resonator is used.
- $PLLV_{DD}$ (PLL power voltage)
  1.65 V to 1.95 V (1.8 V typ.)
  or 1.7 V to 1.9 V (1.8 V typ.) when a ceramic resonator is used.
- $RTCV_{DD}$ (RTC/BBRAM power voltage)
  1.65 V to 1.95 V (1.8 V typ.)
  or 1.7 V to 1.9 V (1.8 V typ.) when a ceramic resonator is used.

∗ $LV_{DD} = PLLV_{DD} = RTCV_{DD}$, $HV_{DD} = AV_{DD}$
The S1C33L27 does not support 5 V tolerant I/O.

Operating Temperatures

- -40 to 85°C
- 0 to 70°C when the USB module is used or when a ceramic resonator is used.

Power Consumption

(No I/O current is included.)

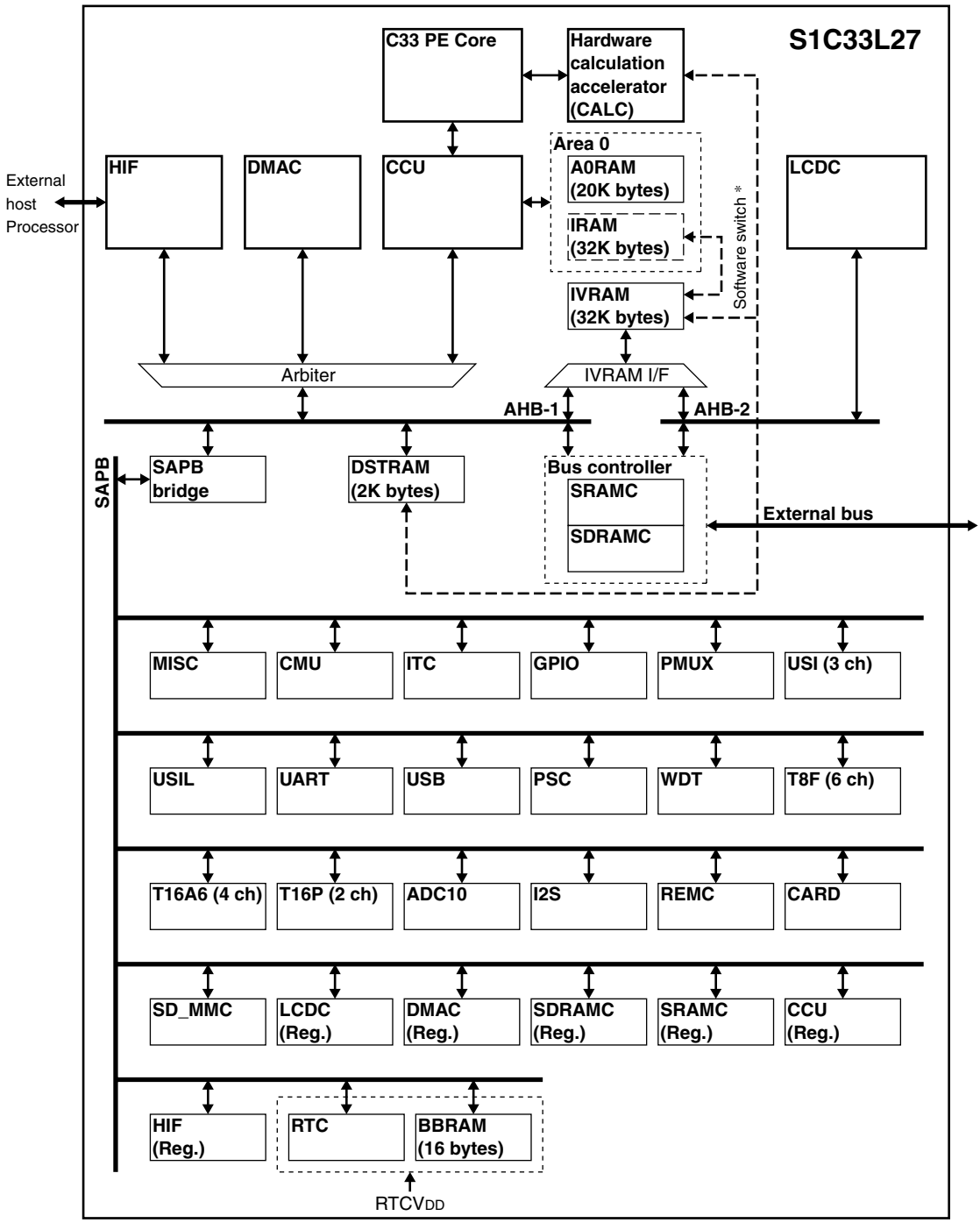
- During SLEEP: 2.3 µA typ. when RTC is running.
  1.0 µA typ. when RTC is not used.
- During HALT: 4.3 mA typ. when 48 MHz OSC3 clock is used as the system clock, all peripheral clocks = Off.
- During execution: 18 mA typ. when 48 MHz OSC3 clock is used as the system clock, CPU is running, all peripheral clocks = Off.

∗ Power consumption can be reduced by controlling the clocks through the clock management unit (CMU).

Shipping Form

- Die form: 200 pads (5.213 mm × 5.213 mm, pad pitch: 90 µm)
- Plastic package: TQFP15-128pin (14 mm × 14 mm × 1.0 mm, lead pitch: 0.4 mm)
  TQFP24-144pin (16 mm × 16 mm × 1.0 mm, lead pitch: 0.4 mm)
  QFP20-144pin (20 mm × 20 mm × 1.4 mm, lead pitch: 0.5 mm)
  PFBGA12U-180 (12 mm × 12 mm × 1.2 mm, ball pitch: 0.8 mm)

1.2 Block Diagram



* See the “Area 3 (IVRAM, DSTRAM)” section in the “Memory Map” chapter for details.

Figure 1.2.1 Block Diagram

1.3 Pin Descriptions

1.3.1 Pin Configuration Diagrams

The S1C33L27 comes in a TQFP15-128pin, TQFP24-144pin, QFP20-144pin, or PFBGA12U-180 package, or a die form.

TQFP15-128pin package

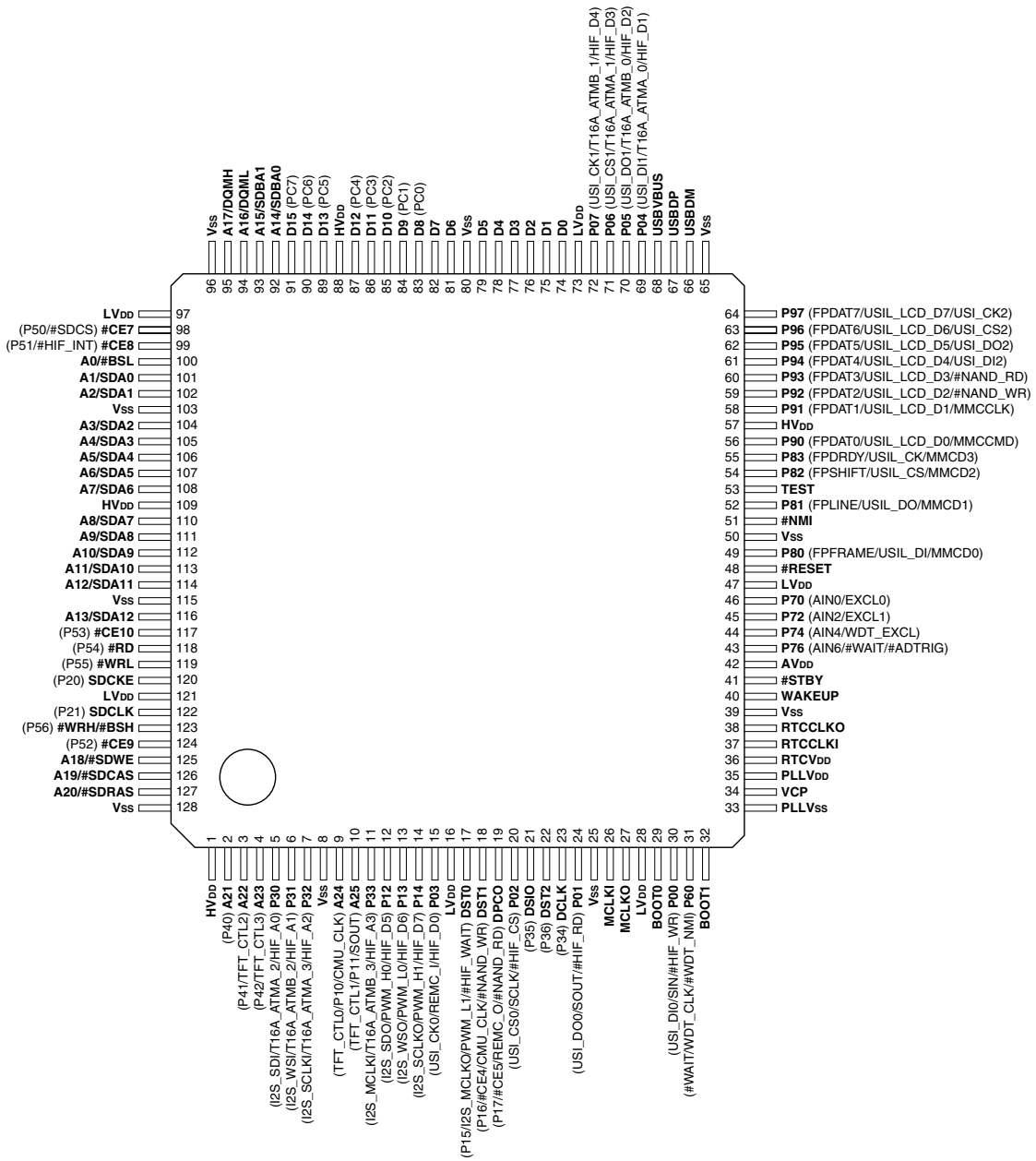


Figure 1.3.1.1 Pin Configuration Diagram (TQFP15-128pin)

TQFP24-144pin/QFP20-144pin package

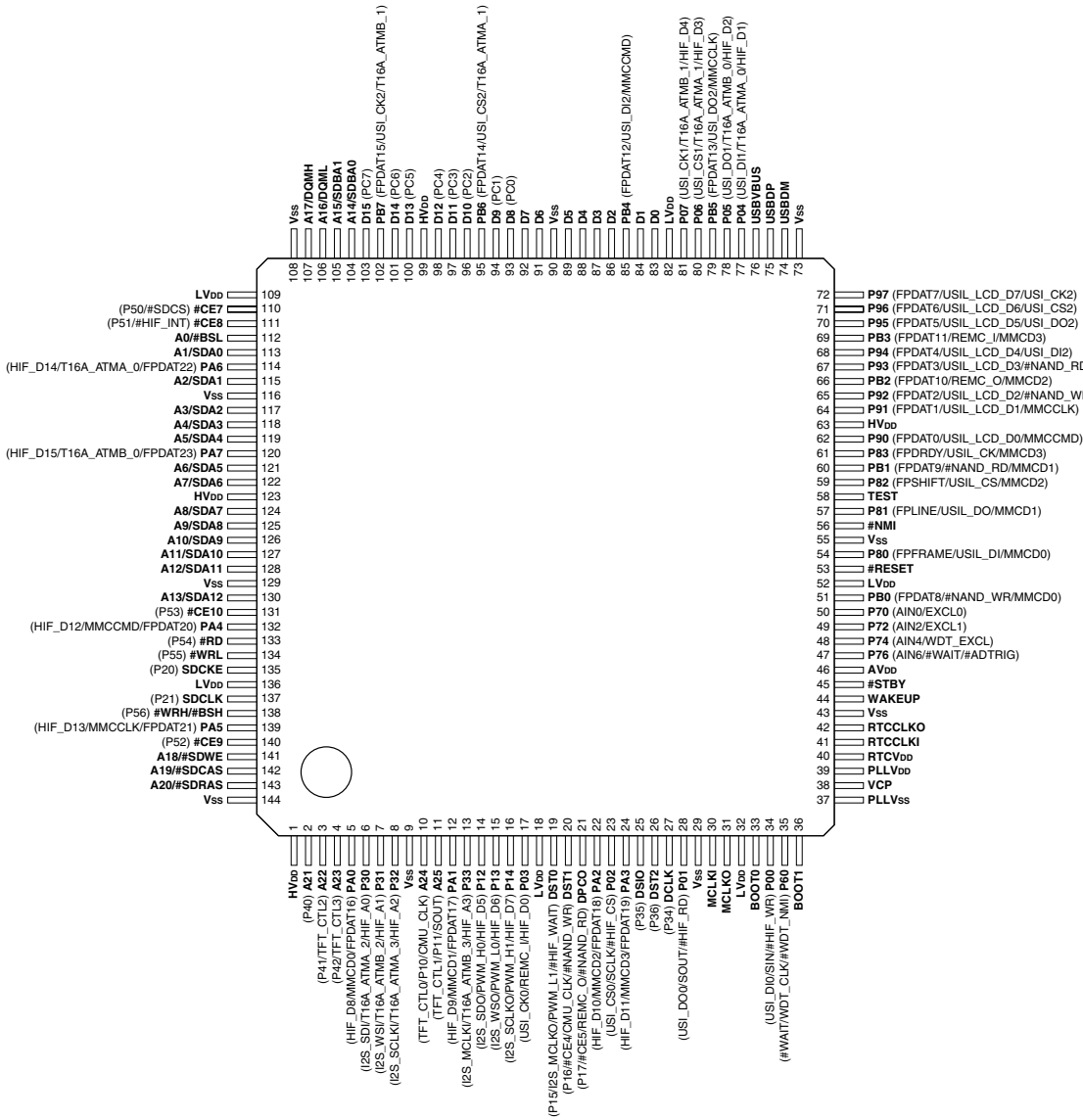
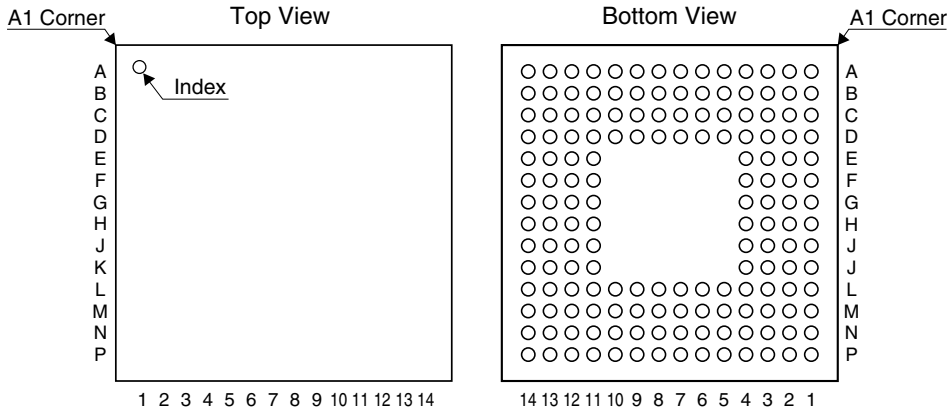


Figure 1.3.1.2 Pin Configuration Diagram (TQFP24-144pin/QFP20-144pin)

PFBGA12U-180 package

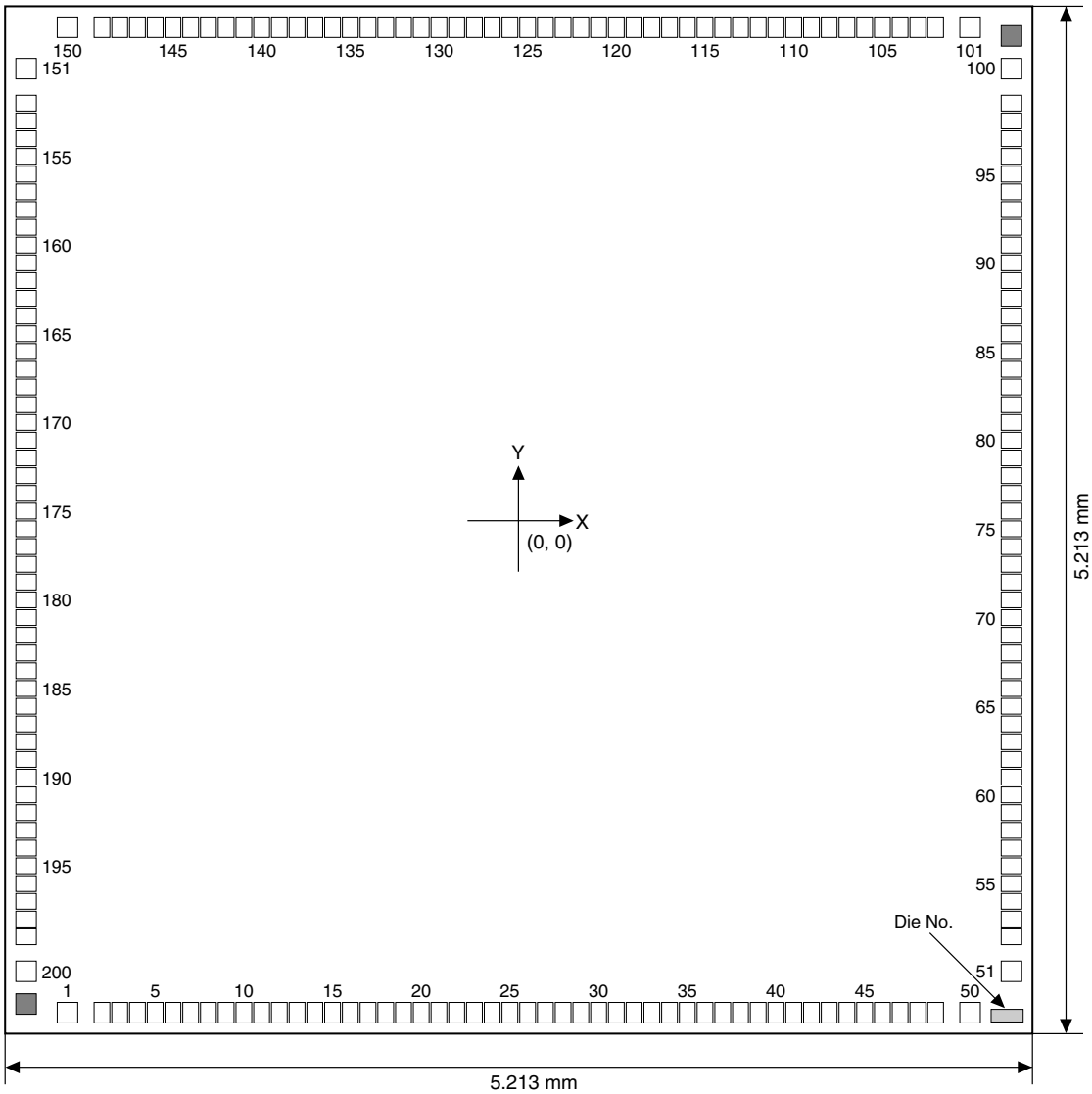


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
A	N.C.	**A20/ #SDRAS**	**A18/ #SDWE**	**SDCLK** P21	**SDCKE** P20	**A13/SDA12**	**A10/SDA9**	**A8/SDA7**	**A6/SDA5**	**A5/SDA4**	**A3/SDA2**	**A1/SDA0**	**#CE7** P50 #SDCS	N.C.	A
B	**A21** P40	**HVDD**	**A19/ #SDCAS**	**PA5** HIF_D13 MMCCLK FPDAT21	**#WRL** P55	**#CE10** P53	**A11/SDA10**	**A9/SDA8**	**A7/SDA6**	**A4/SDA3**	**A2/SDA1**	**A0/#BSL**	**A17/DQMH**	**A16/DQML**	B
C	**A22** P41 TFT_CTL2	**PD0** FPDAT16 USI_DI0 SIN	**PA0** HIF_D8 MMCD0 FPDAT16	**#CE9** P52	**PF1** I2S_WSO PWM_L0 TFT_CTL1	**PA4** HIF_D12 MMCCMD FPDAT20	**A12/SDA11**	**PF0** I2S_SDO PWM_H0 TFT_CTL0	**PF4** I2S_SDI #ADTRIG	**PA6** HIF_D14 T16A_ATMA_0 FPDAT22	**PF3** I2S_MCLKO PWM_L1 TFT_CTL3	**#CE8** P51 #HIF_INT	**A15/SDBA1**	**A14/SDBA0**	C
D	**A23** P42 TFT_CTL3	**P30** I2S_SDI T16A_ATMA_2 HIF_A0	**P31** I2S_WSI T16A_ATMB_2 HIF_A1	**PF2** I2S_SCLKO PWM_H1 TFT_CTL2	**#WRH/ #BSH** P56	**#RD** P54	**BOOT2**	**PA7** HIF_D15 T16A_ATMB_0 FPDAT23	**VSS**	**LVDD**	**PE5** MMCCLK T16A_ATMB_1 REMC_I	**PB7** FPDAT15 USI_CK2 T16A_ATMB_1	**D14** PC6	**D15** PC7	D
E	**A24** TFT_CTL0 P10 CMU_CLK	**PD1** FPDAT17 USI_DO0 SOUT	**P32** I2S_SCLKI T16A_ATMA_3 HIF_A2	**HVDD**	**VSS**	**LVDD**	**VSS**	**HVDD**	**VSS**	**HVDD**	**PE4** MMCCMD T16A_ATMA_1 REMC_O	**D13** PC5	**D12** PC4	**D11** PC3	E
F	**A25** TFT_CTL1 P11 SOUT	**PA1** HIF_D9 MMCD1 FPDAT17	**P33** I2S_MCLKI T16A_ATMB_3 HIF_A3	**P12** I2S_SDO PWM_H0 HIF_D5	**VSS**					**PB6** FPDAT14 USI_CS2 T16A_ATMA_1	**PE3** MMCD3 T16A_ATMB_3 USI_CK2	**D10** PC2	**D9** PC1	**D8** PC0	F
G	**P14** I2S_SCLKO PWM_H1 HIF_D7	**P03** USI_CK0 REMC_I HIF_D0	**P13** I2S_WSO PWM_L0 HIF_D6	**PD2** FPDAT18 USI_CS0 SCLK	**VSS**					**PB4** FPDAT12 USI_DI2 MMCCMD	**PE2** MMCD2 T16A_ATMA_3 USI_CS2	**D7**	**D6**	**D5**	G
H	**DST0** P15 I2S_MCLKO PWM_L1 #HIF_WAIT	**DST1** P16 #CE4 CMU_CLK #NAND_WR	**DPCO** P17 #CE5 REMC_O #NAND_RD	**PD3** FPDAT19 USI_CK0	**LVDD**					**LVDD**	**PE1** MMCD1 T16A_ATMB_2 USI_DO2	**D4**	**D2**	**D3**	H
J	**PA3** HIF_D11 MMCD3 FPDAT19	**PD4** FPDAT20 USI_DI1 #NAND_WR	**P02** USI_CS0 SCLK #HIF_CS	**PA2** HIF_D10 MMCD2 FPDAT18	**VSS**					**LVDD**	**P06** USI_CS1 T16A_ATMA_1 HIF_D3	**P07** USI_CK1 T16A_ATMB_1 HIF_D4	**D1**	**D0**	J
K	**DCLK** P34	**DST2** P36	**DSIO** P35	**P01** USI_DO0 SOUT #HIF_RD	**LVDD**	**HVDD**	**LVDD**	**LVDD**	**PB1** FPDAT9 #NAND_RD MMCD1	**HVDD**	**P93** FPDAT3 USIL_LCD_D3 #NAND_RD	**P04** USI_DI1 T16A_ATMA_0 HIF_D1	**P05** USI_DO1 T16A_ATMB_0 HIF_D2	**PB5** FPDAT13 USI_DO2 MMCCLK	K
L	**MCLKI**	**VSS**	**BOOT0**	**P00** USI_DI0 SIN #HIF_WR	**WAKEUP**	**#STBY**	**VSS**	**VSS**	**PD6** FPDAT22 USI_CS1 T16A_ATMA_0	**PB3** FPDRDY USIL_CK MMCD3	**PB2** FPDAT10 REMC_O MMCD2	**P96** FPDAT6 USIL_LCD_D6 USI_CS2	**PE0** MMCD0 T16A_ATMA_2 USI_DI2	**USBVBUS**	L
M	**MCLKO**	**PD5** FPDAT21 USI_DO1 #NAND_RD	**P60** #WAIT WDT_CLK #WDT_NMI	**VSS**	**AVDD**	**VSS**	**VSS**	**PB0** FPDAT8 #NAND_WR MMCD0	**P81** FPLINE USIL_DO MMCD1	**P82** FPSHIFT USIL_CS MMCD2	**P92** FPDAT2 USIL_LCD_D2 #NAND_WR	**P95** FPDAT5 USIL_LCD_D5 USI_DO2	**VSS**	**USBDP**	M
N	**BOOT1**	**PLLVSS**	**PLLVDD**	**RTCVDD**	**P76** AIN6 #WAIT #ADTRIG	**P74** AIN4 WDT_EXCL	**P72** AIN2 EXCL1	**P70** AIN0 EXCL0	**P80** FPFRAME USIL_DI MMCD0	**TEST**	**P91** FPDAT1 USIL_LCD_D1 MMCCLK	**P94** FPDAT4 USIL_LCD_D4 USI_DI2	**P97** FPDAT7 USIL_LCD_D7 USI_CK2	**USBDM**	N
P	N.C.	**VCP**	**RTCCLKI**	**RTCCLKO**	**P77** AIN7	**P75** AIN5 I2S_MCLKI	**P73** AIN3 I2S_SCLKI	**P71** AIN1 I2S_WSI	**#RESET**	**#NMI**	**P90** FPDAT0 USIL_LCD_D0 MMCCMD	**PD7** FPDAT23 USI_CK1 T16A_ATMB_0	**PB3** FPDAT11 REMC_I MMCD3	N.C.	P
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	

Top View

Figure 1.3.1.3 Pin Configuration Diagram (PFBGA12U-180)

Chip



Pad opening No. 1, 50, 51, 100, 101, 150, 151, 200: 104 × 104 µm
No. 2 to 49, 102 to 149: 80 × 104 µm
No. 52 to 99, 152 to 199: 104 × 80 µm
Chip thickness 400 µm

Figure 1.3.1.4 Pad Layout Diagram

Table 1.3.1.1 Pad Coordinates

No.	Pad name	X (µm)	Y (µm)	No.	Pad name	X (µm)	Y (µm)
1	**HVDD**	-2290	-2500	51	N.C.	2500	-2290
2	**HVDD**	-2115	-2500	52	**PLLVSS**	2500	-2115
3	**A21**/P40	-2025	-2500	53	**VCP**	2500	-2025
4	**PD0**/FPDAT16/USI_DI0/SIN	-1935	-2500	54	**PLLVDD**	2500	-1935
5	**A22**/P41/TFT_CTL2	-1845	-2500	55	N.C.	2500	-1845
6	**A23**/P42/TFT_CTL3	-1755	-2500	56	N.C.	2500	-1755
7	**PA0**/HIF_D8/MMCD0/FPDAT16	-1665	-2500	57	**RTCVDD**	2500	-1665
8	N.C.	-1575	-2500	58	**RTCCLKI**	2500	-1575
9	**P30**/I2S_SDI/T16A_ATMA_2/HIF_A0	-1485	-2500	59	**RTCCLKO**	2500	-1485
10	**P31**/I2S_WSI/T16A_ATMB_2/HIF_A1	-1395	-2500	60	**VSS**	2500	-1395
11	**P32**/I2S_SCLKI/T16A_ATMA_3/HIF_A2	-1305	-2500	61	**WAKEUP**	2500	-1305
12	**PD1**/FPDAT17/USI_DO0/SOUT	-1215	-2500	62	**#STBY**	2500	-1215
13	**VSS**	-1125	-2500	63	**LVDD**	2500	-1125
14	**VSS**	-1035	-2500	64	**AVDD**	2500	-1035
15	**A24**/TFT_CTL0/P10/CMU_CLK	-945	-2500	65	**P77**/AIN7	2500	-945
16	**A25**/TFT_CTL1/P11/SOUT	-855	-2500	66	**P76**/AIN6/#WAIT/#ADTRIG	2500	-855
17	**PA1**/HIF_D9/MMCD1/FPDAT17	-765	-2500	67	**P75**/AIN5/I2S_MCLKI	2500	-765
18	**P33**/I2S_MCLKI/T16A_ATMB_3/HIF_A3	-675	-2500	68	**P74**/AIN4/WDT_EXCL	2500	-675
19	**P12**/I2S_SDO/PWM_H0/HIF_D5	-585	-2500	69	**P73**/AIN3/I2S_SCLKI	2500	-585
20	**PD2**/FPDAT18/USI_CS0/SCLK	-495	-2500	70	**P72**/AIN2/EXCL1	2500	-495
21	**P13**/I2S_WSO/PWM_L0/HIF_D6	-405	-2500	71	**P71**/AIN1/I2S_WSI	2500	-405
22	**P14**/I2S_SCLKO/PWM_H1/HIF_D7	-315	-2500	72	**P70**/AIN0/EXCL0	2500	-315
23	**P03**/USI_CK0/REMC_I/HIF_D0	-225	-2500	73	**VSS**	2500	-225
24	**LVDD**	-135	-2500	74	**PB0**/FPDAT8/#NAND_WR/MMCD0	2500	-135
25	**LVDD**	-45	-2500	75	**LVDD**	2500	-45
26	**PD3**/FPDAT19/USI_CK0	45	-2500	76	**#RESET**	2500	45
27	**DST0**/P15/I2S_MCLKO/PWM_L1/#HIF_WAIT	135	-2500	77	**P80**/FPFRAME/USIL_DI/MMCD0	2500	135
28	**DST1**/P16/#CE4/CMU_CLK/#NAND_WR	225	-2500	78	**VSS**	2500	225
29	**DPCO**/P17/#CE5/REMC_O/#NAND_RD	315	-2500	79	**VSS**	2500	315
30	**PA2**/HIF_D10/MMCD2/FPDAT18	405	-2500	80	**#NMI**	2500	405
31	**P02**/USI_CS0/SCLK/#HIF_CS	495	-2500	81	**P81**/FPLINE/USIL_DO/MMCD1	2500	495
32	**PD4**/FPDAT20/USI_DI1/#NAND_WR	585	-2500	82	**PD6**/FPDAT22/USI_CS1/T16A_ATMA_0	2500	585
33	**PA3**/HIF_D11/MMCD3/FPDAT19	675	-2500	83	**TEST**	2500	675
34	**DSIO**/P35	765	-2500	84	**P82**/FPSHIFT/USIL_CS/MMCD2	2500	765
35	**DST2**/P36	855	-2500	85	**PB1**/FPDAT9/#NAND_RD/MMCD1	2500	855
36	**DCLK**/P34	945	-2500	86	**P83**/FPDRDY/USIL_CK/MMCD3	2500	945
37	**P01**/USI_DO0/SOUT/#HIF_RD	1035	-2500	87	**P90**/FPDAT0/USIL_LCD_D0/MMCCMD	2500	1035
38	**VSS**	1125	-2500	88	**HVDD**	2500	1125
39	**VSS**	1215	-2500	89	**HVDD**	2500	1215
40	**MCLKI**	1305	-2500	90	**P91**/FPDAT1/USIL_LCD_D1/MMCCLK	2500	1305
41	**MCLKO**	1395	-2500	91	**LVDD**	2500	1395
42	**LVDD**	1485	-2500	92	**P92**/FPDAT2/USIL_LCD_D2/#NAND_WR	2500	1485
43	**LVDD**	1575	-2500	93	**PB2**/FPDAT10/REMC_O/MMCD2	2500	1575
44	**BOOT0**	1665	-2500	94	**PD7**/FPDAT23/USI_CK1/T16A_ATMB_0	2500	1665
45	**P00**/USI_DI0/SIN/#HIF_WR	1755	-2500	95	**P93**/FPDAT3/USIL_LCD_D3/#NAND_RD	2500	1755
46	**HVDD**	1845	-2500	96	**P94**/FPDAT4/USIL_LCD_D4/USI_DI2	2500	1845
47	**HVDD**	1935	-2500	97	**PB3**/FPDAT11/REMC_I/MMCD3	2500	1935
48	**PD5**/FPDAT21/USI_DO1/#NAND_RD	2025	-2500	98	**P95**/FPDAT5/USIL_LCD_D5/USI_DO2	2500	2025
49	**P60**/#WAIT/WDT_CLK/#WDT_NMI	2115	-2500	99	**P96**/FPDAT6/USIL_LCD_D6/USI_CS2	2500	2115
50	**BOOT1**	2290	-2500	100	**P97**/FPDAT7/USIL_LCD_D7/USI_CK2	2500	2290

No.	Pad name	X (µm)	Y (µm)	No.	Pad name	X (µm)	Y (µm)
101	**VSS**	2290	2500	151	**LVDD**	-2500	2290
102	N.C.	2115	2500	152	**LVDD**	-2500	2115
103	N.C.	2025	2500	153	**#CE7**/P50/#SDCS	-2500	2025
104	**USBDM**	1935	2500	154	**#CE8**/P51/#HIF_INT	-2500	1935
105	**USBDP**	1845	2500	155	**PF3**/I2S_MCLKO/PWM_L1/TFT_CTL3	-2500	1845
106	N.C.	1755	2500	156	**A0/#BSL**	-2500	1755
107	**USBVBUS**	1665	2500	157	**A1/SDA0**	-2500	1665
108	**PE0**/MMCD0/T16A_ATMA_2/USI_DI2	1575	2500	158	**PA6**/HIF_D14/T16A_ATMA_0/FPDAT22	-2500	1575
109	**P04**/USI_DI1/T16A_ATMA_0/HIF_D1	1485	2500	159	**A2/SDA1**	-2500	1485
110	**P05**/USI_DO1/T16A_ATMB_0/HIF_D2	1395	2500	160	**VSS**	-2500	1395
111	**PB5**/FPDAT13/USI_DO2/MMCCLK	1305	2500	161	**VSS**	-2500	1305
112	**P06**/USI_CS1/T16A_ATMA_1/HIF_D3	1215	2500	162	**A3/SDA2**	-2500	1215
113	**P07**/USI_CK1/T16A_ATMB_1/HIF_D4	1125	2500	163	**PF4**/I2S_SDI/#ADTRIG	-2500	1125
114	**PE1**/MMCD1/T16A_ATMB_2/USI_DO2	1035	2500	164	**A4/SDA3**	-2500	1035
115	**LVDD**	945	2500	165	N.C.	-2500	945
116	**LVDD**	855	2500	166	**A5/SDA4**	-2500	855
117	**D0**	765	2500	167	**PA7**/HIF_D15/T16A_ATMB_0/FPDAT23	-2500	765
118	**D1**	675	2500	168	**A6/SDA5**	-2500	675
119	**PB4**/FPDAT12/USI_DI2/MMCCMD	585	2500	169	**A7/SDA6**	-2500	585
120	**D2**	495	2500	170	**HVDD**	-2500	495
121	**D3**	405	2500	171	**HVDD**	-2500	405
122	**PE2**/MMCD2/T16A_ATMA_3/USI_CS2	315	2500	172	**A8/SDA7**	-2500	315
123	**D4**	225	2500	173	**A9/SDA8**	-2500	225
124	**D5**	135	2500	174	**A10/SDA9**	-2500	135
125	**VSS**	45	2500	175	**PF0**/I2S_SDO/PWM_H0/TFT_CTL0	-2500	45
126	**VSS**	-45	2500	176	**A11/SDA10**	-2500	-45
127	**D6**	-135	2500	177	**A12/SDA11**	-2500	-135
128	**D7**	-225	2500	178	**VSS**	-2500	-225
129	**PE3**/MMCD3/T16A_ATMB_3/USI_CK2	-315	2500	179	**VSS**	-2500	-315
130	**D8**/PC0	-405	2500	180	**BOOT2**	-2500	-405
131	**D9**/PC1	-495	2500	181	**A13/SDA12**	-2500	-495
132	**PB6**/FPDAT14/USI_CS2/T16A_ATMA_1	-585	2500	182	**#CE10**/P53	-2500	-585
133	**D10**/PC2	-675	2500	183	**PA4**/HIF_D12/MMCCMD/FPDAT20	-2500	-675
134	**D11**/PC3	-765	2500	184	**#RD**/P54	-2500	-765
135	**PE4**/MMCCMD/T16A_ATMA_1/REMC_O	-855	2500	185	N.C.	-2500	-855
136	**D12**/PC4	-945	2500	186	**#WRL**/P55	-2500	-945
137	**HVDD**	-1035	2500	187	**PF1**/I2S_WSO/PWM_L0/TFT_CTL1	-2500	-1035
138	**HVDD**	-1125	2500	188	**SDCKE**/P20	-2500	-1125
139	**D13**/PC5	-1215	2500	189	**LVDD**	-2500	-1215
140	**D14**/PC6	-1305	2500	190	**LVDD**	-2500	-1305
141	**PB7**/FPDAT15/USI_CK2/T16A_ATMB_1	-1395	2500	191	**SDCLK**/P21	-2500	-1395
142	N.C.	-1485	2500	192	**#WRH/#BSH**/P56	-2500	-1485
143	**D15**/PC7	-1575	2500	193	**PA5**/HIF_D13/MMCCLK/FPDAT21	-2500	-1575
144	**A14/SDBA0**	-1665	2500	194	**#CE9**/P52	-2500	-1665
145	**PE5**/MMCCLK/T16A_ATMB_1/REMC_I	-1755	2500	195	**A18/#SDWE**	-2500	-1755
146	**A15/SDBA1**	-1845	2500	196	**PF2**/I2S_SCLKO/PWM_H1/TFT_CTL2	-2500	-1845
147	**A16/DQML**	-1935	2500	197	**A19/#SDCAS**	-2500	-1935
148	**A17/DQMH**	-2025	2500	198	**A20/#SDRAS**	-2500	-2025
149	**VSS**	-2115	2500	199	**VSS**	-2500	-2115
150	**VSS**	-2290	2500	200	**VSS**	-2500	-2290

1.3.2 Pin Functions

The tables below list the S1C33L27 pin functions.

Notes: • Pin name
- The # prefixed to pin names indicates that the pin inputs/outputs an active low signal.
- The pin names listed in boldface denote the default pin (signal) name.
  Each pin is assigned one to four functions and the function to be used must be selected using the corresponding port function select bit. See the "I/O Ports (GPIO)" chapter for more information on the pin function selections.

• I/O
- The I/O listed in boldface and uppercase denote the default input/output direction.
- (H), (L), and (X) denote the initial output level, High, Low, and Undefined, respectively.

• Pin No.
- Chip: Pad No.
- 128: TQFP15-128pin
- 144: TQFP24-144pin and QFP20-144pin
- 180: PFBGA12U-180

• PWR (power system)
- P1: LV_{DD}
- P2: HV_{DD}
- P3: $RTCV_{DD}$
- P4: AV_{DD}
- P5: $PLLV_{DD}$

• DC characteristics
- Output Type 1: I_{OH}/I_{OL} = 2 mA (HV_{DD} = 3.3 V)*
  Type 2: I_{OH}/I_{OL} = 4 mA (HV_{DD} = 3.3 V)*
  Type 3: I_{OH}/I_{OL} = 8 mA (HV_{DD} = 3.3 V)*
  * See "DC Characteristics" in the "Electrical Characteristics" chapter.
- PU/PD PU: Pull-up
  PD: Pull-down
  PUc: Pull-up with software control
  (en): Enabled by default
  (dis): Disabled by default

Table 1.3.2.1 List of Power Supply Pins

No.	Pin name	I/O	Description	Pin No.				Power voltage
				Chip	128	144	180	
1	**HV_{DD}**	–	I/O power supply pin	1, 2, 46, 47, 88, 89, 137, 138, 170, 171	1, 57, 88, 109	1, 63, 99, 123	B2, E4, E8, E10, K6, K10	3.3 V typ. (2.7 to 3.6 V) (3.0 to 3.6 V when the USB module is used.)
2	**AV_{DD}**	–	Analog power supply pin	64	42	46	M5	3.3 V typ. (= HV_{DD})
3	**LV_{DD}**	–	Internal logic/internal memory power supply pin	24, 25, 42, 43, 63, 75, 91, 115, 116, 151, 152, 189, 190	16, 28, 47, 73, 97, 121	18, 32, 52, 82, 109, 136	D10, E6, H5, H10, J10, K5, K7, K8	1.8 V typ. (1.65 to 1.95 V) (1.7 to 1.9 V when a ceramic resonator is used.)
4	**$RTCV_{DD}$**	–	RTC/BBRAM power supply pin	57	36	40	N4	1.8 V typ. (= LV_{DD})
5	**V_{SS}**	–	Ground pin	13, 14, 38, 39, 60, 73, 78, 79, 101, 125, 126, 149, 150, 160, 161, 178, 179, 199, 200	8, 25, 39, 50, 65, 80, 96, 103, 115, 128	9, 29, 43, 55, 73, 90, 108, 116, 129, 144	D9, E5, E7, E9, F5, G5, J5, L2, L7, L8, M4, M6, M7, M13	GND
6	**$PLLV_{DD}$**	–	PLL power supply pin	54	35	39	N3	1.8 V typ. (= LV_{DD})
7	**$PLLV_{SS}$**	–	PLL power supply ground pin	52	33	37	N2	GND (= V_{SS})

Table 1.3.2.2 List of Clock Pins

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
1	**MCLKI**	**I**	High speed (OSC3) oscillation input (crystal/ceramic resonator or external clock input)	40	26	30	L1	P1	–	–	–
2	**MCLKO**	**O**	High speed (OSC3) oscillation output (crystal/ceramic resonator)	41	27	31	M1	P1	–	–	–
3	**RTCCLKI**	**I**	RTC (OSC1) oscillation input (crystal resonator or external clock input)	58	37	41	P3	P3	–	–	–
4	**RTCCLKO**	**O**	RTC (OSC1) oscillation output (crystal resonator)	59	38	42	P4	P3	–	–	–

Table 1.3.2.3 List of External Bus Pins

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
1	**D0**	**I**/o	Data bus D0	117	74	83	J14	P2	LVTTL	Type 2	Bus hold latch
2	**D1**	**I**/o	Data bus D1	118	75	84	J13				
3	**D2**	**I**/o	Data bus D2	120	76	86	H13				
4	**D3**	**I**/o	Data bus D3	121	77	87	H14				
5	**D4**	**I**/o	Data bus D4	123	78	88	H12				
6	**D5**	**I**/o	Data bus D5	124	79	89	G14				
7	**D6**	**I**/o	Data bus D6	127	81	91	G13				
8	**D7**	**I**/o	Data bus D7	128	82	92	G12				
9	**D8**	**I**/o	Data bus D8 (default)	130	83	93	F14				
	PC0	i/o	I/O port								
10	**D9**	**I**/o	Data bus D9 (default)	131	84	94	F13				
	PC1	i/o	I/O port								
11	**D10**	**I**/o	Data bus D10 (default)	133	85	96	F12				
	PC2	i/o	I/O port								
12	**D11**	**I**/o	Data bus D11 (default)	134	86	97	E14				
	PC3	i/o	I/O port								
13	**D12**	**I**/o	Data bus D12 (default)	136	87	98	E13				
	PC4	i/o	I/O port								
14	**D13**	**I**/o	Data bus D13 (default)	139	89	100	E12				
	PC5	i/o	I/O port								
15	**D14**	**I**/o	Data bus D14 (default)	140	90	101	D13				
	PC6	i/o	I/O port								
16	**D15**	**I**/o	Data bus D15 (default)	143	91	103	D14				
	PC7	i/o	I/O port								
17	**A0/#BSL**	**O** (L)	Address bus A0 / Bus strobe (low byte) signal output	156	100	112	B12	P2	–	Type 2	–
18	**A1/SDA0**	**O** (L)	Address bus A1 / SDRAM address A0	157	101	113	A12				
19	**A2/SDA1**	**O** (L)	Address bus A2 / SDRAM address A1	159	102	115	B11				
20	**A3/SDA2**	**O** (L)	Address bus A3 / SDRAM address A2	162	104	117	A11				
21	**A4/SDA3**	**O** (L)	Address bus A4 / SDRAM address A3	164	105	118	B10				
22	**A5/SDA4**	**O** (L)	Address bus A5 / SDRAM address A4	166	106	119	A10				
23	**A6/SDA5**	**O** (L)	Address bus A6 / SDRAM address A5	168	107	121	A9				
24	**A7/SDA6**	**O** (L)	Address bus A7 / SDRAM address A6	169	108	122	B9				
25	**A8/SDA7**	**O** (L)	Address bus A8 / SDRAM address A7	172	110	124	A8				
26	**A9/SDA8**	**O** (L)	Address bus A9 / SDRAM address A8	173	111	125	B8				
27	**A10/SDA9**	**O** (L)	Address bus A10 / SDRAM address A9	174	112	126	A7				
28	**A11/SDA10**	**O** (L)	Address bus A11 / SDRAM address A10	176	113	127	B7				

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
29	**A12/SDA11**	**O** (L)	Address bus A12 / SDRAM address A11	177	114	128	C7	P2	–	Type 2	–
30	**A13/SDA12**	**O** (L)	Address bus A13 / SDRAM address A12	181	116	130	A6				
31	**A14/SDBA0**	**O** (L)	Address bus A14 / SDRAM bank address BA0	144	92	104	C14				
32	**A15/SDBA1**	**O** (L)	Address bus A15 / SDRAM bank address BA1	146	93	105	C13				
33	**A16/DQML**	**O** (H)	Address bus A16 / SDRAM data mask (low-order byte) signal output	147	94	106	B14				
34	**A17/DQMH**	**O** (H)	Address bus A17 / SDRAM data mask (high-order byte) signal output	148	95	107	B13				
35	**A18/#SDWE**	**O** (H)	Address bus A18 / SDRAM write signal output	195	125	141	A3				
36	**A19/#SDCAS**	**O** (H)	Address bus A19 / SDRAM column address strobe signal output	197	126	142	B3				
37	**A20/#SDRAS**	**O** (H)	Address bus A20 / SDRAM row address strobe signal output	198	127	143	A2				
38	**A21**	**O** (L)	Address bus A21 (default)	3	2	2	B1	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	P40	i/o	I/O port								
39	**A22**	**O** (L)	Address bus A22 (default)	5	3	3	C1				
	P41	i/o	I/O port								
	TFT_CTL2	o	LCDC TFT I/F control signal 2 output								
40	**A23**	**O** (L)	Address bus A23 (default)	6	4	4	D1				
	P42	i/o	I/O port								
	TFT_CTL3	o	LCDC TFT I/F control signal 3 output								
41	**A24**	**O** (L)	Address bus A24 (default)	15	9	10	E1				
	TFT_CTL0	o	LCDC TFT I/F control signal 0 output								
	P10	i/o	I/O port								
	CMU_CLK	o	CMU clock external output								
42	**A25**	**O** (L)	Address bus A25 (default)	16	10	11	F1				
	TFT_CTL1	o	LCDC TFT I/F control signal 1 output								
	P11	i/o	I/O port								
	SOUT	o	UART data output								
43	**#CE7**	**O** (H)	Area 7/19 chip enable signal output (default)	153	98	110	A13	P2	LVTTL Schmitt	Type 2	100k PUc (dis)
	P50	i/o	I/O port								
	#SDCS	o	SDRAM chip enable signal output								
44	**#CE8**	**O** (H)	Area 8/21 chip enable signal output (default)	154	99	111	C12	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	P51	i/o	I/O port								
	#HIF_INT	o	Host I/F interrupt signal output								
45	**#CE9**	**O** (H)	Area 9/22 chip enable signal output (default)	194	124	140	C4				
	P52	i/o	I/O port								
46	**#CE10**	**O** (H)	Area 10/13/20 chip enable signal output (default)	182	117	131	B6	P2	LVTTL Schmitt	Type 1	100k PUc (en/dis)*
	P53	i/o	I/O port								
47	**#RD**	**O** (H)	Read signal output (default)	184	118	133	D6	P2	LVTTL Schmitt	Type 2	100k PUc (dis)
	P54	i/o	I/O port								
48	**#WRL**	**O** (H)	Write (low-order byte) signal output (default)	186	119	134	B5				
	P55	i/o	I/O port								

* The #CE10 pull-up resistor is disabled by default. When using the #CE10 pin as an input pin for NAND Flash, SPI-EEPROM or PC RS232C boot mode, connect an external pull-down resistor to the #CE10 pin to set the pin level to 0 or an external pull-up resistor to set the pin level to 1.

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
49	**#WRH/#BSH**	**O** (H)	Write (high-order byte) signal / Bus strobe (high-order byte) signal output (default)	192	123	138	D5	P2	LVTTL Schmitt	Type 2	100k PUc (dis)
	P56	i/o	I/O port								
50	**SDCKE**	**O** (H)	SDRAM clock enable signal output (default)	188	120	135	A5	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	P20	i/o	I/O port								
51	**SDCLK**	**O** (L)	SDRAM clock output (default)	191	122	137	A4	P2	LVTTL Schmitt	Type 2	100k PUc (dis)
	P21	i/o	I/O port								

Table 1.3.2.4 List of I/O Port and Peripheral Function Pins

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
1	**P00**	**I**/o	I/O port (default)	45	30	34	L4	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	USI_DI0	i/o	USI Ch.0 data input/output (see Table 1.3.2.8.)								
	SIN	i	UART data input								
	#HIF_WR	i	Host I/F write signal input								
2	**P01**	**I**/o	I/O port (default)	37	24	28	K4				
	USI_DO0	o	USI Ch.0 data output (see Table 1.3.2.8.)								
	SOUT	o	UART data output								
	#HIF_RD	i	Host I/F read signal input								
3	**P02**	**I**/o	I/O port (default)	31	20	23	J3				
	USI_CS0	i/o	USI Ch.0 slave select input, data input/output (see Table 1.3.2.8.)								
	SCLK	i	UART clock input								
	#HIF_CS	i	Host I/F chip select signal input								
4	**P03**	**I**/o	I/O port (default)	23	15	17	G2				
	USI_CK0	i/o	USI Ch.0 clock input/output (see Table 1.3.2.8.)								
	REMC_I	i	REMC receive signal input								
	HIF_D0	i/o	Host I/F data bit 0								
5	**P04**	**I**/o	I/O port (default)	109	69	77	K12				
	USI_DI1	i/o	USI Ch.1 data input/output (see Table 1.3.2.8.)								
	T16A_ATMA_0	i/o	T16A6 Ch.0 capture A signal input/ compare A signal output								
	HIF_D1	i/o	Host I/F data bit 1								
6	**P05**	**I**/o	I/O port (default)	110	70	78	K13				
	USI_DO1	o	USI Ch.1 data output (see Table 1.3.2.8.)								
	T16A_ATMB_0	i/o	T16A6 Ch.0 capture B signal input/ compare B signal output								
	HIF_D2	i/o	Host I/F data bit 2								
7	**P06**	**I**/o	I/O port (default)	112	71	80	J11				
	USI_CS1	i/o	USI Ch.1 slave select input, data input/output (see Table 1.3.2.8.)								
	T16A_ATMA_1	i/o	T16A6 Ch.1 capture A signal input/ compare A signal output								
	HIF_D3	i/o	Host I/F data bit 3								
8	**P07**	**I**/o	I/O port (default)	113	72	81	J12				
	USI_CK1	i/o	USI Ch.1 clock input/output (see Table 1.3.2.8.)								
	T16A_ATMB_1	i/o	T16A6 Ch.1 capture B signal input/ compare B signal output								
	HIF_D4	i/o	Host I/F data bit 4								
9	**P12**	**I**/o	I/O port (default)	19	12	14	F4				
	I2S_SDO	o	I²S serial data output								
	PWM_H0	o	T16P Ch.0 PWM_H signal output								
	HIF_D5	i/o	Host I/F data bit 5								
10	**P13**	**I**/o	I/O port (default)	21	13	15	G3				
	I2S_WSO	o	I²S word select signal output								
	PWM_L0	o	T16P Ch.0 PWM_L signal output								
	HIF_D6	i/o	Host I/F data bit 6								
11	**P14**	**I**/o	I/O port (default)	22	14	16	G1				
	I2S_SCLKO	o	I²S serial bit clock output								
	PWM_H1	o	T16P Ch.1 PWM_H signal output								
	HIF_D7	i/o	Host I/F data bit 7								

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
12	**P30**	**I**/o	I/O port (default)	9	5	6	D2	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	I2S_SDI	i	I²S serial data input								
	T16A_ATMA_2	i/o	T16A6 Ch.2 capture A signal input/ compare A signal output								
	HIF_A0	i	Host I/F address bit 0								
13	**P31**	**I**/o	I/O port (default)	10	6	7	D3				
	I2S_WSI	i	I²S word select signal input								
	T16A_ATMB_2	i/o	T16A6 Ch.2 capture B signal input/ compare B signal output								
	HIF_A1	i	Host I/F address bit 1								
14	**P32**	**I**/o	I/O port (default)	11	7	8	E3				
	I2S_SCLKI	i	I²S serial bit clock input								
	T16A_ATMA_3	i/o	T16A6 Ch.3 capture A signal input/ compare A signal output								
	HIF_A2	i	Host I/F address bit 2								
15	**P33**	**I**/o	I/O port (default)	18	11	13	F3				
	I2S_MCLKI	i	I²S master clock input								
	T16A_ATMB_3	i/o	T16A6 Ch.3 capture B signal input/ compare B signal output								
	HIF_A3	i	Host I/F address bit 3								
16	**P60**	**I**/o	I/O port (default)	49	31	35	M3	P2	LVTTL Schmitt	Type 1	100k PUc (en)
	#WAIT	i	Wait cycle request input								
	WDT_CLK	o	Watchdog timer clock output								
	#WDT_NMI	o	Watchdog timer NMI signal output								
17	**P70**	**I**	Input port (default)	72	46	50	N8	P4	LVCMOS	–	100k PUc (dis)
	AIN0	i	ADC10 Ch.0 analog input						Analog		
	EXCL0	i	T16A6 Ch.0–1/T16P Ch.0 external clock input						LVCMOS		
18	**P71**	**I**	Input port (default)	71	–	–	P8	P4	LVCMOS	–	100k PUc (dis)
	AIN1	i	ADC10 Ch.1 analog input						Analog		
	I2S_WSI	i	I²S word select signal input						LVCMOS		
19	**P72**	**I**	Input port (default)	70	45	49	N7	P4	LVCMOS	–	100k PUc (dis)
	AIN2	i	ADC10 Ch.2 analog input						Analog		
	EXCL1	i	T16A6 Ch.2–3/T16P Ch.1 external clock input						LVCMOS		
20	**P73**	**I**	Input port (default)	69	–	–	P7	P4	LVCMOS	–	100k PUc (dis)
	AIN3	i	ADC10 Ch.3 analog input						Analog		
	I2S_SCLKI	i	I²S serial bit clock input						LVCMOS		
21	**P74**	**I**	Input port (default)	68	44	48	N6	P4	LVCMOS	–	100k PUc (dis)
	AIN4	i	ADC10 Ch.4 analog input						Analog		
	WDT_EXCL	i	WDT external clock input						LVCMOS		
22	**P75**	**I**	Input port (default)	67	–	–	P6	P4	LVCMOS	–	100k PUc (dis)
	AIN5	i	ADC10 Ch.5 analog input						Analog		
	I2S_MCLKI	i	I²S master clock input						LVCMOS		
23	**P76**	**I**	Input port (default)	66	43	47	N5	P4	LVCMOS	–	100k PUc (en)
	AIN6	i	ADC10 Ch.6 analog input						Analog		
	#WAIT	i	Wait cycle request input						LVCMOS		
	#ADTRIG	i	ADC10 trigger input						LVCMOS		
24	**P77**	**I**	Input port (default)	65	–	–	P5	P4	LVCMOS	–	100k PUc (dis)
	AIN7	i	ADC10 Ch.7 analog input						Analog		
25	**P80**	**I**/o	I/O port (default)	77	49	54	N9	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	FPFRAME	o	LCD frame clock output								
	USIL_DI	i/o	USIL data input/output or LCD control signal output (see Table 1.3.2.9.)								
	MMCD0	i/o	SD/MMC I/F data bit 0								
26	**P81**	**I**/o	I/O port (default)	81	52	57	M9				
	FPLINE	o	LCD line clock output								
	USIL_DO	o	USIL data output or LCD control signal output (see Table 1.3.2.9.)								
	MMCD1	i/o	SD/MMC I/F data bit 1								
27	**P82**	**I**/o	I/O port (default)	84	54	59	M10				
	FPSHIFT	o	LCD shift clock output								
	USIL_CS	i/o	USIL slave select input, data input/output, or LCD control signal output (see Table 1.3.2.9.)								
	MMCD2	i/o	SD/MMC I/F data bit 2								

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
28	**P83**	I/o	I/O port (default)	86	55	61	L10	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	FPDRDY	o	LCD DRDY/MOD signal output								
	USIL_CK	i/o	USIL clock input/output or LCD control signal output (see Table 1.3.2.9.)								
	MMCD3	i/o	SD/MMC I/F data bit 3								
29	**P90**	I/o	I/O port (default)	87	56	62	P11				
	FPDAT0	o	LCD data bit 0								
	USIL_LCD_D0	i/o	USIL LCD data bit 0								
	MMCCMD	i/o	SD/MMC I/F command input/output								
30	**P91**	I/o	I/O port (default)	90	58	64	N11				
	FPDAT1	o	LCD data bit 1								
	USIL_LCD_D1	i/o	USIL LCD data bit 1								
	MMCCLK	o	SD/MMC I/F clock output								
31	**P92**	I/o	I/O port (default)	92	59	65	M11				
	FPDAT2	o	LCD data bit 2								
	USIL_LCD_D2	i/o	USIL LCD data bit 2								
	#NAND_WR	o	NAND Flash write signal output								
32	**P93**	I/o	I/O port (default)	95	60	67	K11				
	FPDAT3	o	LCD data bit 3								
	USIL_LCD_D3	i/o	USIL LCD data bit 3								
	#NAND_RD	o	NAND Flash read signal output								
33	**P94**	I/o	I/O port (default)	96	61	68	N12				
	FPDAT4	o	LCD data bit 4								
	USIL_LCD_D4	i/o	USIL LCD data bit 4								
	USI_DI2	i/o	USI Ch.2 data input/output (see Table 1.3.2.8.)								
34	**P95**	I/o	I/O port (default)	98	62	70	M12				
	FPDAT5	o	LCD data bit 5								
	USIL_LCD_D5	i/o	USIL LCD data bit 5								
	USI_DO2	o	USI Ch.2 data output (see Table 1.3.2.8.)								
35	**P96**	I/o	I/O port (default)	99	63	71	L12				
	FPDAT6	o	LCD data bit 6								
	USIL_LCD_D6	i/o	USIL LCD data bit 6								
	USI_CS2	i/o	USI Ch.2 slave select input, data input/output (see Table 1.3.2.8.)								
36	**P97**	I/o	I/O port (default)	100	64	72	N13				
	FPDAT7	o	LCD data bit 7								
	USIL_LCD_D7	i/o	USIL LCD data bit 7								
	USI_CK2	i/o	USI Ch.2 clock input/output (see Table 1.3.2.8.)								
37	**PA0**	I/o	I/O port (default)	7	–	5	C3				
	HIF_D8	i/o	Host I/F data bit 8								
	MMCD0	i/o	SD/MMC I/F data bit 0								
	FPDAT16	o	LCD data bit 16								
38	**PA1**	I/o	I/O port (default)	17	–	12	F2				
	HIF_D9	i/o	Host I/F data bit 9								
	MMCD1	i/o	SD/MMC I/F data bit 1								
	FPDAT17	o	LCD data bit 17								
39	**PA2**	I/o	I/O port (default)	30	–	22	J4				
	HIF_D10	i/o	Host I/F data bit 10								
	MMCD2	i/o	SD/MMC I/F data bit 2								
	FPDAT18	o	LCD data bit 18								
40	**PA3**	I/o	I/O port (default)	33	–	24	J1				
	HIF_D11	i/o	Host I/F data bit 11								
	MMCD3	i/o	SD/MMC I/F data bit 3								
	FPDAT19	o	LCD data bit 19								
41	**PA4**	I/o	I/O port (default)	183	–	132	C6				
	HIF_D12	i/o	Host I/F data bit 12								
	MMCCMD	i/o	SD/MMC I/F command input/output								
	FPDAT20	o	LCD data bit 20								
42	**PA5**	I/o	I/O port (default)	193	–	139	B4				
	HIF_D13	i/o	Host I/F data bit 13								
	MMCCLK	o	SD/MMC I/F clock output								
	FPDAT21	o	LCD data bit 21								

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
43	**PA6**	**I**/o	I/O port (default)	158	–	114	C10	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	HIF_D14	i/o	Host I/F data bit 14								
	T16A_ATMA_0	i/o	T16A6 Ch.0 capture A signal input/ compare A signal output								
	FPDAT22	o	LCD data bit 22								
44	**PA7**	**I**/o	I/O port (default)	167	–	120	D8				
	HIF_D15	i/o	Host I/F data bit 15								
	T16A_ATMB_0	i/o	T16A6 Ch.0 capture B signal input/ compare B signal output								
	FPDAT23	o	LCD data bit 23								
45	**PB0**	**I**/o	I/O port (default)	74	–	51	M8				
	FPDAT8	o	LCD data bit 8								
	#NAND_WR	o	NAND Flash write signal output								
	MMCD0	i/o	SD/MMC I/F data bit 0								
46	**PB1**	**I**/o	I/O port (default)	85	–	60	K9				
	FPDAT9	o	LCD data bit 9								
	#NAND_RD	o	NAND Flash read signal output								
	MMCD1	i/o	SD/MMC I/F data bit 1								
47	**PB2**	**I**/o	I/O port (default)	93	–	66	L11	P2	LVTTL Schmitt	Type 3	100k PUc (dis)
	FPDAT10	o	LCD data bit 10								
	REMC_O	o	REMC transmit signal output								
	MMCD2	i/o	SD/MMC I/F data bit 2								
48	**PB3**	**I**/o	I/O port (default)	97	–	69	P13	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	FPDAT11	o	LCD data bit 11								
	REMC_I	i	REMC receive signal input								
	MMCD3	i/o	SD/MMC I/F data bit 3								
49	**PB4**	**I**/o	I/O port (default)	119	–	85	G10				
	FPDAT12	o	LCD data bit 12								
	USI_DI2	i/o	USI Ch.2 data input/output (see Table 1.3.2.8.)								
	MMCCMD	i/o	SD/MMC I/F command input/output								
50	**PB5**	**I**/o	I/O port (default)	111	–	79	K14				
	FPDAT13	o	LCD data bit 13								
	USI_DO2	o	USI Ch.2 data output (see Table 1.3.2.8.)								
	MMCCLK	o	SD/MMC I/F clock output								
51	**PB6**	**I**/o	I/O port (default)	132	–	95	F10				
	FPDAT14	o	LCD data bit 14								
	USI_CS2	i/o	USI Ch.2 slave select input, data input/output (see Table 1.3.2.8.)								
	T16A_ATMA_1	i/o	T16A6 Ch.1 capture A signal input/ compare A signal output								
52	**PB7**	**I**/o	I/O port (default)	141	–	102	D12				
	FPDAT15	o	LCD data bit 15								
	USI_CK2	i/o	USI Ch.2 clock input/output (see Table 1.3.2.8.)								
	T16A_ATMB_1	i/o	T16A6 Ch.1 capture B signal input/ compare B signal output								
53	**PD0**	**I**/o	I/O port (default)	4	–	–	C2				
	FPDAT16	o	LCD data bit 16								
	USI_DI0	i/o	USI Ch.0 data input/output (see Table 1.3.2.8.)								
	SIN	i	UART data input								
54	**PD1**	**I**/o	I/O port (default)	12	–	–	E2				
	FPDAT17	o	LCD data bit 17								
	USI_DO0	o	USI Ch.0 data output (see Table 1.3.2.8.)								
	SOUT	o	UART data output								
55	**PD2**	**I**/o	I/O port (default)	20	–	–	G4				
	FPDAT18	o	LCD data bit 18								
	USI_CS0	i/o	USI Ch.0 slave select input, data input/output (see Table 1.3.2.8.)								
	SCLK	i	UART clock input								
56	**PD3**	**I**/o	I/O port (default)	26	–	–	H4				
	FPDAT19	o	LCD data bit 19								
	USI_CK0	i/o	USI Ch.0 clock input/output (see Table 1.3.2.8.)								

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
57	**PD4**	**I**/o	I/O port (default)	32	–	–	J2	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	FPDAT20	o	LCD data bit 20								
	USI_DI1	i/o	USI Ch.1 data input/output (see Table 1.3.2.8.)								
	#NAND_WR	o	NAND Flash write signal output								
58	**PD5**	**I**/o	I/O port (default)	48	–	–	M2				
	FPDAT21	o	LCD data bit 21								
	USI_DO1	o	USI Ch.1 data output (see Table 1.3.2.8.)								
	#NAND_RD	o	NAND Flash read signal output								
59	**PD6**	**I**/o	I/O port (default)	82	–	–	L9				
	FPDAT22	o	LCD data bit 22								
	USI_CS1	i/o	USI Ch.1 slave select input, data input/output (see Table 1.3.2.8.)								
	T16A_ATMA_0	i/o	T16A6 Ch.0 capture A signal input/ compare A signal output								
60	**PD7**	**I**/o	I/O port (default)	94	–	–	P12				
	FPDAT23	o	LCD data bit 23								
	USI_CK1	i/o	USI Ch.1 clock input/output (see Table 1.3.2.8.)								
	T16A_ATMB_0	i/o	T16A6 Ch.0 capture B signal input/ compare B signal output								
61	**PE0**	**I**/o	I/O port (default)	108	–	–	L13				
	MMCD0	i/o	SD/MMC I/F data bit 0								
	T16A_ATMA_2	i/o	T16A6 Ch.2 capture A signal input/ compare A signal output								
	USI_DI2	i/o	USI Ch.2 data input/output (see Table 1.3.2.8.)								
62	**PE1**	**I**/o	I/O port (default)	114	–	–	H11				
	MMCD1	i/o	SD/MMC I/F data bit 1								
	T16A_ATMB_2	i/o	T16A6 Ch.2 capture B signal input/ compare B signal output								
	USI_DO2	o	USI Ch.2 data output (see Table 1.3.2.8.)								
63	**PE2**	**I**/o	I/O port (default)	122	–	–	G11				
	MMCD2	i/o	SD/MMC I/F data bit 2								
	T16A_ATMA_3	i/o	T16A6 Ch.3 capture A signal input/ compare A signal output								
	USI_CS2	i/o	USI Ch.2 slave select input, data input/output (see Table 1.3.2.8.)								
64	**PE3**	**I**/o	I/O port (default)	129	–	–	F11				
	MMCD3	i/o	SD/MMC I/F data bit 3								
	T16A_ATMB_3	i/o	T16A6 Ch.3 capture B signal input/ compare B signal output								
	USI_CK2	i/o	USI Ch.2 clock input/output (see Table 1.3.2.8.)								
65	**PE4**	**I**/o	I/O port (default)	135	–	–	E11	P2	LVTTL Schmitt	Type 3	100k PUc (dis)
	MMCCMD	i/o	SD/MMC I/F command input/output								
	T16A_ATMA_1	i/o	T16A6 Ch.1 capture A signal input/ compare A signal output								
	REMC_O	o	REMC transmit signal output								
66	**PE5**	**I**/o	I/O port (default)	145	–	–	D11	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	MMCCLK	o	SD/MMC I/F clock output								
	T16A_ATMB_1	i/o	T16A6 Ch.1 capture B signal input/ compare B signal output								
	REMC_I	i	REMC receive signal input								
67	**PF0**	**I**/o	I/O port (default)	175	–	–	C8				
	I2S_SDO	o	I²S serial data output								
	PWM_H0	o	T16P Ch.0 PWM_H signal output								
	TFT_CTL0	o	LCDC TFT I/F control signal 0 output								
68	**PF1**	**I**/o	I/O port (default)	187	–	–	C5				
	I2S_WSO	o	I²S word select signal output								
	PWM_L0	o	T16P Ch.0 PWM_L signal output								
	TFT_CTL1	o	LCDC TFT I/F control signal 1 output								
69	**PF2**	**I**/o	I/O port (default)	196	–	–	D4				
	I2S_SCLKO	o	I²S serial bit clock output								
	PWM_H1	o	T16P Ch.1 PWM_H signal output								
	TFT_CTL2	o	LCDC TFT I/F control signal 2 output								

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
70	**PF3**	**I**/o	I/O port (default)	155	–	–	C11	P2	LVTTL Schmitt	Type 1	100k PUc (dis)
	I2S_MCLKO	o	I²S master clock output								
	PWM_L1	o	T16P Ch.1 PWM_L signal output								
	TFT_CTL3	o	LCDC TFT I/F control signal 3 output								
71	**PF4**	**I**/o	I/O port (default)	163	–	–	C9				
	I2S_SDI	i	I²S serial data input								
	#ADTRIG	i	ADC10 trigger input								

Table 1.3.2.5 List of USB Pins

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
1	**USBDP**	**I**/o	USB D+ pin	105	67	75	M14	P2	–	–	–
2	**USBDM**	**I**/o	USB D- pin	104	66	74	N14	P2	–	–	–
3	**USBVBUS**	**I**	USB VBUS pin. Allows input of 5 V	107	68	76	L14	P2	–	–	–

Table 1.3.2.6 List of Debug Control Pins

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
1	**DCLK**	i/**O** (H)	DCLK signal output for debugging (default)	36	23	27	K1	P2	LVTTL Schmitt	Type 2	50k PUc (dis)
	P34	i/o	I/O port								
2	**DSIO**	**I**/o	Serial input/output for debugging (default)	34	21	25	K3	P2	LVTTL Schmitt	Type 2	50k PUc (en)
	P35	i/o	I/O port								
3	**DST2**	i/**O** (L)	DST2 signal output for debugging (default)	35	22	26	K2	P2	LVTTL Schmitt	Type 2	100k PUc (dis)
	P36	i/o	I/O port								
4	**DST1**	i/**O** (H)	DST1 signal output for debugging (default)	28	18	20	H2				
	P16	i/o	I/O port								
	#CE4	o	Area 4/14 chip enable signal output								
	CMU_CLK	o	CMU clock external output								
	#NAND_WR	o	NAND Flash write signal output								
5	**DST0**	i/**O** (H)	DST0 signal output for debugging (default)	27	17	19	H1				
	P15	i/o	I/O port								
	I2S_MCLKO	o	I²S master clock output								
	PWM_L1	o	T16P Ch.1 PWM_L signal output								
	#HIF_WAIT	o	Host I/F wait signal output								
6	**DPCO**	i/**O** (H)	DPCO signal output for debugging (default)	29	19	21	H3	P2	LVTTL Schmitt	Type 3	100k PUc (dis)
	P17	i/o	I/O port								
	#CE5	o	Area 5/15/16 chip enable signal output								
	REMC_O	o	REMC transmit signal output								
	#NAND_RD	o	NAND Flash read signal output								

Table 1.3.2.7 List of Other Pins

No.	Pin name	I/O	Description	Pin No.				PWR	DC characteristics		
				Chip	128	144	180		Input	Output	PU/PD
1	**#RESET**	**I**	Reset input	76	48	53	P9	P2	LVTTL Schmitt	–	50k PU
2	**#NMI**	**I**	NMI request input	80	51	56	P10				
3	**BOOT0**	**I**	Boot mode select signal 0 input	44	29	33	L3	P2	LVTTL Schmitt	–	–
4	**BOOT1**	**I**	Boot mode select signal 1 input	50	32	36	N1				
5	**BOOT2**	**I**	Boot mode select signal 2 input	180	–	–	D7				
6	**WAKEUP**	**O** (X)	C33 wakeup signal output from RTC	61	40	44	L5	P3	–	1 mA	–
7	**#STBY**	**I**	C33 standby input (except for RTC)	62	41	45	L6	P3	LVTTL Schmitt	–	–
8	**TEST**	**I**	Test input. Connect to V_{SS} in user mode.	83	53	58	N10	P1	LVCMOS	–	–
9	**VCP**	**O** (H)	Test output	53	34	38	P2	P5	–	–	–

Table 1.3.2.8 USI Pin Configuration

Pin name	Function by interface mode				
	UART	SPI master	SPI slave	I²C master	I²C slave
USI_DI*x*	Data input (uart_rx)	Data input (spi_di)	Data input (spi_di)	Data input/output (i2c_sda)	Data input/output (i2c_sda)
USI_DO*x*	Data output (uart_tx)	Data output (spi_do)	Data output (spi_do) when #spi_ss = 0 Hi-Z when #spi_ss = 1	Unused (output)	Unused (output)
USI_CK*x*	Unused (input)	Clock output (spi_ck)	Clock input (spi_ck)	Clock input/output (i2c_sck)	Clock input/output (i2c_sck)
USI_CS*x*	Unused (input)	Unused (input)	Slave select input (#spi_ss)	Data input/output (i2c_sda)	Data input/output (i2c_sda)

(*x*: channel number, 0 to 2)

Note: In I²C mode, both the USI_DI*x* and USI_CS*x* pins can be configured as I²C data input/output pins. However, they can not be used for I²C data input/output at the same time.

Table 1.3.2.9 USIL Pin Configuration

Pin name	Function by interface mode						
	UART	SPI master	SPI slave	I²C master	I²C slave	LCD Parallel	LCD SPI
USIL_DI	Data input (uart_rx)	Data input (spi_di)	Data input (spi_di)	Data input/ output (i2c_sda)	Data input/ output (i2c_sda)	Address output (lcdp_a0)	Address output (lcds_a0)
USIL_DO	Data output (uart_tx)	Data output (spi_do)	Data output (spi_do) when #spi_ss = 0 Hi-Z when #spi_ss = 1	Unused (output)	Unused (output)	Write signal output (lcdp_wr)	Data output (lcds_do)
USIL_CK	Unused (input)	Clock output (spi_ck)	Clock input (spi_ck)	Clock input/ output (i2c_sck)	Clock input/ output (i2c_sck)	Read signal output (lcdp_rd)	Clock output (lcds_ck)
USIL_CS	Unused (input)	Unused (input)	Slave select input (#spi_ss)	Data input/ output (i2c_sda)	Data input/ output (i2c_sda)	Chip select signal output (lcdp_cs)	Unused (input)
USIL_LCD _D[7:0]	Unused (input)	Unused (input)	Unused (input)	Unused (input)	Unused (input)	8-bit data input/output	Unused (input)

Note: In I²C mode, both the USIL_DI and USIL_CS pins can be configured as I²C data input/output pins. However, they can not be used for I²C data input/output at the same time.

1.3.3 Package

TQFP15-128pin Package

(Unit: mm)

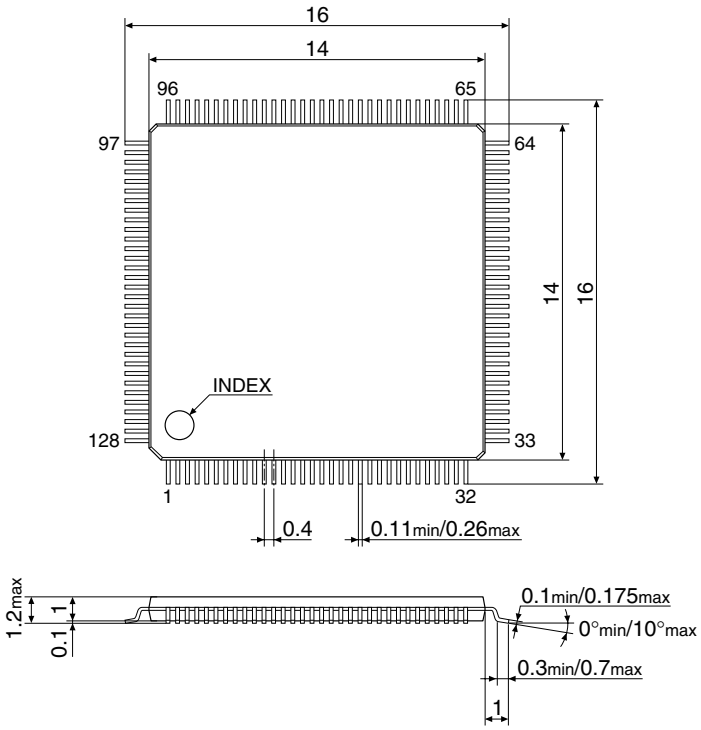


Figure 1.3.3.1 TQFP15-128pin Package Dimensions

TQFP24-144pin Package

(Unit: mm)

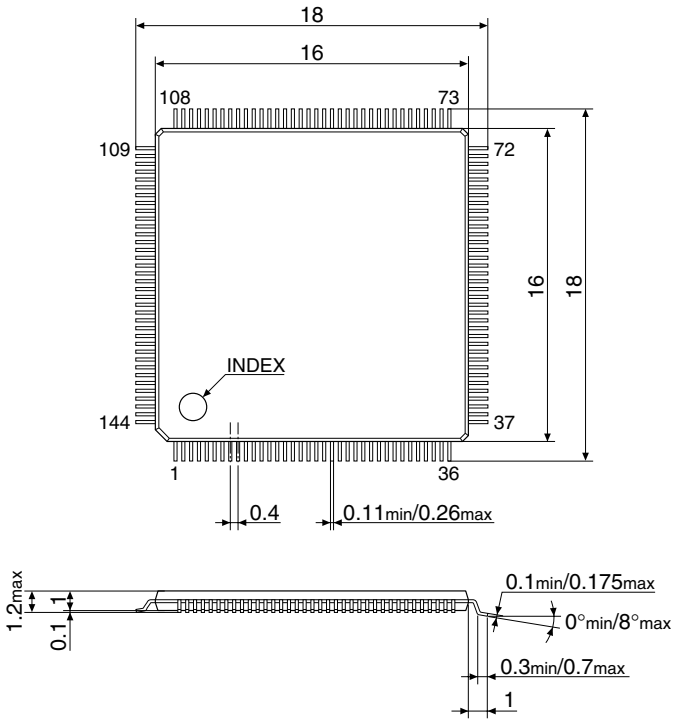


Figure 1.3.3.2 TQFP24-144pin Package Dimensions

QFP20-144pin Package

(Unit: mm)

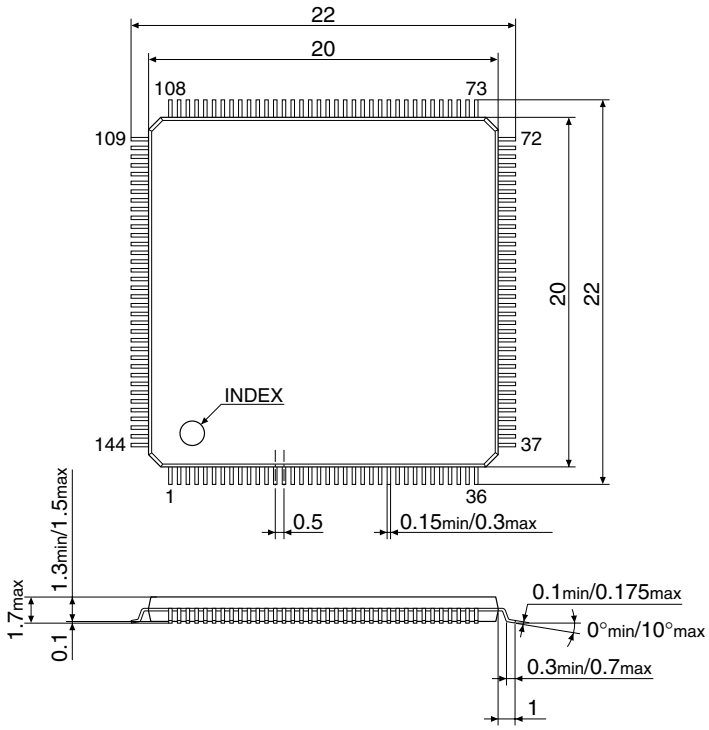
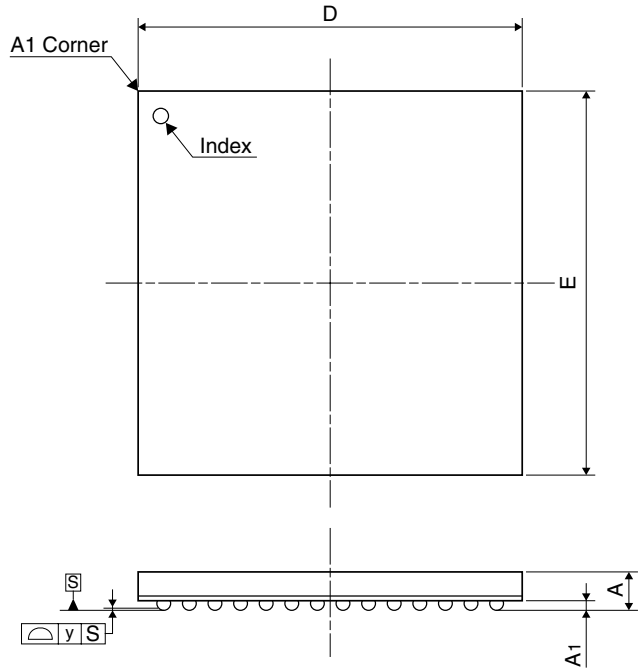


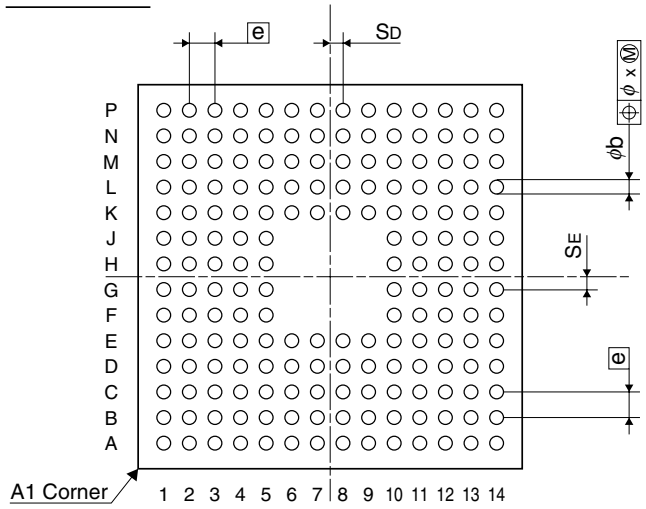
Figure 1.3.3.3 QFP20-144pin Package Dimensions

PFBGA12U-180 Package

Top View



Bottom View



Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	–	12	–
E	–	12	–
A	–	–	1.2
A_1	–	0.3	–
e	–	0.8	–
b	0.38	–	0.48
x	–	–	0.08
y	–	–	0.1
S_D	–	0.4	–
S_E	–	0.4	–

Figure 1.3.3.4 PFBGA12U-180 Package Dimensions

1.3.4 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 when the USB function controller is not used, or 100°C or less while the USB function controller is operating.

Thermal resistance of the TQFP/QFP package

1. When mounted on a board (windless condition)

Thermal resistance (θj-a) = 39°C/W (TQFP24), 42°C/W (TQFP15), 33.3°C/W (QFP20)

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.

Thermal resistance of the PFBGA package

1. When mounted on a board (windless condition)

Thermal resistance (θj-a) = 24°C/W (PFBGA12U)

This value indicates the thermal resistance of the package when measured under a windless condition, with the sample mounted on a measurement board (size: 114.5 × 101.5 × 1.6 mm thick, FR4/4 layered board).

2. When suspended alone (windless condition)

Thermal resistance = 165°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.

2 CPU

The S1C33L27 contains the C33 PE Core as its core processor.
The C33 PE (Processor Element) Core is a Seiko Epson original 32-bit RISC-type core processor for the S1C33 Family microprocessors.
For details of the C33 PE Core, refer to the "S1C33 Family C33 PE Core Manual."

2.1 Features of the C33 PE Core

Processor type

- Seiko Epson original 32-bit RISC processor
- 32-bit internal data processing
- Contains a 32-bit × 8-bit multiplier

Operating-clock frequency

- Depends on the processor model and process technology.

Instruction set

- Code length 16-bit fixed length
- Number of instructions 128
- Execution cycle Main instructions executed in one cycle
- Extended immediate instructions Immediate extended up to 32 bits
- Multiplication instructions Multiplications for 16 × 16 and 32 × 32 bits supported

Register set

- 32-bit general-purpose registers
- 32-bit special registers

Memory space and external bus

- Instruction, data, and I/O coexisting linear space
- Up to 4G bytes of memory space
- Harvard architecture using separated instruction bus and data bus

Interrupts

- Reset, NMI, and 240 external interrupts supported
- Four software exceptions
- Three instruction execution exceptions
- Direct branching from vector table to interrupt handler routine

Power-down mode

- HALT mode
- SLEEP mode

Coprocessor interface

- Hardware calculation accelerator

2.2 CPU Registers

The C33 PE Core contains 16 general-purpose registers and 8 special registers.

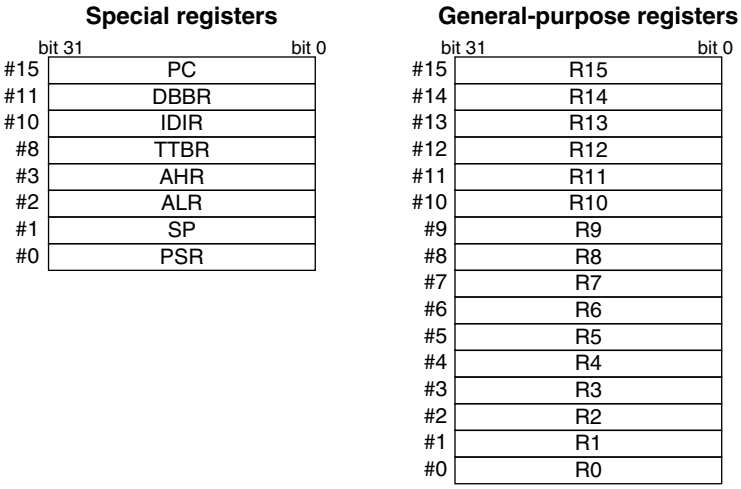


Figure 2.2.1 Registers

2.3 Instruction Set

The C33 PE Core instruction set consists of the function-extended instruction set of the C33 STD Core CPU and the new instructions, in addition to the conventional S1C33-series instructions. Some instructions of the C33 STD Core CPU are deleted. As the C33 PE Core is object-code compatible with the C33 STD Core CPU, software assets can be transported from the S1C33 series to the C33 PE model easily, with minimal modifications required.
All of the instruction codes are 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 "S1C33 Family C33 PE Core Manual."

Table 2.3.1 S1C33-Series-Compatible Instructions

Classification	Mnemonic		Function
Arithmetic operation	add	%rd,%rs	Addition between general-purpose registers
		%rd,imm6	Addition of a general-purpose register and immediate
		%sp,imm10	Addition of SP and immediate (with immediate zero-extended)
	adc	%rd,%rs	Addition with carry between general-purpose registers
	sub	%rd,%rs	Subtraction between general-purpose registers
		%rd,imm6	Subtraction of general-purpose register and immediate
		%sp,imm10	Subtraction of SP and immediate (with immediate zero-extended)
	sbc	%rd,%rs	Subtraction with carry between general-purpose registers
	cmp	%rd,%rs	Arithmetic comparison between general-purpose registers
		%rd,sign6	Arithmetic comparison of general-purpose register and immediate (with immediate zero-extended)
	mlt.h	%rd,%rs	Signed integer multiplication (16 bits × 16 bits → 32 bits)
	mltu.h	%rd,%rs	Unsigned integer multiplication (16 bits × 16 bits → 32 bits)
	mlt.w	%rd,%rs	Signed integer multiplication (32 bits × 32 bits → 64 bits)
	mltu.w	%rd,%rs	Unsigned integer multiplication (32 bits × 32 bits → 64 bits)
Branch	jrgt jrgt.d	sign8	PC relative conditional jump Delayed branching possible — Branch condition: !Z & !(N ^ V)
	jrge jrge.d	sign8	PC relative conditional jump Delayed branching possible — Branch condition: !(N ^ V)
	jrlt jrlt.d	sign8	PC relative conditional jump Delayed branching possible — Branch condition: N ^ V
	jrle jrle.d	sign8	PC relative conditional jump Delayed branching possible — Branch condition: Z | N ^ V
	jrugt jrugt.d	sign8	PC relative conditional jump Delayed branching possible — Branch condition: !Z & !C
	jruge jruge.d	sign8	PC relative conditional jump Delayed branching possible — Branch condition: !C

Classification	Mnemonic		Function
Branch	jrult jrult.d	*sign8*	PC relative conditional jump Branch condition: C Delayed branching possible
	jrule jrule.d	*sign8*	PC relative conditional jump Branch condition: Z | C Delayed branching possible
	jreq jreq.d	*sign8*	PC relative conditional jump Branch condition: Z Delayed branching possible
	jrne jrne.d	*sign8*	PC relative conditional jump Branch condition: !Z Delayed branching possible
	jp	*sign8*	PC relative jump Delayed branching possible
	jp.d	%*rb*	Absolute jump Delayed branching possible
	call	*sign8*	PC relative subroutine call Delayed call possible
	call.d	%*rb*	Absolute subroutine call Delayed call possible
	ret ret.d		Subroutine return Delayed return possible
	reti		Return from interrupt or exception handling
	retd		Return from the debug processing routine
	int	*imm2*	Software exception
	brk		Debug exception
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*]+	Postincrement possible
		%*rd*,[%sp+*imm6*]	Stack (byte) → general-purpose register (sign-extended)
		[%*rb*],%*rs*	General-purpose register (byte) → memory
		[%*rb*]+,%*rs*	Postincrement possible
		[%sp+*imm6*],%*rs*	General-purpose register (byte) → stack
	ld.ub	%*rd*,%*rs*	General-purpose register (byte) → general-purpose register (zero-extended)
		%*rd*,[%*rb*]	Memory (byte) → general-purpose register (zero-extended)
		%*rd*,[%*rb*]+	Postincrement possible
		%*rd*,[%sp+*imm6*]	Stack (byte) → general-purpose register (zero-extended)
	ld.h	%*rd*,%*rs*	General-purpose register (halfword) → general-purpose register (sign-extended)
		%*rd*,[%*rb*]	Memory (halfword) → general-purpose register (sign-extended)
		%*rd*,[%*rb*]+	Postincrement possible
		%*rd*,[%sp+*imm6*]	Stack (halfword) → general-purpose register (sign-extended)
		[%*rb*],%*rs*	General-purpose register (halfword) → memory
		[%*rb*]+,%*rs*	Postincrement possible
		[%sp+*imm6*],%*rs*	General-purpose register (halfword) → stack
	ld.uh	%*rd*,%*rs*	General-purpose register (halfword) → general-purpose register (zero-extended)
		%*rd*,[%*rb*]	Memory (halfword) → general-purpose register (zero-extended)
		%*rd*,[%*rb*]+	Postincrement possible
		%*rd*,[%sp+*imm6*]	Stack (halfword) → general-purpose register (zero-extended)
	ld.w	%*rd*,%*rs*	General-purpose register (word) → general-purpose register
		%*rd*,*sign6*	Immediate → general-purpose register (sign-extended)
		%*rd*,[%*rb*]	Memory (word) → general-purpose register
		%*rd*,[%*rb*]+	Postincrement possible
		%*rd*,[%sp+*imm6*]	Stack (word) → general-purpose register
		[%*rb*],%*rs*	General-purpose register (word) → memory
		[%*rb*]+,%*rs*	Postincrement possible
		[%sp+*imm6*],%*rs*	General-purpose register (word) → stack
System control	nop		No operation
	halt		HALT
	slp		SLEEP
Immediate extension	ext	*imm13*	Extend operand in the following instruction
Bit manipulation	btst	[%*rb*],*imm3*	Test a specified bit in memory data
	bclr	[%*rb*],*imm3*	Clear a specified bit in memory data
	bset	[%*rb*],*imm3*	Set a specified bit in memory data
	bnot	[%*rb*],*imm3*	Invert a specified bit in memory data
Other	swap	%*rd*,%*rs*	Bytewise swap on byte boundary in word
	pushn	%*rs*	Push general-purpose registers *%rs*–%r0 onto the stack
	popn	%*rd*	Pop data for general-purpose registers *%rd*–%r0 off the stack

Table 2.3.2 Function Extended Instructions

Classification	Mnemonic		Function	Extended function
Logical operation	and	%rd,%rs	Logical AND between general-purpose registers	The V flag is cleared after the instruction has been executed.
		%rd,sign6	Logical AND of general-purpose register and immediate	
	or	%rd,%rs	Logical OR between general-purpose registers	
		%rd,sign6	Logical OR of general-purpose register and immediate	
	xor	%rd,%rs	Exclusive OR between general-purpose registers	
		%rd,sign6	Exclusive OR of general-purpose register and immediate	
	not	%rd,%rs	Logical inversion between general-purpose registers (1's complement)	
		%rd,sign6	Logical inversion of general-purpose register and immediate (1's complement)	
Shift and rotate	srl	%rd,%rs	Logical shift to the right (Bits 0–31 shifted as specified by the register)	For rotate/shift operation, it has been made possible to shift 9–31 bits.
		%rd,imm5	Logical shift to the right (Bits 0–31 shifted as specified by immediate)	
	sll	%rd,%rs	Logical shift to the left (Bits 0–31 shifted as specified by the register)	
		%rd,imm5	Logical shift to the left (Bits 0–31 shifted as specified by immediate)	
	sra	%rd,%rs	Arithmetic shift to the right (Bits 0–31 shifted as specified by the register)	
		%rd,imm5	Arithmetic shift to the right (Bits 0–31 shifted as specified by immediate)	
	sla	%rd,%rs	Arithmetic shift to the left (Bits 0–31 shifted as specified by the register)	
		%rd,imm5	Arithmetic shift to the left (Bits 0–31 shifted as specified by immediate)	
	rr	%rd,%rs	Rotate to the right (Bits 0–31 rotated as specified by the register)	
		%rd,imm5	Rotate to the right (Bits 0–31 rotated as specified by immediate)	
	rl	%rd,%rs	Rotate to the left (Bits 0–31 rotated as specified by the register)	
		%rd,imm5	Rotate to the left (Bits 0–31 rotated as specified by immediate)	
Data transfer	ld.w	%rd,%ss	Special register (word) → general-purpose register	The number of special registers that can be used to load data has been increased.
		%sd,%rs	General-purpose register (word) → special register	

Table 2.3.3 Instructions Added to the C33 PE Core

Classification	Mnemonic		Function
Branch	jpr jpr.d	%rb	PC relative jump Delayed branching possible
System control	psrset	imm5	Set a specified bit in PSR
	psrclr	imm5	Clear a specified bit in PSR
Coprocessor control	ld.c	%rd,imm4	Load data from coprocessor
	ld.c	imm4,%rs	Store data in coprocessor
	do.c	imm6	Execute coprocessor
	ld.cf		Load zero to C, V, Z, and N flags
Other	swaph	%rd,%rs	Bytewise swap on halfword boundary in word
	push	%rs	Push single general-purpose register
	pop	%rd	Pop single general-purpose register
	pushs	%ss	Push special registers *%ss*–ALR onto the stack
	pops	%sd	Pop data for special registers *%sd*–ALR off the stack

Table 2.3.4 Instructions Removed

Classification	Mnemonic		Function
Arithmetic operation	div0s	%rs	First step in signed integer division
	div0u	%rs	First step in unsigned integer division
	div1	%rs	Execution of step division
	div2s	%rs	Data correction for the result of signed integer division 1
	div3s		Data correction for the result of signed integer division 2
Other	mirror	%rd,%rs	Bitwise swap every byte in word
	mac	%rs	Multiply-accumulate operation 16 bits × 16 bits + 64 bits → 64 bits
	scan0	%rd,%rs	Search for bits whose value = 0
	scan1	%rd,%rs	Search for bits whose value = 1

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

Table 2.3.5 Symbol Meanings

Symbol	Description
%rs	General-purpose register, source
%rd	General-purpose register, destination
%ss	Special register, source
%sd	Special register, destination
[%rb]	General-purpose register, indirect addressing
[%rb]+	General-purpose register, indirect addressing with postincrement
%sp	Stack pointer
imm2,imm4,imm3,imm5, imm6,imm10,imm13	Unsigned immediate (numerals indicating bit length) However, numerals in shift instructions indicate the number of bits shifted, while those in bit manipulation indicate bit positions.
sign6,sign8	Signed immediate (numerals indicating bit length)

2.4 Debug Mode

The C33 PE Core has debug mode to assist in software development by the user.
The debug mode provides the following functions:

Instruction Break

A debug exception is generated before the set instruction address is executed. An instruction break can be set at two addresses.

Data Break

A debug exception is generated when the set address is accessed for read or write.
A data break can be set at only one address.

Single Step

A debug exception is generated for each instruction.

Forcible Break

A debug exception is generated by an external input signal.

PC Trace

The status of instruction execution by the processor is traced.

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

(1) Suspends the instruction currently being executed.
A debug exception is generated at the end of the E stage of the currently executed instruction, and is accepted at the next rise of the system clock.

(2) Saves the contents of the PC and R0, in that order, to the addresses specified below.
PC → 0x60008
R0 → 0x6000c

(3) Loads the debug exception vector located at the address 0x00060000 to PC and branches to the debug exception handler routine.

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

2.5 Chip ID

The S1C33L27 has chip ID bits shown below that allow the application software to identify CPU type, model, and chip version.

Core ID Bits (D[7:0]/0x20000)

These bits provide an 8-bit ID code that indicates the chip core type.

ID	Chip Core Type
0x02	C33 standard macro core (C33 STD Core)
0x03	C33 mini-macro core
0x04	C33 advanced macro core (C33 ADV Core)
0x05	C33 PE Core
0x06	C33 PE little endian core

The S1C33L27 has adopted the C33 PE little endian core, so the chip core ID is 0x06.

Product Series ID Bits (D[7:0]/0x20001)

These bits provide an 8-bit ID code that indicates the product series of the S1C33 Family.

ID	Product Series
0x03	S1C333xx Series
0x04	S1C334xx Series
0x0E	S1C33Exx Series
0x15	S1C33Lxx Series

The product series ID of the S1C33L27 is 0x15.

Model ID Bits (D[7:0]/0x20002)

These bits provide an 8-bit ID code that indicates the model.
The model ID of the S1C33L27 is 0x27.

Version Bits (D[7:0]/0x20003)

These bits provide an 8-bit ID code that indicates the version number.
0x00 is a version number.

3 Memory Map

Figure 3.1 shows a memory map of the entire S1C33L27 address space. Figure 3.2 shows a memory map of the embedded memory and the internal I/O area.

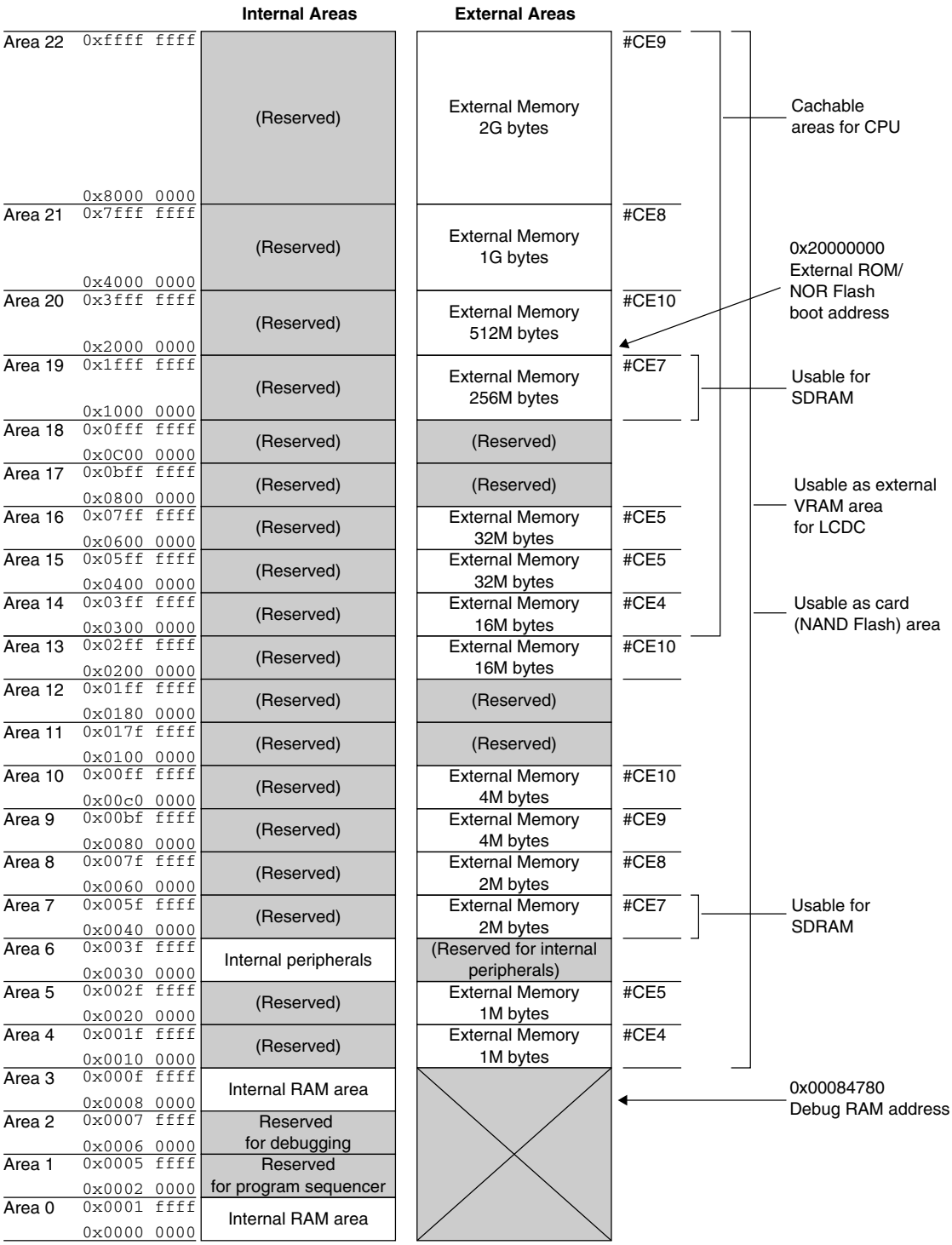


Figure 3.1 Entire Memory Map

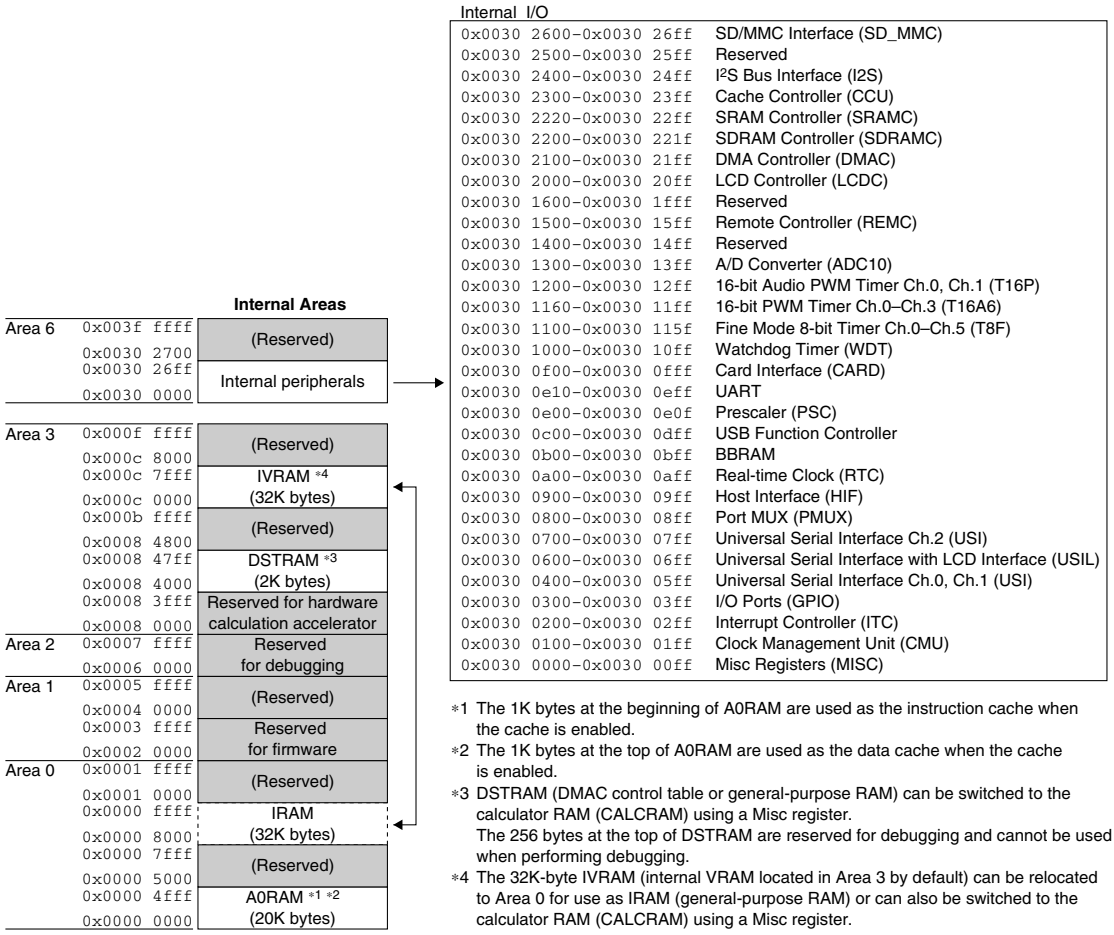


Figure 3.2 Internal Area Map

The following describes the area configuration of the S1C33L27.

3.1 Boot Address

When the S1C33L27 is powered on or reset, the system boots up from a NAND Flash, the host device connected via the host interface, a NOR-Flash/external ROM, an SPI-EEPROM, or the PC connected via RS232C interface specified using the BOOT[1:0] and #CE10 pins. Table 3.1.1 lists the pin status and the boot mode selected.

Table 3.1.1 Boot Mode Settings

Pin configuration			Boot mode	Program execution start address
BOOT1	BOOT0	#CE10		
0	0	1 (Input)	Large page MLC/SLC NAND Flash (>1024 + 32 bytes/page)	The system loads MBR to A0RAM (address 0x0) and starts executing the code loaded.
		0 (Input)	Small page MLC/SLC NAND Flash (≤1024 + 32 bytes/page)	
0	1	Output	Host interface	The system loads MBR to the memory specified by the host device and starts executing the code loaded.
1	0	Output	NOR Flash/external ROM	The system starts executing from the address written at address 0x20000000.
1	1	1 (Input)	SPI-EEPROM	The system loads MBR to A0RAM (address 0x0) and starts executing the code loaded.
		0 (Input)	PC RS232C	

Notes: • The #CE10 pull-up resistor is disabled by default. When using the #CE10 pin as an input pin for NAND Flash, SPI-EEPROM or PC RS232C boot mode, connect an external pull-down resistor to the #CE10 pin to set the pin level to 0 or an external pull-up resistor to set the pin level to 1.

• The BOOT2 pin exists in the PFBGA12U-180 package or chip model. This pin must be connected to VSS.

3.2 Area 0 (A0RAM, IRAM)

3.2.1 A0RAM

Area 0 includes a 20K-byte RAM (A0RAM) that can be used as a general-purpose RAM to store data and execute instructions. A0RAM is located at addresses 0x0 to 0x4fff.

Cache memory

When caching is enabled, the cache controller uses the A0RAM areas shown below as cache memories.

Instruction cache: Address 0x0 to 0x3ff in A0RAM (1K bytes at the beginning of A0RAM)
Data cache: Address 0x4c00 to 0x4fff in A0RAM (1K bytes at the top of A0RAM)

Be sure to avoid accessing these areas from the application when the cache is enabled.
For more information on the caches, see the "Cache Controller (CCU)" chapter.

3.2.2 IRAM

In addition to the 20K-byte A0RAM, the 32K-byte IVRAM (internal VRAM), which is located in Area 3 by default, can be relocated to Area 0 (addresses 0x8000 to 0xffff) to use it as a general-purpose RAM (IRAM). See Section 3.7, "RAM Configuration," for more information on relocation.

IRAM (relocated to Area 0) is accessed without an wait cycle inserted by setting A0WT/MISC_RAMWT register to 0. (See the "Misc Registers (MISC)" chapter for more information on the register.)

3.3 Areas 1 and 2 (Reserved for System)

Areas 1 and 2 are reserved for the system. Be sure to avoid accessing these areas from the user program and the debugger.

3.4 Area 3 (IVRAM, DSTRAM)

3.4.1 IVRAM

Area 3 includes a 32K-byte IVRAM (internal VRAM) for the LCD controller. It is located at addresses 0xc0000 to 0xc7fff and can also be used as a general-purpose RAM.
The 32K-byte IVRAM can be relocated to Area 0 to use it as a high-speed general-purpose RAM or can also be used as the calculator RAM (CALCRAM). See Section 3.7, "RAM Configuration," for more information on relocation.

IVRAM located in Area 3 is accessed without an wait cycle inserted by setting A3WT/MISC_RAMWT register to 0. (See the "Misc Registers (MISC)" chapter for more information on the register.)

3.4.2 DSTRAM

Area 3 includes a 2K-byte RAM (DSTRAM) located at addresses 0x84000 to 0x847ff.
DSTRAM is used to store the control table for the DMA controller or used as a general-purpose RAM. It can also be used as the calculator RAM (CALCRAM). See Section 3.7, "RAM Configuration," for more information on relocation.

DSTRAM is accessed in four cycles.

Notes: • The address range from 0x84700 to 0x847ff (256 bytes) in DSTRAM is reserved for use as the debugging RAM area. Be sure to avoid accessing this area from the user program and the debugger. When using a debugger, specify 0x84780 as the debugging RAM address.

```
c33 das 0x60000 0x84780
```

• When DSTRAM is switched to CALCRAM, locate the DMAC control table in IVRAM (Area 3) or an external RAM.

3.5 Area 6 (I/O Area)

Area 6 is allocated to the I/O area for the internal peripheral circuits.

For details on the internal peripheral circuits mapped to this area, see the descriptions of each peripheral module. For the list of control registers, see the "List of I/O Registers" section in Appendix.

Area 6 includes BBRAM (16 bytes) located at addresses 0x300b00 to 0x300b0f. BBRAM operates with a power source ($RTCV_{DD}$) separated from the system power. Thus, BBRAM can retain data even if the system power is off. BBRAM is accessed in the same manner as the Area 6 registers.

3.6 External Memory Area

Areas 4, 5, 7 to 10, 13 to 16, and 19 to 22 can be used for external memory and other external devices. Configure the SRAMC and/or SDRAMC according to the devices connected. Although the internal address and internal data buses of the S1C33L27 are both 32 bits wide, the maximum external data bus width is 16 bits (D[15:0]) and the maximum external address bus width is 26 bits (A[25:0]) due to the limited number of pins available.

Notes: • An SDRAM can be connected to Area 7 (2M bytes) or Area 19 (256M bytes).

• The external VRAM used for the LCD controller and a NAND Flash can be connected to Areas 4 to 22.

3.7 RAM Configuration

IVRAM and DSTRAM can be configured to a general-purpose RAM or a hardware-specific memory.
RAMCFG[2:0]/MISC_RAMCFG register is used for IVRAM/DSTRAM configuration.

Table 3.7.1 IVRAM/DSTRAM Configuration

RAMCFG[2:0]	IVRAM	DSTRAM
0x7–0x4	CALCRAM (CALC)	DSTRAM (Area 3)
0x3	IRAM (Area 0)	CALCRAM (CALC)
0x2	IRAM (Area 0)	DSTRAM (Area 3)
0x1	VRAM (Area 3)	CALCRAM (CALC)
0x0	VRAM (Area 3)	DSTRAM (Area 3)

(Default: 0x0)

For more information on the MISC_RAMCFG register, see the "Misc Registers (MISC)" chapter.

3.8 Bus Masters and Accessible Memories

The table below lists the bus masters and the memories that can be accessed.

Table 3.8.1 Bus Masters and Accessible Memories

Memory	Bus master				CALC
	CPU	DMAC	HIF	LCDC	
A0RAM (Area 0)	✓	–	–	–	–
IVRAM → IRAM (Area 0)	✓	–	–	–	–
IVRAM (Area 3)	✓	✓	✓	✓	–
IVRAM → CALCRAM (CALC)	–	–	–	–	✓
DSTRAM (Area 3)	✓	✓	✓	–	–
DSTRAM → CALCRAM (CALC)	–	–	–	–	✓
I/O registers (Area 6)	✓	✓	✓	–	–
External memory	✓	✓	✓	✓	–

✓: Can be accessed. –: Cannot be accessed.

3.9 Memory Access Rate

Table 3.9.1 Internal RAM and External SRAM Access Rate

Memory access cycle	Number of access cycles				
	Single byte	Single half-word	Single word	Successive address	Burst access
A0RAM (20KB/Area 0) read with cache enabled or disabled	2	2	2	–	–
A0RAM (20KB/Area 0) write with cache enabled or disabled	1	1	1	–	–
IRAM (32KB/Area 0) read	2 + W	2 + W	2 + W	1 or 1 + W	–
IRAM (32KB/Area 0) write	1	1	1	–	–
DSTRAM (2KB/Area 3) read	4	4	4	–	–
DSTRAM (2KB/Area 3) write	4	4	4	–	–
IVRAM (32KB/Area 3) read	4 + W	4 + W	4 + W	–	1 + W (∗1)
IVRAM (32KB/Area 3) write	4	4	4	–	–
Area 6 register (8-bit) read	6	9	15	–	–
Area 6 register (8-bit) write	3	6	12	–	–
Area 6 register (16-bit) read	6	6	9	–	–
Area 6 register (16-bit) write	3	3	6	–	–
Area 6 register (32-bit) read	6	6	6	–	–
Area 6 register (32-bit) write	3	3	3	–	–
External SRAM/ROM (8-bit) read/write	4 + S + W + H	6 + (S + W + H) × 2	8 + (S + W + H) × 4	–	(∗2)
External SRAM/ROM (16-bit) read/write	4 + S + W + H	4 + S + W + H	6 + (S + W + H) × 2	–	(∗2)

W: Number of WAIT cycles (0 min.) S: Number of SETUP cycles (1 min.) H: Number of HOLD cycles (1 min.)

(Note)

∗1: The cache controller does not support IVRAM, so burst read will be issued by the LCDC when IVRAM is used as a VRAM or by the DMAC when used as a DMA control table.

∗2: Burst access can be regarded as 4 or 8 single word access cycles.

4 Power Supply

This section explains the operating voltage of the S1C33L27.

4.1 Power Supply Pins

The S1C33L27 has the power supply pins shown in Table 4.1.1.

Table 4.1.1 Power Supply Pins

No.	Pin name	Description	Power voltage
1	**HVDD**	I/O power supply pin	3.3 V typ. (2.7 to 3.6 V) (3.0 to 3.6 V when the USB module is used.)
2	**AVDD**	Analog power supply pin	3.3 V typ. (= HVDD)
3	**LVDD**	Internal logic/internal memory power supply pin	1.8 V typ. (1.65 to 1.95 V) (1.7 to 1.9 V when a ceramic resonator is used.)
4	**RTCVDD**	RTC/BBRAM power supply pin	1.8 V typ. (= LVDD)
5	**VSS**	Ground pin	GND
6	**PLLVDD**	PLL power supply pin	1.8 V typ. (= LVDD)
7	**PLLVSS**	PLL power supply ground pin	GND (= VSS)

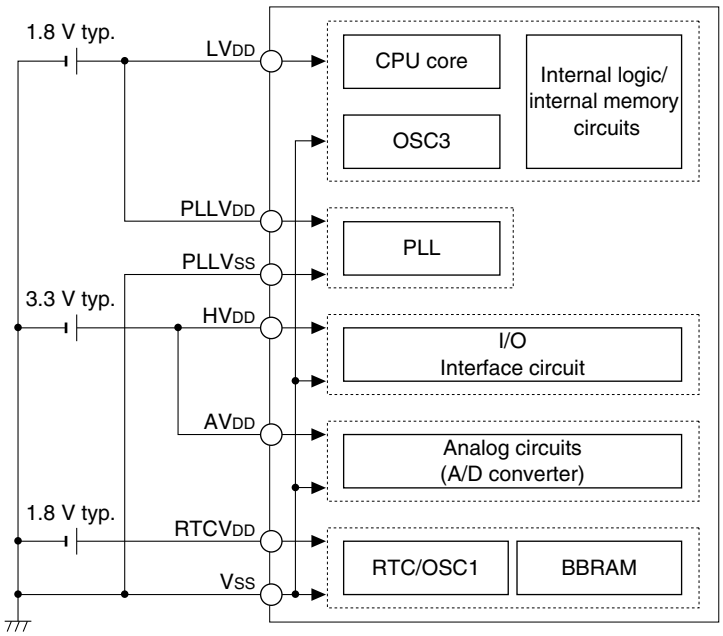


Figure 4.1.1 Power Supply System

4.2 Operating Voltage (LV_{DD})

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

LV_{DD} = 1.65 to 1.95 V (V_{SS} = GND)
or 1.7 to 1.9 V when a ceramic resonator is used.

Note: The S1C33L27 packages have more than one LV_{DD} and V_{SS} pins. Be sure to supply the operating voltage to all the pins. Do not open any of them.

4.3 Power Supply for PLL ($PLLV_{DD}$, $PLLV_{SS}$)

The PLL power supply pins ($PLLV_{DD}$ and $PLLV_{SS}$) are provided separately from the LV_{DD} and V_{SS} pins so that the digital circuits will not affect the PLL circuit. Supply the same voltage level as the LV_{DD} to the $PLLV_{DD}$.

$PLLV_{DD} = LV_{DD}$, $PLLV_{SS} = V_{SS}$

Noise on the PLL power lines decrease the PLL output precision, so use a stabilized power supply and make the board pattern with consideration given to that.

4.4 Power Supply for RTC ($RTCV_{DD}$)

The RTC power supply pin ($RTCV_{DD}$) is provided separately from the LV_{DD} pin in order to run the RTC and OSC1 oscillator at system power down. Supply the same voltage level as the LV_{DD} to the $RTCV_{DD}$ pin.

$RTCV_{DD} = LV_{DD}$ (V_{SS} = GND)

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

4.5 I/O Interface Voltage (HV_{DD})

The HV_{DD} voltage is used for interfacing with external I/O signals. For the output interface of the S1C33L27, the HV_{DD} voltage is used as high level and the V_{SS} voltage as low level. The V_{SS} pin is used for the ground common with LV_{DD}. The effective HV_{DD} voltage range is as follows:

HV_{DD} = 2.7 to 3.6 V (V_{SS} = GND)
or 3.0 to 3.6 V when the USB module is used

Note: The S1C33L27 packages have more than one HV_{DD} pins. Be sure to supply the operating voltage to all the pins. Do not open any of them.

4.6 Power Supply for Analog Circuits (AV_{DD})

The analog power supply pin (AV_{DD}) is provided separately from other power supply pins so that the digital circuits will not affect the analog circuits (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 effective AV_{DD} voltage range is as follows:

AV_{DD} = 2.7 to 3.6 V (V_{SS} = GND)

Note: Be sure to supply a voltage within the range from 2.7 to 3.6 V to the AV_{DD} pin even if the analog circuit is not used.

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.

4.7 Precautions on Power Supply

Power-on sequence

In order to operate the device normally, supply power in accordance with the following timing.
To ensure that the device will operate normally, observe the timing requirements given below when turning the power on.

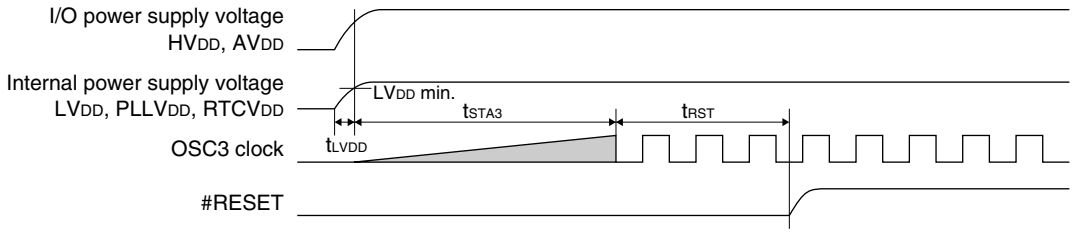


Figure 4.7.1 Power-On Sequence

(1) t_{LVDD}: Elapsed time until the power supply stabilizes after power-on
Supply power in the following sequence.
Power-on: 1. LV_{DD}, $PLLV_{DD}$, (and $RTCV_{DD}$)
2. HV_{DD}, AV_{DD} (May be applied with 1 above at the same time.)
3. Apply the input signal

∗ The $RTCV_{DD}$ can be always supplied to the chip to operate the RTC and BBRAM.

(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 three clocks; Keep the #RESET signal low.

Power-off sequence

Shut off the power supply in the following sequence.
Power-off: 1. Turn off the input signal
2. HV_{DD}, AV_{DD}
3. LV_{DD}, $PLLV_{DD}$ (and $RTCV_{DD}$) (May be turned off with 2 above at the same time.)

Note: Be sure to avoid applying HV_{DD} or AV_{DD} for a duration of one second or more when the LV_{DD} power is off, as a breakdown may occur in the device or the characteristics may be degraded due to flow-through current of the HV_{DD} or AV_{DD}.

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 LV_{DD} and V_{SS} 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 LV_{DD} 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 LV_{DD} and V_{SS} 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 V_{SS}. 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 LV_{DD}, HV_{DD}, AV_{DD}, or V_{SS}.
(4) No outputs should be shorted.

5 Reset and NMI

5.1 Initial Reset

The S1C33L27 has two initial reset sources that initialize the internal circuits.

(1) #RESET pin (external initial reset)
(2) Watchdog timer (software selectable internal initial reset)

Figure 5.1.1 shows the configuration of the initial reset circuit.

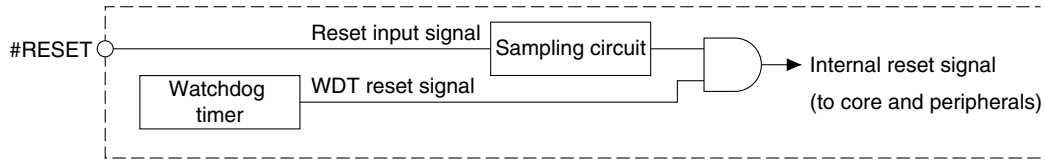


Figure 5.1.1 Configuration of Initial Reset Circuit

The CPU and peripheral circuits are initialized by the active signal from an initial reset source. When the reset signal is negated, the CPU starts reset handling. The reset handling reads the reset vector (reset handler start address) from the beginning of the vector table and starts executing the program (initial routine) beginning with the read address.

5.1.1 #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. And as shown in Figure 5.1.1.1, to assert the internal reset signal, low level 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 ensure that the chip is reset. Also the internal reset signal is negated when the default OSC3 oscillation stabilization wait cycle has elapsed after the #RESET pin goes high.
The S1C33L27 is reset by the low state (= 0) on the internal reset signal, and starts operating when the reset signal goes high (= 1).

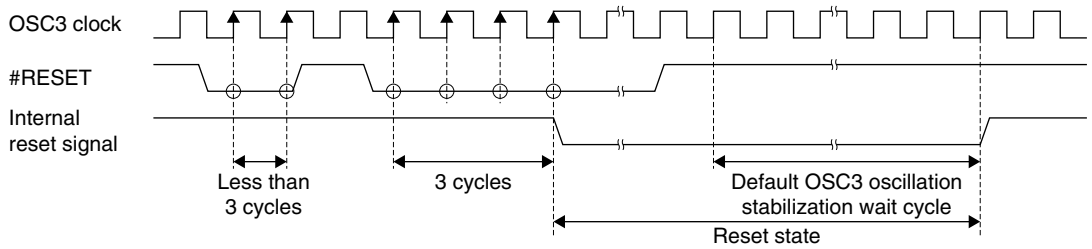


Figure 5.1.1.1 #RESET Sampling

5.1.2 Resetting by the Watchdog Timer

The S1C33L27 has a built-in watchdog timer to detect runaway of the CPU. The watchdog timer outputs a signal if it is not reset via software (due to CPU runaway) in the programmed cycles. The output signal can generate either NMI or reset. Write 1 to the RESEN/WDT_EN register to generate reset.
For details of the watchdog timer, see the "Watchdog Timer (WDT)" chapter.

Notes: • When using the reset function of the watchdog timer, program the watchdog timer so that it will be reset within the programmed cycles to avoid occurrence of an unnecessary reset.

• The reset function of the watchdog timer cannot be used for power-on reset as it must be enabled with software.

5.1.3 Initial Reset Sequence

Even if the #RESET pin input negates the reset signal after power is turned on, the CPU cannot boot up until the oscillation stabilization waiting cycle (128 OSC3 clock cycles) has elapsed.
Figure 5.1.3.1 shows the operating sequence following cancellation of initial reset.
The CPU starts operating in synchronization with the OSC3 clock after reset state is canceled.

Note: The oscillation stabilization cycle described in this section does not include oscillation start time. Therefore the time interval until the CPU starts executing instructions after power is turned on may be longer than that indicated in the figure below.

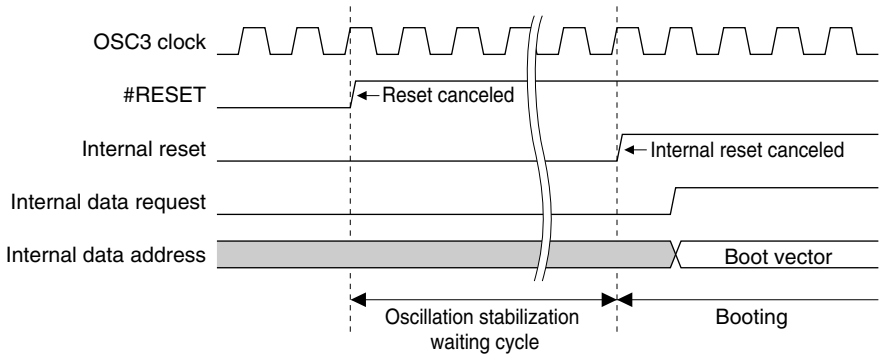


Figure 5.1.3.1 Operation Sequence Following Cancellation of Initial Reset

5.1.4 Initial Reset Status

The C33 PE Core and internal peripheral circuits are initialized while the internal reset signal is at kept 0. The following shows the reset status of the internal IC with the initial reset.

Table 5.1.4.1 Initial Reset Status

Item	Boot mode	Initial reset status
CPU - TTBR	All boot modes	Initialized to 0x20000000
CPU - PC	NOR Flash/ external ROM boot	The reset vector at address 0x20000000 is loaded to the PC.
	NAND Flash boot	0x0 is loaded to the PC.
	SPI-EEPROM boot	
	PC RS232C boot	
	HIF boot	The address stored in the HIF scratchpad register is loaded to the PC.
CPU - PSR	–	Undefined
CPU - Other registers	–	Undefined
CPU - Operating clock	–	The CPU operates with OSC3 × 1/1 clock.
Oscillator circuits	–	Both the high-speed (OSC3) and low-speed (OSC1) oscillator circuits are turned on. (PLL and SSCG are turned off.)
Clock supply to peripheral modules	–	Clocks are supplied to all the peripheral modules except LCDC, USB, and HIF.
I/O pin state	–	I/O pins are initialized. (See the “Pin Functions” section in the “Overview” chapter or the “Boot” chapter in the supplementary volume.)
Other internal peripheral circuits	–	Initialized or undefined (See each I/O map.)

Note: The S1C33L27 does not support a hot reset feature that maintains I/O pin status and the TTBR value.

5.1.5 Precautions to be Taken during Initial Reset

Core CPU

When initially reset, all internal registers of the core CPU are undefined. Therefore, these registers must be initialized in a program. In particular, the Stack Pointer (SP) should always be initialized before accessing the stack. Note that NMI requests are masked with hardware until data is written to the SP after an initial reset, to prevent erratic operation.

Internal RAM

The contents of internal RAM are undefined when initially reset. Internal RAM must be initialized as required.

High-speed (OSC3) oscillator circuit

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

Low-speed (OSC1) oscillator circuit

When the chip is reset at power-on or while the low-speed (OSC1) oscillator circuit is idle, the low-speed (OSC1) oscillator circuit also starts oscillating. The low-speed (OSC1) oscillator circuit requires a longer time for starting oscillation than the high-speed (OSC3) oscillator circuit. (See the electrical characteristics table.) To prevent erratic operation due to an unstable clock, the OSC1 clock should not be used only after this start time elapsed.

I/O ports and I/O pins

Initial reset initializes the I/O port control and data registers, 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 undefined by initial reset. Therefore, setup of these registers with a program is required.
For the specific initial settings done on the peripheral circuits after an initial reset, see each I/O map or circuit descriptions.

5.2 NMI Input

The S1C33L27 has three NMI sources that generate NMI.

(1) #NMI pin (external input)
(2) Watchdog timer (software selectable)
(3) Host processor via the host interface (software selectable)

Figure 5.2.1 shows the configuration of the NMI circuit.

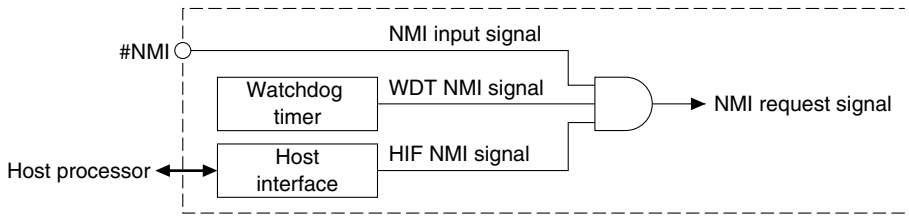


Figure 5.2.1 Configuration of NMI Circuit

The NMI signal, which is input from the #NMI pin, generated by the watchdog timer (WDT), or generated by the host processor via the host interface (HIF), generates a nonmaskable interrupt to the C33 PE Core. This interrupt takes precedence over other interrupts and is unconditionally accepted by the C33 PE Core.
For details about NMI exception handling by the C33 PE Core, refer to the "S1C33 Family C33 PE Core Manual."

5.2.1 #NMI Pin

The #NMI pin is used to generate a non-maskable interrupt to the C33 PE Core.
To generate an NMI (Non-Maskable Interrupt) to the C33 PE Core, the #NMI pin must be maintained at a low level for three or more system clock cycles.

5.2.2 NMI by the Watchdog Timer

The S1C33L27 has a built-in watchdog timer to detect runaway of the CPU. The watchdog timer outputs a signal if it is not reset with software (due to CPU runaway) in the programmed cycles. The output signal can generate either NMI or reset. Write 1 to the NMIEN/WDT_EN register to generate NMI.
For details of the watchdog timer, see the "Watchdog Timer (WDT)" chapter.

5.2.3 NMI by the Host Processor

Writing 1 from the external host processor to the HINT/HIF_FLAG1 register generates an NMI. However, HINT_MD/HIF_EXTCTRL1 register must be set to 0 to enable the host interface for generating NMI.
For more information on NMI generation by the host processor, see the "Host Interface (HIF)" chapter.

6 Clock Management Unit (CMU)

6.1 CMU Module Overview

The CMU module controls the internal oscillators and the system clock.
The features of the CMU module are listed below.

- Generates the operating clocks with the built-in oscillators.
  - OSC3 oscillator circuit: 48 MHz (max.) crystal or ceramic oscillator circuit
    Supports an external clock input.
  - OSC1 oscillator circuit: 32.768 kHz (typ.) crystal oscillator circuit
    Supports an external clock input.
- Switches the system clock. The system clock source can be selected from OSC3, PLL, and OSC1 with software.
- Controls PLL and SSCG.
- Generates the system clock by dividing the source clock by 1 to 32.
- Controls the clock supply to the peripheral modules.
- Controls the clocks according to the standby mode (HALT, or SLEEP).
- Controls a clock output to external devices.

To reduce current consumption, control the clock in conjunction with processing and use standby mode. For more information on reducing current consumption, see "Power Saving" in Appendix.
Figure 6.1.1 shows the clock system and CMU module configuration.

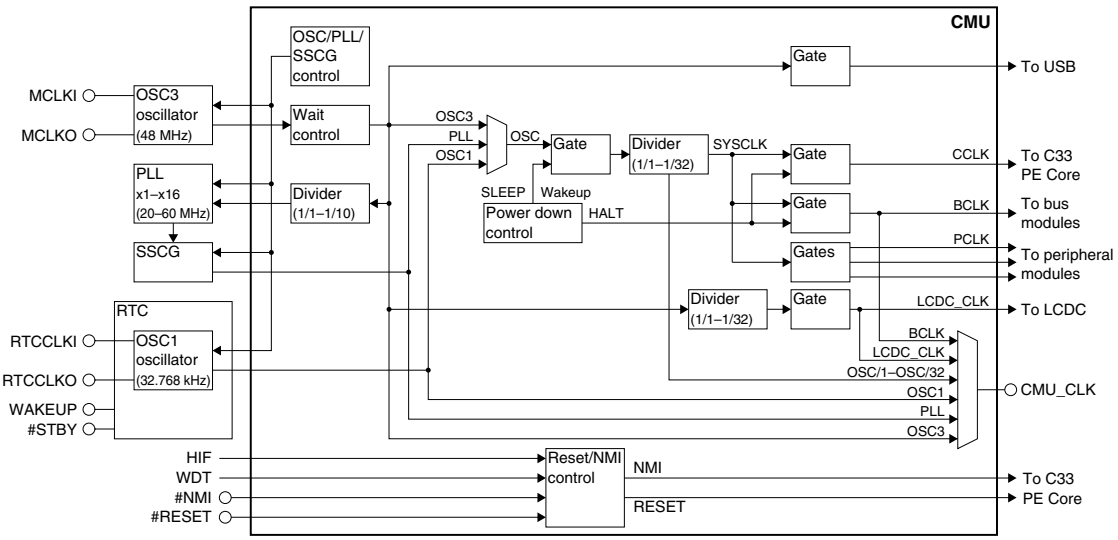


Figure 6.1.1 CMU Module Configuration

Note: The CMU control registers (addresses 0x300100 to 0x30010d and 0x300114 to 0x300118) are write-protected. Before the CMU control registers can be rewritten, write protection of these registers must be removed by writing data 0x96 to CMUP[7:0]/CMU_PROTECT register. Note that since unnecessary rewrites to the CMU control registers could lead to erratic system operation, CMUP[7:0] should be set to other than 0x96 unless the CMU control registers must be rewritten.

6.2 CMU Pins

Table 6.2.1 lists the input/output pins for the CMU module.

Table 6.2.1 List of CMU Pins

Pin name	I/O	Qty	Function
RTCCLKI	I	1	OSC1 oscillator input pin Connect a crystal resonator (32.768 kHz), a feedback resistor, and a gate capacitor. Or input an external clock.
RTCCLKO	O	1	OSC1 oscillator output pin Connect a crystal resonator (32.768 kHz), a feedback resistor, a drain resistor, and a drain capacitor.
MCLKI	I	1	OSC3 oscillator input pin Connect a crystal or ceramic resonator (max. 48 MHz), a feedback resistor, and a gate capacitor. Or input an external clock.
MCLKO	O	1	OSC3 oscillator output pin Connect a crystal or ceramic resonator (max. 48 MHz), a feedback resistor, a drain resistor, and a drain capacitor.
CMU_CLK	O	1	CMU_CLK output pin Outputs the clock selected from OSC3, OSC1, PLL, OSC/1–OSC/32, and LCDC_CLK.

The CMU output pin (CMU_CLK) is shared with an I/O port and are initially set as general purpose I/O port pin. The pin function must be switched using the port function select bit to use the general purpose I/O port pin as the CMU output pin. For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

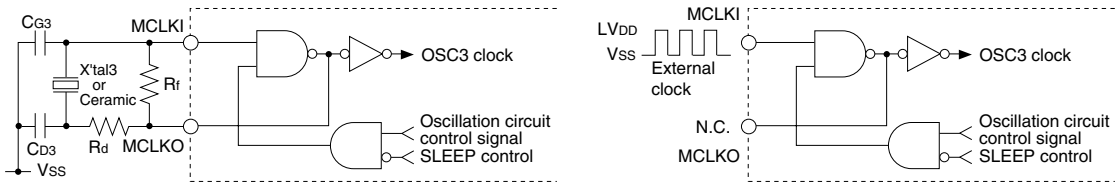
6.3 Oscillators

6.3.1 OSC3 Oscillator Circuit

The OSC3 oscillator circuit generates the main clock for high-speed operation of the C33 PE Core and peripheral circuits.

Structure of the OSC3 oscillator circuit

The OSC3 oscillator circuit accommodates a crystal/ceramic oscillator and external clock input.
Figure 6.3.1.1 shows the structure of the OSC3 oscillator circuit.



(1) Crystal/ceramic oscillation circuit (2) External clock input

Figure 6.3.1.1 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 MCLKI and MCLKO pins and V_{SS}. To use an external clock, leave the MCLKO pin open and input a LV_{DD}-level clock (with a 50% duty cycle) to the MCLKI pin.
The range of oscillation frequencies is as follows:

- Crystal oscillator: 5 MHz (min.) to 48 MHz (max.)
- Ceramic oscillator: 5 MHz (min.) to 48 MHz (max.)
- External clock input: 2 MHz (min.) to 48 MHz (max.)
- A 48 MHz clock source with 0.25% of accuracy should be connected for using the USB function.
- Before using a ceramic resonator, please be sure to contact Murata Manufacturing Co., Ltd. for further information on conditions of use for ceramic resonators.

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

OSC3 oscillation on/off

The OSC3 oscillator circuit stops oscillating when OSC3EN/CMU_OSCCTL register is set to 0 and starts oscillating when set to 1. The OSC3 oscillator circuit stops oscillating in SLEEP mode. After an initial reset, OSC3EN is set to 1 and the OSC3 oscillator circuit is activated.

Stabilization wait time at start of OSC3 oscillation

The OSC3 oscillator circuit incorporates an oscillation stabilization wait timer to prevent malfunctions due to unstable clock operations at the start of OSC3 oscillation—e.g., after an initial reset or waking from SLEEP mode when OSC3 or PLL is used as the system clock source. The OSC3 or PLL clock is not supplied to the system until the time set for this timer has elapsed. Use OSC3WT[3:0]/CMU_OSCCTL register to select one of 16 oscillation stabilization wait times.

Table 6.3.1.1 OSC3 Oscillation Stabilization Wait Time Settings

OSC3WT[3:0]	Oscillation stabilization wait time
0xf	128 cycles
0xe	256 cycles
0xd	512 cycles
0xc	1,024 cycles
0xb	2,048 cycles
0xa	4,096 cycles
0x9	8,192 cycles
0x8	16,384 cycles
0x7	32,768 cycles
0x6	65,536 cycles
0x5	131,072 cycles
0x4	262,144 cycles
0x3	524,288 cycles
0x2	1,048,576 cycles
0x1	2,097,152 cycles
0x0	4,194,304 cycles

(Default: 0xf)

This is set to 128 cycles (OSC3 clock) after an initial reset.

Notes: • The OSC3 oscillation stabilization wait timer cannot be used when the OSC3 oscillator is turned on with software. Therefore, a software wait routine must be implemented.

• Oscillation stability will vary, depending on the resonator and other external components. Carefully consider the OSC3 oscillation stabilization wait time before reducing the time. When waking from SLEEP mode if OSC3 or PLL is used as the system clock source, set the OSC3 oscillation stabilization wait time as follows:

OSC3 oscillation stabilization wait time [cycle] ≥ OSC3 oscillation start time [s] (max.) × f_{SYSCLK} [Hz]

f_{SYSCLK}: SYSCLK frequency when the clock source is OSC3 or PLL.

Example: When OSC3 oscillation start time (max.) = 25 ms and f_{SYSCLK} = 48 MHz
OSC3 oscillation stabilization wait time ≥ 1,200,000 [cycles]

OSC3WT[3:0] should be set to 0x1 (OSC3 oscillation stabilization wait time = 2,097,152 cycles).

6.3.2 OSC1 Oscillator Circuit

The S1C33L27 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.

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 6.3.2.1 shows the structure of the OSC1 oscillator circuit.

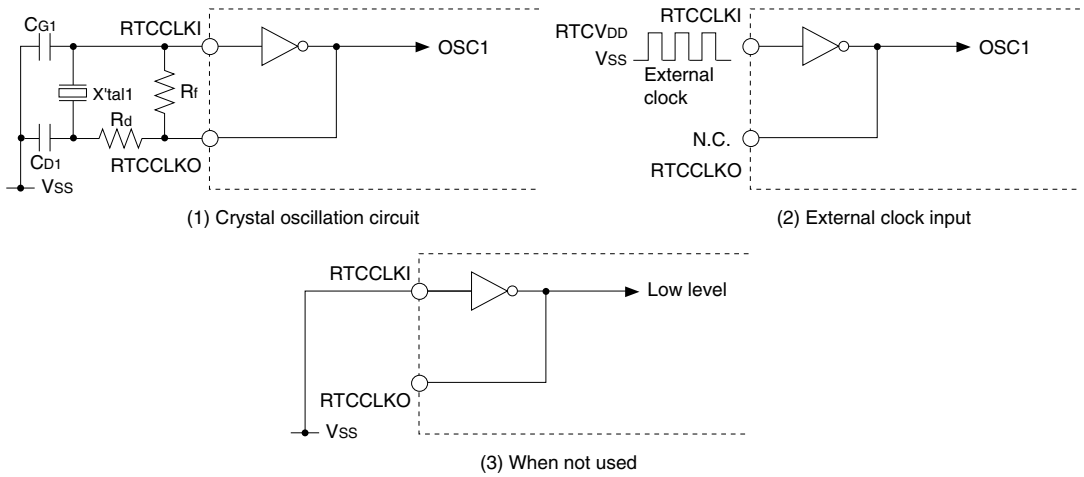


Figure 6.3.2.1 OSC1 Oscillator Circuit

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 RTCCLKI and RTCCLKO pins and V_{SS}, as shown in the figure above.

To use an external clock, leave the RTCCLKO pin open and input an $RTCV_{DD}$-level clock (whose duty cycle is 50%) to the RTCCLKI 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 RTCCLKI pin to V_{SS} and leave the RTCCLKO pin open.

OSC1 oscillation on/off

The OSC1 oscillator circuit stops oscillating when OSC1EN/CMU_OSCCTL register is set to 0 and starts oscillating when set to 1. After an initial reset, OSC1EN is set to 1 and the OSC1 oscillator circuit is activated. The OSC1 oscillator circuit does not stop oscillating in SLEEP mode.

Note: A finite time (see "Electrical Characteristics") is required until oscillation stabilizes after the OSC1 oscillator starts oscillating. To prevent system malfunction, do not use the oscillator-derived clock until this oscillation stabilization time elapses.

6.4 PLL

The PLL multiplies the OSC3 clock frequency by a given value to generate a source clock for high-speed operation.

6.4.1 PLL On/Off Control

PLLPOWR/CMU_PLLCTL0 register can be used to turn the PLL on or off. Setting PLLPOWR to 1 initiates PLL operation. When initially reset, PLLPOWR is set to 0 (power-down mode), with the PLL turned off.

Notes:
- An output stabilization time is required (see "Electric Characteristics") immediately after the PLL is started by setting PLLPOWR to 1. After the PLL is turned on, insert a waiting time for the output clock to stabilize, before switching the system clock source to the PLL.
- The PLL must be turned off before setting the CPU to SLEEP mode (before executing the slp instruction).
- Switch the system clock source to OSC3 or OSC1 before turning the PLL off.

6.4.2 Selecting the PLL Input Clock

The PLL input clock can be selected from among 10 kinds of OSC3 divided clocks, OSC3/1 to OSC3/10, using PLLINDIV[3:0]/CMU_PLLINDIV register.

Table 6.4.2.1 PLL Input Clock (OSC3 Division Ratio) Selections

PLLINDIV[3:0]	Division ratio (OSC3/n)
0xf–0xa	1/8
0x9	1/10
0x8	1/9
0x7	1/8
0x6	1/7
0x5	1/6
0x4	1/5
0x3	1/4
0x2	1/3
0x1	1/2
0x0	1/1

(Default: 0x7)

Notes: • The PLL input clock can only be selected when the PLL is turned off (PLLPOWR/CMU_PLLCTL0 register = 0) and the clock source is other than the PLL (CLKSEL[1:0]/CMU_OSCSEL register is not 0x2). If the PLL input clock is changed while the system is operating with the PLL clock, the system may operate erratically.

• For the range of the input clock frequency, see "Electrical Characteristics."

6.4.3 Setting the Frequency Multiplication Rate

The PLL frequency multiplication rate can be specified as shown in the table below using PLLN[3:0]/CMU_PLLCTL0 register.

Table 6.4.3.1 PLL Frequency Multiplication Rates

PLLN[3:0]	Multiplication rate
0xf	x16
0xe	x15
0xd	x14
0xc	x13
0xb	x12
0xa	x11
0x9	x10
0x8	x9
0x7	x8
0x6	x7
0x5	x6
0x4	x5
0x3	x4
0x2	x3
0x1	x2
0x0	x1

(Default: 0x0)

PLL output clock frequency = PLL input clock frequency × multiplication rate

Notes: • The frequency multiplication rate must be set so that the PLL output clock frequency does not exceed the upper-limit operating clock frequency. For the multiplication rates that can be set and the range of the output clock frequency, see "Electrical Characteristics."

• The frequency multiplication rate can only be set when the PLL is turned off (PLLPOWR/CMU_PLLCTL0 register = 0) and the clock source is other than the PLL (CLKSEL[1:0]/CMU_OSCSEL register is not 0x2). If the frequency multiplication rate is changed while the system is operating with the PLL clock, the system may operate erratically.

6.4.4 Other PLL Settings

V-Divider

To ensure that frequency fvco obtained by <output frequency × W> falls within the range of 100 to 400 MHz, set the proper W value using PLLV[1:0]/CMU_PLLCTL0 register. Lower value is better for low power consumption.

Table 6.4.4.1 W Value Settings

PLLV[1:0]	W
0x3	8
0x2	4
0x1	2
0x0	Setting prohibited

(Default: 0x1)

VCO Kv constant (VC value)

According to the range of fvco frequencies obtained by <output frequency × W>, set the VCO Kv circuit constant (VC value) using PLLVC[3:0]/CMU_PLLCTL1 register.

Table 6.4.4.2 VC Value Settings

PLLVC[3:0]	fvco [MHz]
0x8	360 < fvco ≤ 400
0x7	320 < fvco ≤ 360
0x6	280 < fvco ≤ 320
0x5	240 < fvco ≤ 280
0x4	200 < fvco ≤ 240
0x3	160 < fvco ≤ 200
0x2	120 < fvco ≤ 160
0x1	100 ≤ fvco ≤ 120
Other	Setting prohibited

(Default: 0x1)

LPF resistance value (RS value)

According to the input clock frequency, set the LPF resistance value (RS value) of the PLL by using PLLRS[3:0]/CMU_PLLCTL1 register.

Table 6.4.4.3 RS Value Settings

PLLRS[3:0]	fREFCK [MHz]
0xa	5 ≤ fREFCK < 20
0x8	20 ≤ fREFCK ≤ 150
Other	Setting prohibited

(Default: 0x8)

LPF capacitance value (CS value)

Bits to set the LPF capacitance value (CS value) is provided in the CMU control registers, PLLCS[1:0]/CMU_PLLCTL2 register. However, do not alter the value of these bits, and leave them as initially set (0x0).

Charge pump current value (CP value)

Bits to set the charge pump current value (CP value) is provided in the CMU control registers, PLLCP[4:0]/CMU_PLLCTL2 register. However, do not alter the value of these bits, and leave them as initially set (0x10).

Table 6.4.4.4 PLL Setting Examples

PLL input clock	PLL output clock	PLLN[3:0]	PLLV[1:0]	PLLVC[3:0]	PLLRS[3:0]
6 MHz	60 MHz	x10 (0x9)	0x1	0x1	0xa
10 MHz	60 MHz	x6 (0x5)	0x1	0x1	0xa
	40 MHz	x4 (0x3)	0x2	0x2	0xa
20 MHz	60 MHz	x3 (0x2)	0x1	0x1	0x8
	40 MHz	x2 (0x1)	0x2	0x2	0x8
30 MHz	60 MHz	x2 (0x1)	0x1	0x1	0x8

Note: The PLL can only be set up when the PLL is turned off (PLLPOWR/CMU_PLLCTL0 register = 0) and the clock source is other than the PLL (CLKSEL[1:0]/CMU_OSCSEL register is not 0x2). If settings are changed while the system is operating with the PLL clock, the system may operate erratically.

6.4.5 Power Supply for PLL

In order to prevent undesirable effects of noise, the PLLVDD and PLLVSS pins are provided, in addition to the internal logic/internal memory power supply, to feed power to the PLL. Make sure that the following voltages are supplied to the respective pins.
PLLVDD pin: Supply LVDD level voltage.
PLLVSS pin: Set to VSS level.

For pin assignments, see the "Pin Descriptions" section.

6.5 SSCG

The SSCG (Spread Spectrum Clock Generator) is a circuit used to reduce EMI (Electromagnetic Interference) noise by conducting spread spectrum (or SS modulation) on the PLL output clock signal. SSCG conducts SS modulation on the PLL output clock signal. Thus SS modulation contributes in reducing noise when the PLL output clock is selected as the system clock source since, in this case, the SS modulation is effective for all the operating clocks for the core and peripheral circuits (except the RTC that uses the OSC1 clock).

Note: When the OSC3 or OSC1 clock is selected as the system clock source, SS modulation is not performed for the operating clock (system clock).

∗ **About spectrum spread (SS modulation)**

The SSCG performs SS modulation by adjusting the width of the high section of the input clock. This adjustment is made by increasing or reducing the set value of the internal delay adjust circuit of the SSCG. The maximum width within which the set value is changed constitutes the maximum frequency change width. The relevant control register is used to set the upper-limit value of this width. In the SSCG, an interval timer adjusts the interval at which the set value changes. The relevant control register is also used to set this interval (frequency change cycle).

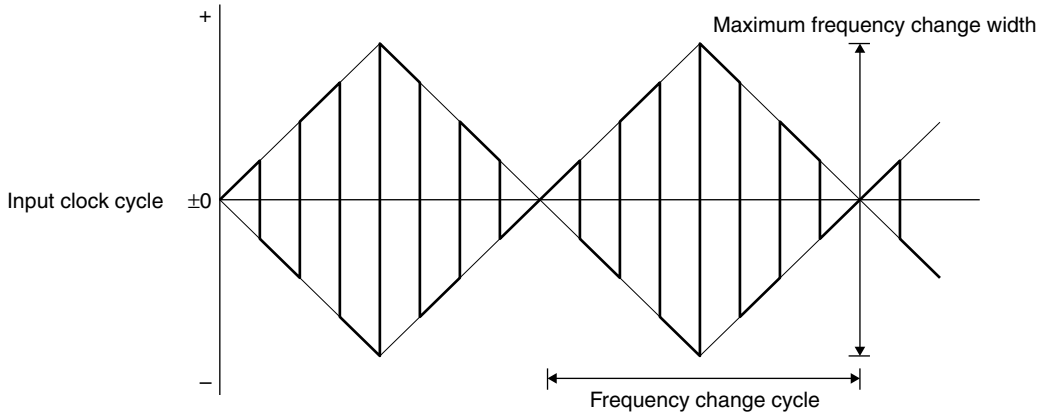


Figure 6.5.1 SS Modulation

6.5.1 SSCG On/Off Control

The SSCG can be turned on or off using SSMCON/CMU_SSCG0 register. Setting SSMCON to 1 causes the SSCG to start operating. When initially reset, SSMCON is initialized to 0, with the SSCG turned off (bypassed).

Notes: • A stabilized clock must be supplied to the SSCG module when turning the SSCG on and off. The following shows the operation procedure.

To turn the SSCG on
1. Turn the PLL on.
2. Wait for the PLL stabilization time to elapse at the minimum.
3. Turn the SSCG on.

To turn the SSCG off
1. Turn the SSCG off.
2. Turn the PLL off.

- SS modulation is conducted on the PLL output clock signal. SS modulation is not applicable to the signals other than the PLL clock signal. When the PLL output clock is not used for the system clock, turn the SSCG off.

6.5.2 SS Modulation Parameter Settings

As described in "About spectrum spread (SS modulation)" above, it is necessary to set the upper-limit value of the maximum frequency change width and the frequency change cycle.

The maximum frequency change width should be set to the appropriate value according to the PLL output clock frequency as shown in the table below using SSMCIDT[3:0]/CMU_SSCG1 register. The maximum frequency change width will be about ±2% of the PLL output clock by the above setting.

Table 6.5.2.1 Maximum Frequency Change Width Settings

PLL output clock frequency f [MHz]	SSMCIDT[3:0]
f ≤ 19.8	0xf
19.8 < f ≤ 21.2	0xe
21.2 < f ≤ 22.5	0xd
22.5 < f ≤ 24.2	0xc
24.2 < f ≤ 25.9	0xb
25.9 < f ≤ 28.4	0xa
28.4 < f ≤ 30.8	0x9
30.8 < f ≤ 34.2	0x8
34.2 < f ≤ 37.8	0x7
37.8 < f ≤ 43.1	0x6
43.1 < f ≤ 48.9	0x5
48.9 < f ≤ 58.5	0x4
58.5 < f ≤ 60.0	0x3
–	0x2
–	0x1
–	0x0

(Default: 0x0)

SSMCITM[3:0]/CMU_SSCG1 register is used to set the frequency change cycle. However, always set it to 0x1.

Notes: • SSMCIDT[3:0] must be set according to the PLL output clock frequency as shown in Table 6.5.2.1. Using the SSCG with an improper setting may cause a malfunction of the IC.

- When the PLL is off, the initial values and the written values cannot be read correctly from SSMCIDT[3:0] and SSMCITM[3:0] since the source clock is not supplied from the PLL (different values are read out). The correct values can be read out when the PLL is on.

6.6 System Clock Settings

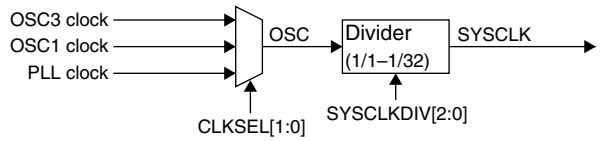


Figure 6.6.1 System Clock Control Circuit

6.6.1 System Clock Source Selection

The system clock source can be selected from OSC3, OSC1, or PLL using CLKSEL[1:0]/CMU_OSCSEL register.

Table 6.6.1.1 System Clock Source Selections

CLKSEL[1:0]	Clock source
0x3	Reserved
0x2	PLL
0x1	OSC1
0x0	OSC3

(Default: 0x0)

The following shows system clock switching procedures:

Switching the system clock to OSC1 from OSC3

1. Turn the OSC1 oscillator on if it is off. (OSC1EN = 1)
2. Wait until the OSC1 oscillation is stabilized.
3. Stop the peripheral circuit operations.
4. Select the OSC1 clock as the system clock. (CLKSEL[1:0] = 0x1)
5. Check if CLKSEL[1:0] is set to 0x1 to confirm that the system clock has been switched to OSC1.
6. Turn the OSC3 oscillator off to reduce current consumption if the CMU_CLK output circuit has not used the OSC3 clock. (OSC3EN = 0)

Switching the system clock to PLL from OSC3

1. Configure the PLL parameters, such as the PLL input clock and the multiplication rate, before activating the PLL.
2. Enable the PLL. (PLLPOWR = 1)
3. Wait until the PLL operation is stabilized.
4. Stop the peripheral circuit operations except the RTC.
5. Select the PLL clock as the system clock. (CLKSEL[1:0] = 0x2)
6. Check if CLKSEL[1:0] is set to 0x2 to confirm that the system clock has been switched to PLL.

Switching the system clock to OSC3 from OSC1

1. Turn the OSC3 oscillator on if it is off. (OSC3EN = 1)
2. Wait until the OSC3 oscillation is stabilized.
3. Stop the peripheral circuit operations except the RTC.
4. Select the OSC3 clock as the system clock. (CLKSEL[1:0] = 0x0)
5. Check if CLKSEL[1:0] is set to 0x0 to confirm that the system clock has been switched to OSC3.

Switching the system clock to PLL from OSC1

1. Switch the system clock to OSC3 from OSC1 by following the procedure shown above.
2. Switch the system clock to PLL from OSC3 by following the procedure shown above.

Switching the system clock to OSC3 from PLL

1. Stop the peripheral circuit operations except the RTC.
2. Select the OSC3 clock as the system clock. (CLKSEL[1:0] = 0x0)
3. Check if CLKSEL[1:0] is set to 0x0 to confirm that the system clock has been switched to OSC3.
4. Disable the PLL to reduce current consumption if the CMU_CLK output circuit has not used the PLL clock. (PLLPOWR = 0)

Switching the system clock to OSC1 from PLL

1. Switch the system clock to OSC3 from PLL by following the procedure shown above.
2. Switch the system clock to OSC1 from OSC3 by following the procedure shown above.

Notes: • Do not select the system clock from deactivated clock sources. It will cause the system to hang as the CMU does not include a protection mechanism against such system clock selection.

• After the system clock is switched, change peripheral circuit settings according to the new clock as necessary.

6.6.2 System Clock Frequency Setting

The source clock frequency can be divided by 1 to 32 to generate the system clock using SYSCLKDIV[2:0]/CMU_SYSCLKDIV register. Setting the system clock to the lowest frequency possible according to the processing can reduce current consumption.

Table 6.6.2.1 System Clock Division Ratio

SYSCLKDIV[2:0]	Division ratio (OSC/n)
0x7–0x6	1/1
0x5	1/32
0x4	1/16
0x3	1/8
0x2	1/4
0x1	1/2
0x0	1/1

(Default: 0x0)

6.7 Clock Supply Control

To reduce current consumption on the chip, the CMU provides some gate circuits to disable clock supply.
The clock supply circuits other than core clock (CCLK) and bus clock (BCLK) provide a clock enable bit. When the clock enable bit is set to 1, the clock is supplied to the peripheral module even in HALT mode as well as normal mode. To stop the peripheral module in HALT mode, set the clock enable bit to 0 before executing the halt instruction. In SLEEP mode (when the slp instruction is executed), the clocks stop even if the clock enable bit is set to 1.

6.7.1 Core Clock (CCLK)

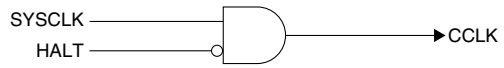


Figure 6.7.1.1 CCLK Control Circuit

The CCLK clock is the C33 PE Core operating clock.
In normal mode, CCLK is always supplied to the C33 PE Core (CPU, CCU, A0RAM, COPRO).
When the C33 PE Core executes the halt or slp instruction, the CMU stops supplying the clock to the C33 PE Core and the C33 PE Core enters a standby (HALT or SLEEP) mode. The CMU resumes the clock supply to the C33 PE Core when the standby mode is canceled by occurrence of an interrupt.

6.7.2 Bus Clock (BCLK)

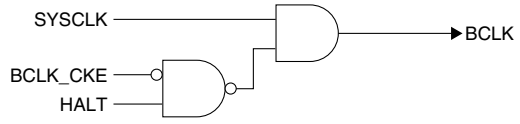


Figure 6.7.2.1 BCLK Control Circuit

The BCLK clock is used to operate the modules listed below.

- IVRAM (Area 3)
- DSTRAM (Area 3)
- SRAM controller (SRAMC) bus interface and registers
- SDRAM controller (SDRAMC) bus interface and registers

- Clock management unit (CMU) registers
- SAPB bus

BCLK is required for bus and memory operations, therefore, it is always supplied to the modules listed above in normal mode.
However, the BCLK supply in HALT mode can be disabled using BCLK_CKE/CMU_CLKCTL2 register.
To stop BCLK in HALT mode, set BCLK_CKE to 0. The CMU stops supplying BCLK when the halt instruction is executed. The CMU resumes the clock supply when the HALT mode is canceled.
To supply BCLK in HALT mode, set BCLK_CKE to 1 (default). The modules listed above operates even in HALT mode.
BCLK stops in SLEEP mode (when the slp instruction is executed) regardless of the BCLK_CKE set value.

6.7.3 Peripheral Module Clocks (PCLK)

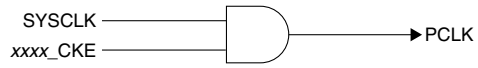


Figure 6.7.3.1 PCLK Control Circuit

The peripheral module clocks (collectively called PCLK) are used to operate the modules listed below.

Table 6.7.3.1 Peripheral Modules and Operating Clocks

PCLK	Clock enable bit	Peripheral modules	Default
T16A_CLK	T16A_CKE/CMU_CLKCTL0 register	• 16-bit PWM timer (T16A6 Ch.0–3) registers • Prescaler (PSC Ch.0) → T16A6 Ch.0–3	Enabled
TCLK *1	TCLK_CKE/CMU_CLKCTL0 register	• 16-bit PWM timer (T16A6 Ch.0–3) counters • Prescaler (PSC Ch.0) → T16A6 Ch.0–3	Enabled
NANDIF_CLK *5	NANDIF_CKE/CMU_CLKCTL1 register	Card interface (CARD)	Enabled
REMC_CLK	REMC_CKE/CMU_CLKCTL1 register	Remote controller (REMC)	Enabled
ADC_CLK	ADC_CKE/CMU_CLKCTL1 register	A/D converter (ADC10)	Enabled
PORT_CLK	PORT_CKE/CMU_CLKCTL1 register	I/O ports and port MUX (GPIO/PMUX)	Enabled
SDMMC_CLK	SDMMC_CKE/CMU_CLKCTL1 register	SD/MMC interface (SD_MMC)	Enabled
I2S_CLK	I2S_CKE/CMU_CLKCTL1 register	I²S (I2S)	Enabled
ITC_CLK	ITC_CKE/CMU_CLKCTL1 register	Interrupt controller (ITC)	Enabled
DMAC_CLK	DMAC_CKE/CMU_CLKCTL1 register	DMA controller (DMAC)	Enabled
HIF_CLK *2	HIF_CKE/CMU_CLKCTL2 register	Host Interface (HIF)	Disabled
LCDC_SAPB_CLK	LCDC_SAPB_CKE/CMU_CLKCTL2 register	LCD controller (LCDC) registers	Disabled
LCDC_AHB_CLK	LCDC_AHB_CKE/CMU_CLKCTL2 register	LCD controller (LCDC) AHB interface and FIFO	Disabled
LCDC_CLK *3	LCDC_CKE/CMU_CLKCTL2 register	LCD controller (LCDC)	Disabled
USBREG_CLK	USBREG_CKE/CMU_CLKCTL2 register	USB function controller (USB) registers	Disabled
USB_CLK *4	USB_CKE/CMU_CLKCTL2 register	USB function controller (USB)	Disabled
BCU_CLK	BCU_CKE/CMU_CLKCTL2 register	• SRAM controller (SRAMC) • SDRAM controller (SDRAMC)	Enabled
PSC1/2_CLK	PSC1/2_CKE/CMU_CLKCTL3 register	• Prescaler (PSC Ch.1) → • T8F Ch.0 → USI Ch.0 • T8F Ch.2 → USI Ch.2 • T8F Ch.4 → ADC10 • ADC10 • Prescaler (PSC Ch.2) → • T8F Ch.1 → USI Ch.1 • T8F Ch.3 → USIL • T8F Ch.5 → UART • UART • REMC • T16P Ch.0–1 • GPIO	Enabled
T8F_CLK	T8F_CKE/CMU_CLKCTL3 register	Fine mode 8-bit timer (T8F Ch.0–5)	Enabled
UART_CLK	UART_CKE/CMU_CLKCTL3 register	UART	Enabled
T16P_CLK	T16P_CKE/CMU_CLKCTL3 register	16-bit audio PWM timer (T16P Ch.0–1)	Enabled
USI_CLK	USI_CKE/CMU_CLKCTL3 register	Universal serial interface (USI Ch.0–2)	Enabled
USIL_CLK	USIL_CKE/CMU_CLKCTL3 register	Universal serial interface with LCD interface (USIL)	Enabled
MISC_CLK	MISC_CKE/CMU_CLKCTL4 register	Misc registers (MISC)	Enabled
IVRAM_ARB_CLK	IVRAM_ARB_CKE/CMU_CLKCTL4 register	IVRAM arbiter (LCDC)	Enabled
A3RAM_CLK	A3RAM_CKE/CMU_CLKCTL4 register	DSTRAM	Enabled
RTC_SAPB_CLK	RTC_SAPB_CKE/CMU_CLKCTL4 register	• Real-time clock (RTC) registers • BBRAM	Enabled
WDT_CLK	WDT_CKE/CMU_CLKCTL4 register	Watchdog timer (WDT)	Enabled

*1 SYSCLK, OSC3, or OSC1 clock (See Section 6.7.4.)
*2 Before the HIF registers (both host side and S1C33L27 side registers) can be read or written, HIF_CLK must be enabled by writing 1 to HIF_CKE/CMU_CLKCTL2 register.

*3 OSC3 dividing clock (See Section 6.7.5.)
*4 OSC3 clock (48 MHz) (See Section 6.7.6.)
*5 NANDIF_CLK must be turned on (NANDIF_CKE = 1) to control the registers and EDC function of the CARD module. When the EDC function is not used (EN/CARD_EDC_CTRL register = 0) or after the card area has been selected using RGN[2:0]/CARD_EDC_RGN register, NANDIF_CLK can be turned off (NANDIF_CKE = 0). The #NAND_RD and #NAND_WR signals can be output normally without NANDIF_CLK.

The peripheral module clock supply can be controlled using the clock enable bit listed above.
The clock supply is enabled by setting the clock enable bit is 1. Disable the clock supply by setting the clock enable bit to 0 to reduce current consumption unless the modules that use the clock need to be running.

6.7.4 16-bit PWM Timer Counter Clock (TCLK)

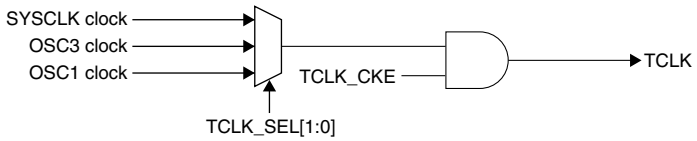


Figure 6.7.4.1 TCLK Control Circuit

The T16A6 counter clock (TCLK) can be generated from the OSC3 and OSC1 clocks as well as SYSCLK.
The source clock is selected using TCLK_SEL[1:0]/CMU_CLKCTL0 register.

Table 6.7.4.1 TCLK Source Clock Selections

TCLK_SEL[1:0]	Source clock
0x3–0x2	OSC1
0x1	OSC3
0x0	SYSCLK

(Default: 0x0)

TCLK_CKE/CMU_CLKCTL0 register is used for clock supply control (default: enabled). Before using the T16A6, make sure that TCLK_CKE is set to 1. Note that PCLK is required to set the T16A6 registers (see Table 6.7.3.1).

Note: Disable TCLK supply (TCLK_CKE = 0) when changing the source clock using TCLK_SEL[1:0].

6.7.5 LCDC Clock (LCDC_CLK)

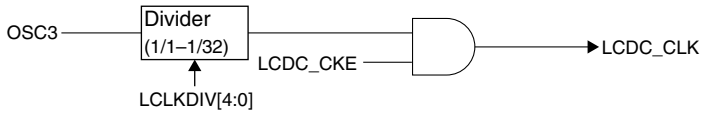


Figure 6.7.5.1 LCDC_CLK Control Circuit

The LCDC_CLK clock is generated by dividing the OSC3 clock and is supplied the LCD controller (LCDC). The frequency divider generates 32 kinds of clocks from OSC3/1 to OSC3/32. Select a division ratio according to the frame rate using LCLKDIV[4:0]/CMU_LCLKDIV register.

$$\text{Frame rate} = \frac{f_{LCDC\_CLK}}{HT \times VT} \text{ [Hz]}$$

f_{LCDC_CLK}: LCDC_CLK frequency
HT: Horizontal total period (horizontal panel size + horizontal non-display period) [pixels]
VT: Vertical total period (vertical panel size + vertical non-display period) [lines]

Table 6.7.5.1 LCDC Clock (OSC3 Division Ratio) Selections

LCLKDIV[4:0]	Division ratio (OSC3/n)	LCLKDIV[4:0]	Division ratio (OSC3/n)
0x1f	1/32	0xf	1/16
0x1e	1/31	0xe	1/15
0x1d	1/30	0xd	1/14
0x1c	1/29	0xc	1/13
0x1b	1/28	0xb	1/12
0x1a	1/27	0xa	1/11
0x19	1/26	0x9	1/10
0x18	1/25	0x8	1/9
0x17	1/24	0x7	1/8
0x16	1/23	0x6	1/7
0x15	1/22	0x5	1/6
0x14	1/21	0x4	1/5
0x13	1/20	0x3	1/4
0x12	1/19	0x2	1/3
0x11	1/18	0x1	1/2
0x10	1/17	0x0	1/1

(Default: 0x7)

LCDC_CKE/CMU_CLKCTL2 register is used for clock supply control (default: disabled). Before using the LCDC, set LCDC_CKE to 1. Note that PCLK is required to set the LCDC registers (see Table 6.7.3.1).

Note: Disable LCDC_CLK supply (LCDC_CKE = 0) when changing the clock division ratio using LCLKDIV[4:0] or before executing the slp instruction.

6.7.6 USB Clock (USB_CLK)

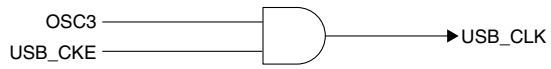


Figure 6.7.6.1 USB_CLK Control Circuit

The USB_CLK clock is the USB operating clock supplied to the USB function controller. Use a 48 MHz ceramic resonator for the OSC3 oscillator circuit when using the USB function. USB_CKE/CMU_CLKCTL2 register is used for clock supply control. The default setting of USB_CKE is 0, which disables the clock supply. Enable the clock supply by setting USB_CKE to 1 before the USB function controller can be used. Note that PCLK is required to set the USB registers (see Table 6.7.3.1).

6.8 Clock External Output (CMU_CLK)

An internally generated clock can be output from the CMU_CLK pin to external devices.
The output clock can be selected from among 11 clocks using CMU_CLKSEL[4:0]/CMU_CMUCLK register.

Table 6.8.1 CMU_CLK Selections

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

(Default: 0x0)

CMU_CLK can be selected at any time. However, switching over the clocks creates hazards.

Note: Settings other than those listed in Table 6.8.1 are reserved for testing. Do not set undescribed values to CMU_CLKSEL[4:0] as undesired clocks may output.

6.9 Standby Modes

The S1C33L27 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.

6.9.1 HALT Mode

The CPU suspends program execution upon executing the halt instruction and enters HALT mode. 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.
In HALT mode, CCLK stops so the CPU, CCU, and A0RAM stop operating. Furthermore, BCLK (bus-related modules) can be stopped in HALT mode (after the halt instruction is executed) by setting BCLK_CKE/CMU_CLKCTL2 register to 0 (see Sections 6.7.2 for BCLK). 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. When an interrupt is used to cancel HALT mode, the C33 PE Core uses the interrupt signal sent from the interrupt controller (ITC). Therefore, the interrupts used to cancel HALT mode must be enabled in the interrupt source modules. The C33 PE Core can restart from HALT mode even if the PSR is set to disable interrupts.
When the IE (interrupt enable) bit in the PSR is set to 1 (enabled), the C33 PE Core executes the interrupt handler routine after HALT mode is canceled. When the IE bit is set to 0 (disabled), an interrupt does not occur and the C33 PE Core resumes execution from the instruction that follows the halt instruction.
The #NMI signal releases the CPU from HALT mode when it goes low level.

6.9.2 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 stops supplying clocks. Therefore, all peripheral modules (except for the OSC1 oscillator circuit and RTC) stop operating.
The CPU is reawaken from SLEEP mode by initial reset, an RTC interrupt, an NMI, other interrupt from an external device (port or key input interrupt), or a forcible break from the debugger.
The C33 PE Core can restart from SLEEP mode even if the PSR is set to disable interrupts. When the IE (interrupt enable) bit in the PSR is set to 0 (disabled), an interrupt does not occur and the C33 PE Core resumes execution from the instruction that follows the slp instruction. When the IE (interrupt enable) bit in the PSR is set to 1 (enabled), the C33 PE Core executes the interrupt handler routine after SLEEP mode is canceled.
The #NMI signal releases the CPU from SLEEP mode when it goes low level.

Notes:
- In SLEEP mode, there is a time lag between inputting an interrupt signal for wake-up and starting the clock supply to the interrupt source module, so a delay will occur until the interrupt flag is set. Therefore, no interrupt will occur if the interrupt signal is negated before the clock is supplied, as the interrupt flag is not set.
  Furthermore, additional time is needed for the C33 PE Core to accept the interrupt request from the ITC, the C33 PE Core may execute a few instructions that follow the slp instruction before it executes the interrupt handler routine.
  When a level trigger port or key input interrupt is used to wake up the C33 PE Core from SLEEP mode, assert the input signal until the clock supply has started. Edge trigger port or key input interrupts can also used to cancel SLEEP mode. The active signal edge will be automatically converted into an active level signal by the GPIO module and it keeps on active until the clock supply has started.
  The same problem may occur when the CPU wakes up from SLEEP mode by an NMI. No interrupt will occur if the #NMI signal is negated before the clock is supplied.
- Before setting the S1C33L27 into SLEEP mode, the clock supply for the LCDC must be disabled.
- If the system clock is PLL, change it to OSC3 or OSC1 before setting SLEEP mode.

6.10 Control Register Details

Table 6.10.1 List of CMU Registers

Address	Register name		Function
0x300100	CMU_OSCSEL	Clock Source Select Register	Select system clock source
0x300101	CMU_OSCCTL	Oscillation Control Register	Control oscillators
0x300103	CMU_LCLKDIV	LCDC Clock Division Ratio Select Register	Set LCDC_CLK frequency
0x300105	CMU_SYSCLKDIV	System Clock Division Ratio Select Register	Set system clock frequency
0x300106	CMU_CMUCLK	CMU_CLK Select Register	Select CMU_CLK output clock
0x300107	CMU_PLLINDIV	PLL Input Clock Division Ratio Select Register	Set PLL input clock frequency
0x300108	CMU_PLLCTL0	PLL Control Register 0	Set PLL multiplication rate and enable PLL
0x300109	CMU_PLLCTL1	PLL Control Register 1	Set PLL parameters
0x30010a	CMU_PLLCTL2	PLL Control Register 2	
0x30010c	CMU_SSCG0	SSCG Macro Control Register 0	Enable SSCG
0x30010d	CMU_SSCG1	SSCG Macro Control Register 1	Set SSCG parameters
0x300110	CMU_PROTECT	CMU Write Protect Register	Enable/disable CMU register write protection
0x300114	CMU_CLKCTL0	Clock Control Register 0	Control clock supply to peripheral/bus modules
0x300115	CMU_CLKCTL1	Clock Control Register 1	
0x300116	CMU_CLKCTL2	Clock Control Register 2	
0x300117	CMU_CLKCTL3	Clock Control Register 3	
0x300118	CMU_CLKCTL4	Clock Control Register 4	

The CMU module registers are described in detail below.

Notes: • When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

• The CMU control registers (addresses 0x300100 to 0x30010d and 0x300114 to 0x300118) are write-protected. Before the CMU control registers can be rewritten, write protection of these registers must be removed by writing data 0x96 to CMUP[7:0]/CMU_PROTECT register. Note that since unnecessary rewrites to the CMU control registers could lead to erratic system operation, CMUP[7:0] should be set to other than 0x96 unless the CMU control registers must be rewritten.

Clock Source Select Register (CMU_OSCSEL)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Clock Source Select Register (CMU_OSCSEL)	**0x300100** (8 bits)	D7–2	–	reserved	–		–	–	0 when being read.
		D1–0	**CLKSEL [1:0]**	System clock source select	CLKSEL[1:0]	Clock source	0x0	R/W	Write-protected
					0x3	Not allowed			
					0x2	PLL			
					0x1	OSC1			
					0x0	OSC3			

D[7:2] Reserved

D[1:0] CLKSEL[1:0]: System Clock Source Select Bits

Selects the system clock source.

Table 6.10.2 System Clock Source Selections

CLKSEL[1:0]	Clock source
0x3	Reserved
0x2	PLL
0x1	OSC1
0x0	OSC3

(Default: 0x0)

Note: Do not select the system clock from deactivated clock sources. It will cause the system to hang as the CMU does not include a protection mechanism against such system clock selection.

Oscillation Control Register (CMU_OSCCTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Oscillation Control Register (CMU_OSCCTL)	0x300101 (8 bits)	D7–4	OSC3WT[3:0]	OSC3 wait cycle select	OSC3WT[3:0] / Wait cycle: 0xf 128 cycles 0xe 256 cycles 0xd 512 cycles 0xc 1,024 cycles 0xb 2,048 cycles 0xa 4,096 cycles 0x9 8,192 cycles 0x8 16,384 cycles 0x7 32,768 cycles 0x6 65,536 cycles 0x5 131,072 cycles 0x4 262,144 cycles 0x3 524,288 cycles 0x2 1,048,576 cycles 0x1 2,097,152 cycles 0x0 4,194,304 cycles	0xf	R/W	Write-protected
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	OSC1EN	OSC1 enable	1 Enable 0 Disable	1	R/W	Write-protected
		D0	OSC3EN	OSC3 enable	1 Enable 0 Disable	1	R/W	

D[7:4] OSC3WT[3:0]: OSC3 Wait Cycle Select Bits

An oscillation stabilization wait timer is set to prevent malfunctions due to unstable clock operation at the start of OSC3 oscillation. The OSC3 or PLL clock is not supplied to the system immediately after OSC3 oscillation starts—e.g., after an initial reset or when waking from SLEEP—until the time set here has elapsed.

Table 6.10.3 OSC3 Oscillation Stabilization Wait Time Settings

OSC3WT[3:0]	Oscillation stabilization wait time
0xf	128 cycles
0xe	256 cycles
0xd	512 cycles
0xc	1,024 cycles
0xb	2,048 cycles
0xa	4,096 cycles
0x9	8,192 cycles
0x8	16,384 cycles
0x7	32,768 cycles
0x6	65,536 cycles
0x5	131,072 cycles
0x4	262,144 cycles
0x3	524,288 cycles
0x2	1,048,576 cycles
0x1	2,097,152 cycles
0x0	4,194,304 cycles

(Default: 0xf)

This is set to 128 cycles (OSC3 clock) after an initial reset.

Notes: • The OSC3 oscillation stabilization wait timer cannot be used when the OSC3 oscillator is turned on with software. Therefore, a software wait routine must be implemented.

- Oscillation stability will vary, depending on the resonator and other external components. Carefully consider the OSC3 oscillation stabilization wait time before reducing the time. When waking from SLEEP mode if OSC3 or PLL is used as the system clock source, set the OSC3 oscillation stabilization wait time as follows:

  OSC3 oscillation stabilization wait time [cycle] ≥ OSC3 oscillation start time [s] (max.) × f_{SYSCLK} [Hz]

  f_{SYSCLK}: SYSCLK frequency when the clock source is OSC3 or PLL.

  Example: When OSC3 oscillation start time (max.) = 25 ms and f_{SYSCLK} = 48 MHz
  OSC3 oscillation stabilization wait time ≥ 1,200,000 [cycles]

  OSC3WT[3:0] should be set to 0x1 (OSC3 oscillation stabilization wait time = 2,097,152 cycles).

D[3:2] Reserved

D1 OSC1EN: OSC1 Enable Bit

Enables or disables OSC1 oscillator operations.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

Note: Do not stop the OSC1 oscillator if the OSC1 clock is being used as the system clock.

D0 OSC3EN: OSC3 Enable Bit

Enables or disables OSC3 oscillator operations.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

Note: The OSC3 oscillator cannot be stopped if the OSC3 or PLL clock is being used as the system clock.

LCDC Clock Division Ratio Select Register (CMU_LCLKDIV)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
LCDC Clock Division Ratio Select Register (CMU_LCLKDIV)	0x300103 (8 bits)	D7–5	–	reserved	–		–	–	0 when being read.
		D4–0	LCLKDIV[4:0]	LCDC clock division ratio select	LCLKDIV[4:0]	Division ratio	0x7	R/W	Clock source = OSC3 Write-protected
					0x1f	1/32			
					0x1e	1/31			
					0x1d	1/30			
					0x1c	1/29			
					0x1b	1/28			
					0x1a	1/27			
					0x19	1/26			
					0x18	1/25			
					0x17	1/24			
					0x16	1/23			
					0x15	1/22			
					0x14	1/21			
					0x13	1/20			
					0x12	1/19			
					0x11	1/18			
					0x10	1/17			
					0xf	1/16			
					0xe	1/15			
					0xd	1/14			
					0xc	1/13			
					0xb	1/12			
					0xa	1/11			
					0x9	1/10			
					0x8	1/9			
					0x7	1/8			
					0x6	1/7			
					0x5	1/6			
					0x4	1/5			
					0x3	1/4			
					0x2	1/3			
					0x1	1/2			
					0x0	1/1			

D[7:5] Reserved

D[4:0] LCLKDIV[4:0]: LCDC Clock Division Ratio Select Bits

Selects the LCDC clock (LCDC_CLK) from among 32 kinds of OSC3 division clocks. Select a clock according to the frame rate.

$$\text{Frame rate} = \frac{f_{LCDC\_CLK}}{HT \times VT} \text{ [Hz]}$$

f_{LCDC_CLK}: LCDC_CLK frequency
HT: Horizontal total period (horizontal panel size + horizontal non-display period) [pixels]
VT: Vertical total period (vertical panel size + vertical non-display period) [lines]

Table 6.10.4 LCDC Clock (OSC3 Division Ratio) Selections

LCLKDIV[4:0]	Division ratio (OSC3/n)	LCLKDIV[4:0]	Division ratio (OSC3/n)
0x1f	1/32	0xf	1/16
0x1e	1/31	0xe	1/15
0x1d	1/30	0xd	1/14
0x1c	1/29	0xc	1/13
0x1b	1/28	0xb	1/12
0x1a	1/27	0xa	1/11
0x19	1/26	0x9	1/10
0x18	1/25	0x8	1/9
0x17	1/24	0x7	1/8
0x16	1/23	0x6	1/7
0x15	1/22	0x5	1/6
0x14	1/21	0x4	1/5
0x13	1/20	0x3	1/4
0x12	1/19	0x2	1/3
0x11	1/18	0x1	1/2
0x10	1/17	0x0	1/1

(Default: 0x7)

System Clock Division Ratio Select Register (CMU_SYSCLKDIV)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
System Clock Division Ratio Select Register (CMU_SYSCLKDIV)	0x300105 (8 bits)	D7–3	–	reserved	–		–	–	0 when being read.
		D2–0	SYSCLKDIV [2:0]	System clock division ratio select	SYSCLKDIV[2:0]	Division ratio	0x0	R/W	Clock source = OSC (OSC3, PLL, or OSC1) Write-protected
					0x7–0x6	1/1			
					0x5	1/32			
					0x4	1/16			
					0x3	1/8			
					0x2	1/4			
					0x1	1/2			
					0x0	1/1			

D[7:3] Reserved

D[2:0] SYSCLKDIV[2:0]: System Clock Division Ratio Select Bits

Selects a division ratio to set the system clock frequency. To reduce current consumption, operate the C33 PE Core and peripheral modules using the slowest possible clock speed.

Table 6.10.5 System Clock Division Ratio

SYSCLKDIV[2:0]	Division ratio (OSC/n)
0x7–0x6	1/1
0x5	1/32
0x4	1/16
0x3	1/8
0x2	1/4
0x1	1/2
0x0	1/1

(Default: 0x0)

CMU_CLK Select Register (CMU_CMUCLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
CMU_CLK Select Register (CMU_CMUCLK)	0x300106 (8 bits)	D7–5	–	reserved	–		–	–	0 when being read.
		D4–0	CMU_ CLKSEL[4:0]	CMU_CLK select	CMU_CLKSEL[4:0]	CMU_CLK	0x0	R/W	OSC: system clock (OSC3, PLL, OSC1) Write-protected
					0xf–0xb	reserved			
					0xa	OSC/32			
					0x9	OSC/16			
					0x8	OSC/8			
					0x7	OSC/4			
					0x6	OSC/2			
					0x5	OSC/1			
					0x4	LCDC_CLK			
					0x3	BCLK			
					0x2	PLL			
					0x1	OSC1			
					0x0	OSC3			

D[7:5] Reserved

D[4:0] CMU_CLKSEL[4:0]: CMU_CLK Select Bits

Selects an internally generated clock to be output from the CMU_CLK pin to external devices.

Table 6.10.6 CMU_CLK Selections

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

(Default: 0x0)

CMU_CLK can be selected at any time. However, switching over the clocks creates hazards.

Note: Settings other than those listed in Table 6.10.6 are reserved for testing. Do not set undescribed values to CMU_CLKSEL[4:0] as undesired clocks may output.

PLL Input Clock Division Ratio Select Register (CMU_PLLINDIV)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PLL Input Clock Division Ratio Select Register (CMU_PLLINDIV)	0x300107 (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3–0	PLLINDIV [3:0]	PLL input clock division ratio select	PLLINDIV[3:0]	Division ratio	0x7	R/W	Clock source = OSC3 Write-protected
					0xf–0xa	1/8			
					0x9	1/10			
					0x8	1/9			
					0x7	1/8			
					0x6	1/7			
					0x5	1/6			
					0x4	1/5			
					0x3	1/4			
					0x2	1/3			
					0x1	1/2			
					0x0	1/1			

D[7:4] Reserved

D[3:0] PLLINDIV[3:0]: PLL Input Clock Division Ratio Select Bits

Selects the PLL input clock (OSC3 division ratio).

Table 6.10.7 PLL Input Clock (OSC3 Division Ratio) Selections

PLLINDIV[3:0]	Division ratio (OSC3/n)
0xf–0xa	1/8
0x9	1/10
0x8	1/9
0x7	1/8
0x6	1/7
0x5	1/6
0x4	1/5
0x3	1/4
0x2	1/3
0x1	1/2
0x0	1/1

(Default: 0x7)

Notes: • The PLL input clock can only be selected when the PLL is turned off (PLLPOWR/CMU_PLLCTL0 register = 0) and the clock source is other than the PLL (CLKSEL[1:0]/CMU_OSCSEL register is not 2). If the PLL input clock is changed while the system is operating with the PLL clock, the system may operate erratically.

• For the range of the input clock frequency, see "Electrical Characteristics."

PLL Control Register 0 (CMU_PLLCTL0)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PLL Control Register 0 (CMU_PLLCTL0)	**0x300108** (8 bits)	D7–4	**PLLN[3:0]**	PLL multiplication rate setup	PLLN[3:0]	Multiplication rate	0x0	R/W	Write-protected
					0xf	x16			
					0xe	x15			
					0xd	x14			
					0xc	x13			
					0xb	x12			
					0xa	x11			
					0x9	x10			
					0x8	x9			
					0x7	x8			
					0x6	x7			
					0x5	x6			
					0x4	x5			
					0x3	x4			
					0x2	x3			
					0x1	x2			
					0x0	x1			
		D3–2	**PLLV[1:0]**	PLL V-divider setup	PLLV[1:0]	W	0x1	R/W	
					0x3	8			
					0x2	4			
					0x1	2			
					0x0	Not allowed			
		D1	**–**	reserved	–		–	–	0 when being read.
		D0	**PLLPOWR**	PLL enable	1 Enable	0 Disable	0	R/W	Write-protected

Note: Make sure that the PLL is turned off (PLLPOWR = 0) before altering D[7:4] in this register.

D[7:4] PLLN[3:0]: PLL Multiplication Rate Setup Bits

Sets the frequency multiplication rate of the PLL.

Table 6.10.8 PLL Frequency Multiplication Rates

PLLN[3:0]	Multiplication rate
0xf	x16
0xe	x15
0xd	x14
0xc	x13
0xb	x12
0xa	x11
0x9	x10
0x8	x9
0x7	x8
0x6	x7
0x5	x6
0x4	x5
0x3	x4
0x2	x3
0x1	x2
0x0	x1

(Default: 0x0)

PLL output clock frequency = PLL input clock frequency × multiplication rate

Note: The frequency multiplication rate must be set so that the PLL output clock frequency does not exceed the upper-limit operating clock frequency. For the multiplication rates that can be set and the range of the output clock frequency, see "Electrical Characteristics."

D[3:2] PLLV[1:0]: PLL V-Divider Setup Bits

Sets the W value so that the fvco frequency obtained by <Output clock frequency × W> falls within the range of 100 to 400 MHz.

Table 6.10.9 W Value Settings

PLLV[1:0]	W
0x3	8
0x2	4
0x1	2
0x0	Setting prohibited

(Default: 0x1)

D1 Reserved

D0 PLLPOWR: PLL Enable Bit

Turns the PLL on or off.
1 (R/W): On
0 (R/W): Off (Default)

Up to 200 µs is required before the PLL output clock stabilizes after PLLPOWR is set to 1. Specify this wait time in the program before changing the system clock source to the PLL.
When not using the PLL, turn the PLL off (power-down mode) to reduce current consumption.

PLL Control Register 1 (CMU_PLLCTL1)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PLL Control Register 1 (CMU_PLLCTL1)	0x300109 (8 bits)	D7–4	PLLVC[3:0]	PLL VCO Kv setup	PLLVC[3:0]	f_{VCO} [MHz]	0x1	R/W	Write-protected
					0x8	360 < f_{VCO} ≤ 400			
					0x7	320 < f_{VCO} ≤ 360			
					0x6	280 < f_{VCO} ≤ 320			
					0x5	240 < f_{VCO} ≤ 280			
					0x4	200 < f_{VCO} ≤ 240			
					0x3	160 < f_{VCO} ≤ 200			
					0x2	120 < f_{VCO} ≤ 160			
					0x1	100 ≤ f_{VCO} ≤ 120			
					Other	Not allowed			
		D3–0	PLLRS[3:0]	PLL LPF resistance setup	PLLRS[3:0]	f_{REFCK} [MHz]	0x8	R/W	
					0xa	5 ≤ f_{REFCK} < 20			
					0x8	20 ≤ f_{REFCK} ≤ 150			
					Other	Not allowed			

Note: Make sure that the PLL is turned off (PLLPOWR/CMU_PLLCTL0 register = 0) before altering this register.

D[7:4] PLLVC[3:0]: PLL VCO Kv Setup Bits

Sets the VCO Kv circuit constant (VC value) according to the range of fvco frequencies obtained by <Output clock frequency × W>.

Table 6.10.10 VC Value Settings

PLLVC[3:0]	f_{VCO} [MHz]
0x8	360 < f_{VCO} ≤ 400
0x7	320 < f_{VCO} ≤ 360
0x6	280 < f_{VCO} ≤ 320
0x5	240 < f_{VCO} ≤ 280
0x4	200 < f_{VCO} ≤ 240
0x3	160 < f_{VCO} ≤ 200
0x2	120 < f_{VCO} ≤ 160
0x1	100 ≤ f_{VCO} ≤ 120
Other	Setting prohibited

(Default: 0x1)

D[3:0] PLLRS[3:0]: PLL LPF Resistance Setup Bits

Sets the LPF resistance value of the PLL (RS value) according to the input clock (OSC3) frequency.

Table 6.10.11 RS Value Settings

PLLRS[3:0]	f_{REFCK} [MHz]
0xa	5 ≤ f_{REFCK} < 20
0x8	20 ≤ f_{REFCK} ≤ 150
Other	Setting prohibited

(Default: 0x8)

PLL Control Register 2 (CMU_PLLCTL2)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PLL Control Register 2 (CMU_PLLCTL2)	0x30010a (8 bits)	D7–6	PLLCS[1:0]	PLL LPF capacitance	0x0	0x0	R	
		D5	PLLBYP	PLL bypass mode	0	0	R	
		D4–0	PLLCP[4:0]	PLL charge pump current	0x10	0x10	R	

D[7:6] PLLCS[1:0]: PLL LPF Capacitance Bits

Indicates the LPF capacitance value (CS value). (Default: 0x0)

D5 PLLBYP: PLL Bypass Mode Bit

Indicates the mode when the PLL is bypassed. (Default: 0)

D[4:0] PLLCP[4:0]: PLL Charge Pump Current Bits

Indicates the charge pump current value (CP value). (Default: 0x10)

SSCG Macro Control Register 0 (CMU_SSCG0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SSCG Macro Control Register 0 (CMU_SSCG0)	0x30010c (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SSMCON	SSCG enable	1 Enable 0 Disable	0	R/W	Write-protected

D[7:1] Reserved

D0 SSMCON: SSCG Enable Bit

Turns the SSCG on or off.

1 (R/W): On

0 (R/W): Off (Default)

Setting this bit to 1 causes the SSCG to start operating. Setting this bit to 0 causes the SSCG to stop, allowing the clock to bypasses the SSCG.

SSCG Macro Control Register 1 (CMU_SSCG1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SSCG Macro Control Register 1 (CMU_SSCG1)	0x30010d (8 bits)	D7–4	SSMCITM [3:0]	SSCG interval timer (ITM) setting	0x0 to 0xf	0xf	R/W	Write-protected
		D3–0	SSMCIDT [3:0]	SSCG maximum frequency change width setting	0x0 to 0xf	0x0	R/W	

Note: When the PLL is off, the initial values and the written values cannot be read correctly from this register since the source clock is not supplied to the SSCG (different values are read out). The correct values can be read out when the PLL is turned on.

D[7:4] SSMCITM[3:0]: SSCG Interval Timer Setting Bits

Sets the frequency change cycle in SS modulation of the SSCG. (See Section 6.5, “SSCG.”)

Always set these bits to 0x1 (Default: 0xf)

D[3:0] SSMCIDT[3:0]: SSCG Maximum Frequency Change Width Setting Bits

Sets the maximum frequency change width in SS modulation of the SSCG. (See Section 6.5, “SSCG.”)

Table 6.10.12 Maximum Frequency Change Width Settings

PLL output clock frequency f [MHz]	SSMCIDT[3:0]
f ≤ 19.8	0xf
19.8 < f ≤ 21.2	0xe
21.2 < f ≤ 22.5	0xd
22.5 < f ≤ 24.2	0xc
24.2 < f ≤ 25.9	0xb
25.9 < f ≤ 28.4	0xa
28.4 < f ≤ 30.8	0x9
30.8 < f ≤ 34.2	0x8
34.2 < f ≤ 37.8	0x7
37.8 < f ≤ 43.1	0x6
43.1 < f ≤ 48.9	0x5
48.9 < f ≤ 58.5	0x4
58.5 < f ≤ 60.0	0x3
–	0x2
–	0x1
–	0x0

(Default: 0x0)

Note: SSMCIDT[3:0] must be set according to the PLL output clock frequency as shown in Table 6.10.12. Using the SSCG with an improper setting may cause a malfunction of the IC.

CMU Write Protect Register (CMU_PROTECT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CMU Write Protect Register (CMU_ PROTECT)	**0x300110** (8 bits)	D7–0	**CMUP[7:0]**	CMU register write-protect flag	Writing 10010110 (0x96) removes the write protection of the CMU registers (0x300100–0x30010d, 0x300114–0x300118). Writing another value set the write protection.	0x0	R/W	

D[7:0] CMUP[7:0]: CMU Register Write-Protect Flag Bits

Enables or disables write protection of the CMU control registers (0x300100–0x30010d, 0x300114–0x300118).

0x96 (R/W): Disable write protection

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

Before altering any CMU control register, write data 0x96 to CMUP[7:0] to disable write protection. If CMUP[7:0] is set to other than 0x96, even if an attempt is made to alter any CMU control register by executing a write instruction, the content of the register will not be altered even though the instruction may have been executed without a problem. Once CMUP[7:0] is set to 0x96, the CMU control registers can be rewritten any number of times until being reset to other than 0x96. When rewriting the CMU control registers has finished, CMUP[7:0] should be set to other than 0x96 to prevent accidental writing to the CMU registers.

Clock Control Register 0 (CMU_CLKCTL0)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Clock Control Register 0 (CMU_ CLKCTL0)	**0x300114** (8 bits)	D7–4	**–**	reserved	–				–	–	0 when being read.
		D3	**T16A_CKE**	T16A6 register clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D2	**TCLK_CKE**	T16A6 counter clock enable	1	Enable	0	Disable	1	R/W	
		D1–0	**TCLK_SEL [1:0]**	T16A6 counter clock source select	TCLK_SEL[1:0]		Clock source		0x0	R/W	
					0x3–0x2		OSC1				
					0x1		OSC3				
					0x0		SYSCLK				

This register is used to control the clock supply to the peripheral modules. If no peripheral module operation is required, stop the clock supply to reduce current consumption.

D[7:4] Reserved

D3 T16A_CKE: T16A6 Register Clock Enable Bit

Enables or disables the T16A6 register clock (PCLK) supply to the T16A6 module and the prescaler Ch.0 (PSC_CTL0) register clock (PCLK) supply to the PSC module (PSC Ch.0).

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D2 TCLK_CKE: T16A6 Counter Clock Enable Bit

Enables or disables the T16A6 counter clock (TCLK) supply to the T16A6 module and the prescaler Ch.0 divider clock (TCLK) supply to the PSC module (PSC Ch.0).

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D[1:0] TCLK_SEL[1:0]: T16A6 Counter Clock Source Select Bits

Selects the source clock for generating the T16A6 counter clock (TCLK) and the prescaler Ch.0 divider clock (TCLK).

Table 6.10.13 TCLK Source Clock Selections

TCLK_SEL[1:0]	Source clock
0x3–0x2	OSC1
0x1	OSC3
0x0	SYSCLK

(Default: 0x0)

Clock Control Register 1 (CMU_CLKCTL1)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Clock Control Register 1 (CMU_CLKCTL1)	0x300115 (8 bits)	D7	NANDIF_CKE	CARD clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D6	REMC_CKE	REMC clock enable	1	Enable	0	Disable	1	R/W	
		D5	ADC_CKE	ADC10 clock enable	1	Enable	0	Disable	1	R/W	
		D4	PORT_CKE	GPIO/PMUX clock enable	1	Enable	0	Disable	1	R/W	
		D3	SDMMC_CKE	SD_MMC clock enable	1	Enable	0	Disable	1	R/W	
		D2	I2S_CKE	I2S clock enable	1	Enable	0	Disable	1	R/W	
		D1	ITC_CKE	ITC clock enable	1	Enable	0	Disable	1	R/W	
		D0	DMAC_CKE	DMAC clock enable	1	Enable	0	Disable	1	R/W	

This register is used to control the clock supply to the peripheral modules. If no peripheral module operation is required, stop the clock supply to reduce current consumption.

D7 NANDIF_CKE: CARD Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the CARD module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

Note: NANDIF_CLK must be turned on (NANDIF_CKE = 1) to control the registers and EDC function of the CARD module. When the EDC function is not used (EN/CARD_EDC_CTRL register = 0) or after the card area has been selected using RGN[2:0]/CARD_EDC_RGN register, NANDIF_CLK can be turned off (NANDIF_CKE = 0). The #NAND_RD and #NAND_WR signals can be output normally without NANDIF_CLK.

D6 REMC_CKE: REMC Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the REMC module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D5 ADC_CKE: ADC10 Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the ADC10 module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D4 PORT_CKE: GPIO/PMUX Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the GPIO/PMUX module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D3 SDMMC_CKE: SD_MMC Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the SD_MMC module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D2 I2S_CKE: I2S Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the I2S module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D1 ITC_CKE: ITC Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the ITC module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

D0 DMAC_CKE: DMAC Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the DMAC module.

1 (R/W): Enabled (on) (default)

0 (R/W): Disabled (off)

Clock Control Register 2 (CMU_CLKCTL2)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Clock Control Register 2 (CMU_CLKCTL2)	0x300116 (8 bits)	D7	HIF_CKE	HIF clock enable	1	Enable	0	Disable	0	R/W	Write-protected
		D6	LCDC_SAPB_CKE	LCDC SAPB I/F clock enable	1	Enable	0	Disable	0	R/W	
		D5	LCDC_AHB_CKE	LCDC AHB I/F clock enable	1	Enable	0	Disable	0	R/W	
		D4	LCDC_CKE	LCDC clock enable	1	Enable	0	Disable	0	R/W	
		D3	USBREG_CKE	USB register clock enable	1	Enable	0	Disable	0	R/W	
		D2	USB_CKE	USB clock enable	1	Enable	0	Disable	0	R/W	
		D1	BCU_CKE	SRAMC/SDRAMC clock enable	1	Enable	0	Disable	1	R/W	
		D0	BCLK_CKE	BCLK clock enable (in HALT mode)	1	Enable	0	Disable	1	R/W	

This register is used to control the clock supply to the peripheral modules. If no peripheral module operation is required, stop the clock supply to reduce current consumption.

D7 HIF_CKE: HIF Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the HIF module.

1 (R/W): Enabled (on)

0 (R/W): Disabled (off) (default)

Note: Before the HIF registers (both host side and S1C33L27 side registers) can be read or written, HIF_CLK must be enabled by writing 1 to HIF_CKE.

D6 LCDC_SAPB_CKE: LCDC SAPB I/F Clock Enable Bit

Enables or disables the LCDC register clock (PCLK) supply to the LCDC module.

1 (R/W): Enabled (on)

0 (R/W): Disabled (off) (default)

D5 LCDC_AHB_CKE: LCDC AHB I/F Clock Enable Bit

Enables or disables the AHB interface clock (PCLK) supply to the LCDC module.

1 (R/W): Enabled (on)

0 (R/W): Disabled (off) (default)

D4 LCDC_CKE: LCDC Clock Enable Bit

Enables or disables the operating clock (LCDC_CLK) supply to the LCDC module.
1 (R/W): Enabled (on)
0 (R/W): Disabled (off) (default)

D3 USBREG_CKE: USB Register Clock Enable Bit

Enables or disables the USB register clock (PCLK) supply to the USB module.
1 (R/W): Enabled (on)
0 (R/W): Disabled (off) (default)

D2 USB_CKE: USB Clock Enable Bit

Enables or disables the operating clock (OSC3) supply to the USB module.
1 (R/W): Enabled (on)
0 (R/W): Disabled (off) (default)

D1 BCU_CKE: SRAMC/SDRAMC Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the SRAMC and SDRAMC modules.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D0 BCLK_CKE: BCLK Clock Enable (in HALT mode) Bit

Enables or disables the BCLK clock supply in HALT mode.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

The BCLK clock is used to operate the modules listed below.
- IVRAM (Area 3)
- DSTRAM (Area 3)
- SRAM controller (SRAMC) bus interface and registers
- SDRAM controller (SDRAMC) bus interface and registers
- Clock management unit (CMU) registers
- SAPB bus

BCLK is required for bus and memory operations, therefore, it is always supplied to the modules listed above in normal mode. However, the BCLK supply in HALT mode can be disabled to reduce current consumption by setting BCLK_CKE to 0.

Clock Control Register 3 (CMU_CLKCTL3)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Clock Control Register 3 (CMU_CLKCTL3)	**0x300117** (8 bits)	D7	**PSC1/2_CKE**	PSC1/2 clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D6	**T8F_CKE**	T8F clock enable	1	Enable	0	Disable	1	R/W	
		D5	**UART_CKE**	UART clock enable	1	Enable	0	Disable	1	R/W	
		D4	**T16P_CKE**	T16P clock enable	1	Enable	0	Disable	1	R/W	
		D3–2	**–**	reserved	–				–	–	0 when being read.
		D1	**USI_CKE**	USI clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D0	**USIL_CKE**	USIL clock enable	1	Enable	0	Disable	1	R/W	

This register is used to control the clock supply to the peripheral modules. If no peripheral module operation is required, stop the clock supply to reduce current consumption.

D7 PSC1/2_CKE: PSC1/2 Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to PSC Ch.1 and Ch.2 in the PSC module. The clock is supplied to the frequency divider and control register (PSC_CTL12) of PSC Ch.1 and Ch.2.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D6 T8F_CKE: T8F Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the T8F module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D5 UART_CKE: UART Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the UART module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D4 T16P_CKE: T16P Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the T16P module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D[3:2] Reserved

D1 USI_CKE: USI Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the USI module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D0 USIL_CKE: USIL Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the USIL module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

Clock Control Register 4 (CMU_CLKCTL4)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Clock Control Register 4 (CMU_CLKCTL4)	0x300118 (8 bits)	D7–5	–	reserved	–				–	–	0 when being read.
		D4	MISC_CKE	MISC clock enable	1	Enable	0	Disable	1	R/W	
		D3	IVRAM_ARB_CKE	IVRAM arbiter clock enable	1	Enable	0	Disable	1	R/W	
		D2	A3RAM_CKE	Area 3 RAM clock enable	1	Enable	0	Disable	1	R/W	
		D1	RTC_SAPB_CKE	RTC SAPB I/F clock enable	1	Enable	0	Disable	1	R/W	
		D0	WDT_CKE	WDT clock enable	1	Enable	0	Disable	1	R/W	

This register is used to control the clock supply to the peripheral modules. If no peripheral module operation is required, stop the clock supply to reduce current consumption.

D[7:5] Reserved

D4 MISC_CKE: MISC Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the MISC module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D3 IVRAM_ARB_CKE: IVRAM Arbiter Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the IVRAM arbiter.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D2 A3RAM_CKE: Area 3 RAM Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to DSTRAM.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D1 RTC_SAPB_CKE: RTC SAPB I/F Clock Enable Bit

Enables or disables the RTC register clock (PCLK) supply to the RTC module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

D0 WDT_CKE: WDT Clock Enable Bit

Enables or disables the operating clock (PCLK) supply to the WDT module.
1 (R/W): Enabled (on) (default)
0 (R/W): Disabled (off)

7 Prescaler (PSC)

7.1 PSC Module Overview

The S1C33L27 incorporates a prescaler (PSC) module to generate clocks for timer and serial interface operations. The PSC module consists of three frequency dividers (PSC Ch.0 to Ch.2) that generate 15 different frequencies by dividing the TCLK or PCLK clock supplied from the clock management unit (CMU) into 1/1 to 1/16K. The peripheral modules to which the clock is supplied include clock-select registers enabling selection of one as a count or operation clock.

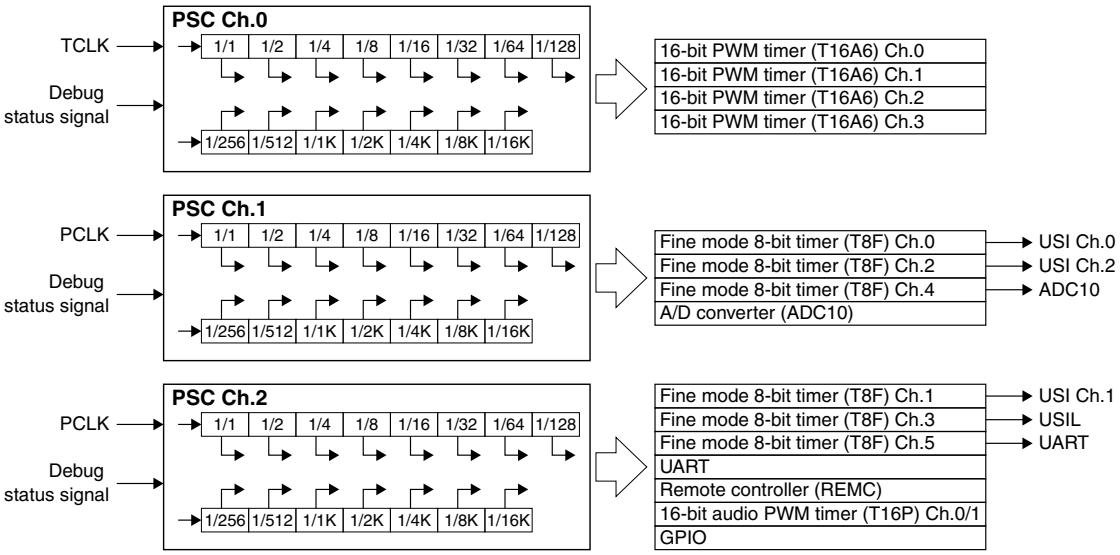


Figure 7.1.1 Prescaler Configuration

PSC Ch.0 is controlled by PRUN/PSC_CTL0 register. PSC Ch.1 and PSC Ch.2 are controlled by PRUN/PSC_CTL12 register. To operate the prescalers, write 1 to PRUN. Writing 0 to PRUN stops the prescalers. Stopping the prescalers while the timer and interface modules are halted enables the current consumption to be reduced. The prescalers are stopped at initial reset.

Note: TCLK and PCLK must be supplied from the CMU to use the PSC Ch.0 and PSC Ch.1/2, respectively.

The PSC module features another control bits, PRUND/PSC_CTL0 register for Ch.0 and PRUND/PSC_CTL12 register for Ch.1/2, that specify prescaler operations in debug mode. Setting PRUND to 1 operates the prescalers in debug mode. Setting it to 0 stops the prescalers when the C33 PE Core enters debug mode. Set PRUND to 1 when operating the timer and interface modules during debugging.

7.2 Control Register Details

Table 7.2.1 PSC Registers

Address	Register name		Function
0x300e00	PSC_CTL0	PSC Ch.0 Control Register	Control prescaler Ch.0
0x300e01	PSC_CTL12	PSC Ch.1/2 Control Register	Control prescaler Ch.1 and 2

The PSC module registers are described in detail below.

Note: When data is written to the register, the “Reserved” bits must always be written as 0 and not 1.

PSC Ch.0 Control Register (PSC_CTL0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PSC0 Control Register (PSC_CTL0)	0x300e00 (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 prescaler operations in debug mode.
1 (R/W): Run
0 (R/W): Stop (default)

Setting PRUND to 1 enables PSC Ch.0 to operate in debug mode. Setting it to 0 stops PSC Ch.0 when the C33 PE Core enters debug mode. Set PRUND to 1 to use the 16-bit PWM timer (T16A6) module during debugging.

D0 PRUN: Prescaler Run/Stop Control Bit

Starts or stops the prescaler.
1 (R/W): Start operation
0 (R/W): Stop (default)

Write 1 to PRUN to operate PSC Ch.0. Write 0 to PRUN to stop PSC Ch.0. To reduce current consumption, stop PSC Ch.0 if the 16-bit PWM timer (T16A6) module is already stopped.

PSC Ch.1/2 Control Register (PSC_CTL12)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PSC1/2 Control Register (PSC_CTL12)	0x300e01 (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 prescaler operations in debug mode.
1 (R/W): Run
0 (R/W): Stop (default)

Setting PRUND to 1 enables PSC Ch.1 and Ch.2 to operate in debug mode. Setting it to 0 stops PSC Ch.1 and Ch.2 when the C33 PE Core enters debug mode. Set PRUND to 1 to use the modules listed below during debugging.

D0 PRUN: Prescaler Run/Stop Control Bit

Starts or stops the prescaler.
1 (R/W): Start operation
0 (R/W): Stop (default)

Write 1 to PRUN to operate PSC Ch.1 and Ch.2. Write 0 to PRUN to stop PSC Ch.1 and Ch.2. To reduce current consumption, stop PSC Ch.1 and Ch.2 if the modules listed below are already stopped.

Modules that use PSC Ch.1 output clocks

- Fine mode 8-bit timer (T8F) Ch.0 (clock source for USI Ch.0)
- Fine mode 8-bit timer (T8F) Ch.2 (clock source for USI Ch.2)
- Fine mode 8-bit timer (T8F) Ch.4 (trigger source for A/D converter)
- A/D converter (ADC10)

Modules that use PSC Ch.2 output clocks

- Fine mode 8-bit timer (T8F) Ch.1 (clock source for USI Ch.1)
- Fine mode 8-bit timer (T8F) Ch.3 (clock source for USIL)
- Fine mode 8-bit timer (T8F) Ch.5 (clock source for UART)
- Remote controller (REMC)
- 16-bit audio PWM timer (T16P) Ch.0 and 1
- I/O ports (GPIO)

8 Real-Time Clock (RTC)

8.1 RTC Module Overview

The S1C33L27 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.
- Includes read buffers to prevent carry over at reading.
- 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.
- Clock interrupts are possible.
- Interrupt period can be selected from 1/512 second, 1/256 second, 1/128 second, 1/64 second, 1 second, 1 minute, or 1 hour. (Level interrupt mode)
- 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. (See the "Clock Management Unit (CMU)" chapter.)
- Provides the #STBY and WAKEUP pins to control the system power supply.

Figure 8.1.1 shows a block diagram of the RTC.

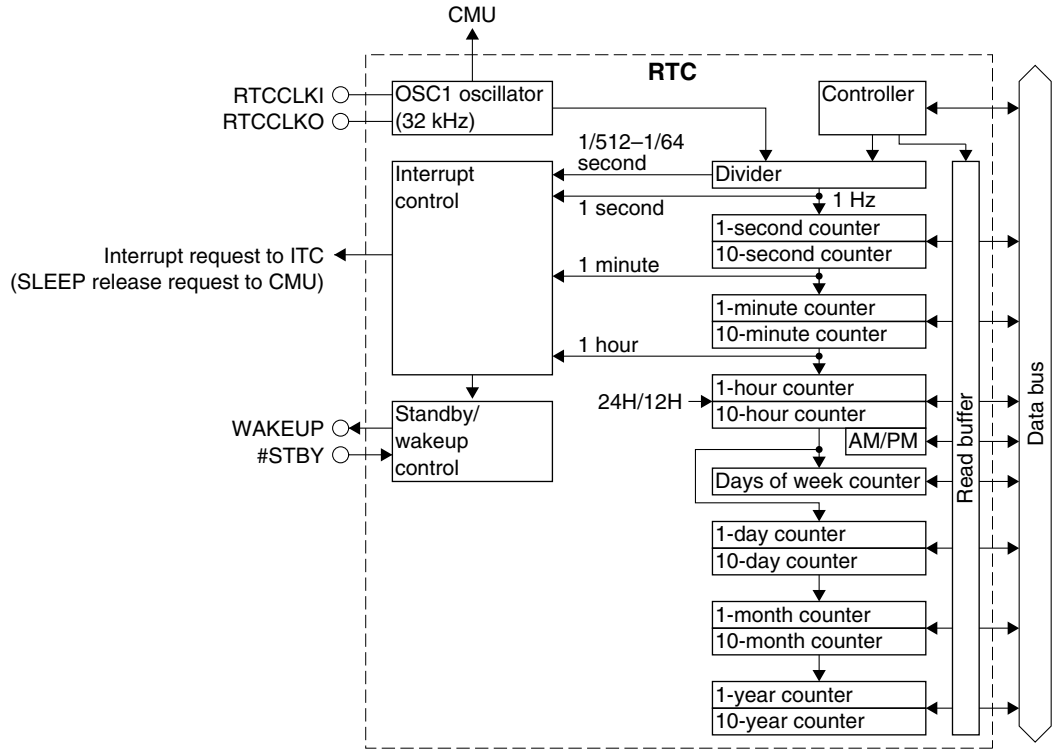


Figure 8.1.1 RTC Block Diagram

8.2 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]/RTC_SEC register.

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]/RTC_SEC register.

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]/RTC_MIN register.

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]/RTC_MIN register.

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]/RTC_HOUR register.

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]/RTC_HOUR register.
When 12-hour mode is selected, RTCAP/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.

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]/RTC_DAY register.

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]/RTC_DAY register.

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]/RTC_MONTH register.

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/RTC_MONTH register.

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]/RTC_YEAR register.

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]/RTC_YEAR register.

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]/RTC_WEEK register.
The correspondence between the counter values and days of the week can be set in a program as desired. Table 8.2.1 lists the basic correspondence.

Table 8.2.1 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

(Default: indeterminate)

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 8.3.2, “RTC Initial Sequence.”

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.

8.3 RTC Control

8.3.1 Operating Clock Control

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 register clock (PCLK) is used for accessing the RTC control registers. To setup the registers, this clock is required. After the registers are set up, the clock supply can be stopped by setting the CMU. For more information on the clock control, see the "Clock Management Unit (CMU)" chapter.

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, a wait cycle can be inserted into the RTC access cycle. The number of SYSCLK cycles to be inserted as a wait cycle can be specified using RTCWT[2:0]/MISC_RTCWT register.

Table 8.3.1.1 Number of Wait Cycles during RTC Access

RTCWT[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 (cannot be set)

(Default: 0x7)

The S1C33L27 is able to operate with RTCWT[2:0] ≥ 1.

8.3.2 RTC Initial Sequence

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. Software reset
   Write 1 to RTCRST/RTC_CNTL0 register and then write 0 to reset the RTC.
4. Confirming accessibility status of the RTC
   See Section 8.3.5, "Counter Hold and Busy Flag."
5. Disabling the divider
   Write 1 to RTCSTP/RTC_CNTL0 register to stop the divider in the RTC module.
6. Setting the RTC interrupt
   Set the RTC_INTMODE register.
   Be sure to set RTCIMD to 1 (level sense).
7. Setting the date and time
   Set the RTC_SEC, RTC_MIN, RTC_HOUR, RTC_DAY, RTC_MONTH, RTC_YEAR, and RTC_WEEK registers. Then, write 0 to RTCHLD/RTC_CNTL1 register to release the 1-second, 10-second, 1-minute, 10-minute, 1-hour, 10-hour, 1-day, 10-day, 1-month, 10-month, 1-year, 10-year, and days of week counters from hold status.

8. Starting the divider
   Write 0 to RTCSTP/RTC_CNTL0 register to run the divider in the RTC module.

8.3.3 12/24-hour Mode and Counter Settings

12-hour/24-hour mode selection

Whether to use the time clock in 12-hour or 24-hour mode can be selected using RTC24H/RTC_CNTL0 register.
RTC24H = 1: 24-hour mode
RTC24H = 0: 12-hour mode

The count range of hour counters changes with this selection.
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/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.
When setting the time of day, write either of the values above to this bit to specify A.M. or P.M.

Counter settings

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 8.3.5, “Counter Hold and Busy Flag.”

8.3.4 Start/Stop and Software Reset

Starting and stopping divider

The RTC starts counting when RTCSTP/RTC_CNTL0 register is set to 0, and stops counting when this bit is set to 1.
The RTC is started/stopped by writing data 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.

Software reset

RTCRST/RTC_CNTL0 register is the software reset bit used to reset the items shown below.
• Divider (32 kHz to 2 Hz bits)
• Interrupt request signal
• WAKEUP signal
• Some register bits (see Section 8.6 for the control bits and their initial values.)

To perform software reset, write 1 to RTCRST and then write back to 0.
The registers initialized by software reset must be re-programmed after releasing from reset status.
The divider bits above are cleared 0. The output signals above become inactive while RTCRST is set to 1 and are enabled to be output again after RTCRST is set to 0.

8.3.5 Counter Hold and Busy Flag

If 1 is carried over when writing the counters, the counter value may be corrupted. Therefore, whether counters are in a carry (busy) state should be checked before writing data to the count registers. For this purpose, control bits RTCBSY/RTC_CNTL1 register and RTCHLD/RTC_CNTL1 register are provided.
RTCBSY is a read-only flag indicating that carry is taking place. RTCBSY is set to 1 when carry is taking place; otherwise, it is 0. RTCBSY should be confirmed as being 0 before accessing the counters to ensure that the correct value will be set.
Writing 1 to RTCHLD suspends the counter operations. Note, however, that writing 1 to RTCHLD is ignored if RTCBSY is set to 1.

RTCBSY = 0 (RTC accessible)

When a value of 0 is read from RTCBSY after writing 1 to RTCHLD, it means that carry is not taking place. In this state, counter data can be written to.

After 1 is written to RTCHLD, the counters stop operating. So RTCBSY is fixed at 0, as carry will not take place. In this case, the counter hold function is also actuated, with a carry over of 1 to the 1-second counter disabled in hardware. The divider (counter for less than one second) continues operating.
Write data to the counter registers.
After writing data, reset RTCHLD to 0.
If 1 is being carried over when data is being written to a counter in the hold state, 1 second is automatically added to correct the counter values when RTCHLD is reset to 0. This correction is only effective for 1 second and no correction is conducted on the carry encountered in the second time and on. In this case, the timekeeping data gets out of order. Therefore, be sure to reset RTCHLD to 0 as soon as possible after completing the required write operation.

RTCBSY = 1 (RTC is busy)

When a value of 1 is read from RTCBSY after writing 1 to RTCHLD, it means that carry is taking place.
In this case, writing 1 to RTCHLD is ignored and RTCHLD retains 0.
A period of 4 ms per second is required for a carry over of 1 to the counters. In this case, [A] repeat writing 1 to RTCHLD and checking RTCBSY or [B] write 1 to RTCHLD and check RTCBSY after waiting for 4 ms.

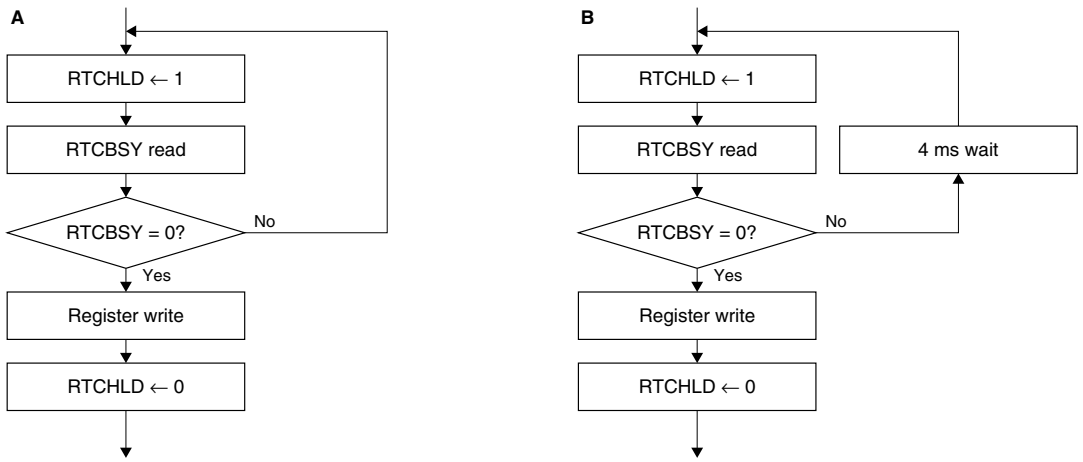


Figure 8.3.5.1 Procedure for Checking whether the RTC is Busy

8.3.6 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/RTC_CNTL0 register.
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. To check whether the 30-second correction processing has completed or not, [A] repeat checking RTCADJ or [B] check RTCADJ after waiting for 4 ms.
Accessing the counters while RTCADJ = 1 is prohibited. Writing 0 to RTCADJ and writing 1 to RTCRST are also prohibited, because it would cause the RTC to operate erratically.
Writing 1 to RTCADJ when RTCBSY is 1 may corrupt the counter values. Always make sure that RTCBSY is set to 0 before writing 1 to RTCADJ.

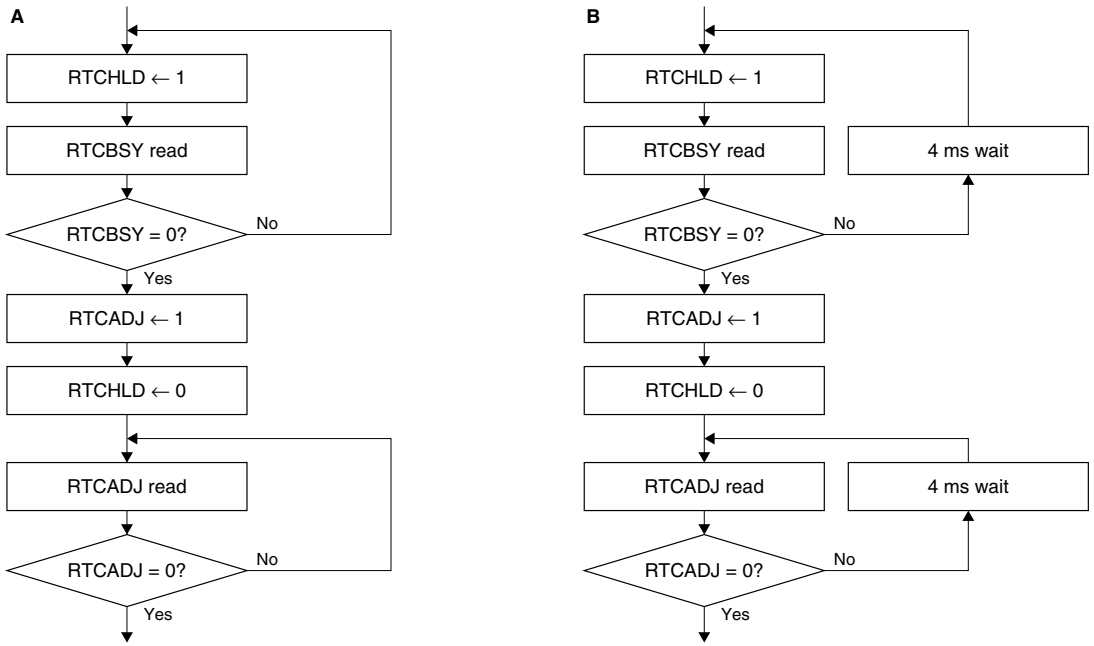


Figure 8.3.6.1 Procedure for Executing 30-second Correction

8.3.7 Counter Read

In order to prevent carry over during reading counters, the RTC includes a read buffer to hold counter data.
Before reading counter data, set RTCRDHLD/RTC_CNTL1 register to 1 to load the current counter data to the read buffer.
While RTCRDHLD is set to 1, the buffered data is read out from the counter registers. Be sure to reset RTCRDHLD to 0 after the buffered data is read out. This operation does not affect the counters. The counters keeps counting while RTCRDHLD is set to 1.

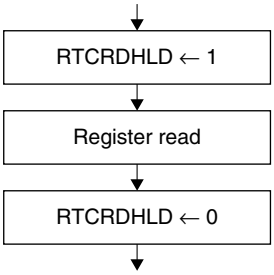


Figure 8.3.7.1 Procedure for Reading Counters

8.4 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 cancel 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 the "Interrupt Controller (ITC)" chapter. For details on how to cancel SLEEP mode using an interrupt, see the "Clock Management Unit (CMU)" chapter.

Interrupt cycle setting

The interrupt cycle (in which the RTC outputs interrupt requests at specific intervals) can be selected from seven choices listed in Table 8.4.1 by using RTCT[2:0]/RTC_INTMODE register.

Table 8.4.1 Interrupt Cycle Settings

RTCT[2:0]	Interrupt cycle
0x7	Reserved
0x6	1/128 second
0x5	1/256 second
0x4	1/512 second
0x3	1 hour
0x2	1 minute
0x1	1 second
0x0	1/64 second

RTCT[2:0] should be set while RTC interrupts are disabled. (See the procedure for enabling and disabling interrupts described below.)

Setting interrupt conditions

The RTC of the S1C33L27 supports level-sensed interrupt only.

Enabling and disabling interrupts

The RTC interrupt requests output to the ITC are enabled by setting RTCIEN/RTC_INTMODE register to 1 and disabled by setting it to 0.
RTC interrupts will be generated according to the divider and counter status and the time between writing 1 to RTCIEN and the first interrupt request is not fixed. Use the second and subsequent interrupts as valid.

Interrupt status

When the RTC is up and running, RTCIRQ/RTC_INTSTAT register is set at the cyclic interrupt intervals set up by RTCT[2:0]. When RTC interrupts are enabled by RTCIEN, interrupt requests are sent to the ITC.
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.
When RTCIEN is set to 0 (interrupt disabled), RTCIRQ is fixed at 0 (will not be set to 1).

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.

When a software reset is performed (RTCRST → 1 → 0), RTCIRQ and RTCIEN are reset to 0 to disable the interrupt request output. Also RTCT[2:0] is reset to 0x1.

8.5 WAKEUP and #STBY Pins

The S1C33L27 has a battery backup function that allows the system to turn the system power (LVDD, PLLVDD, HVDD, AVDD) off with the RTC (including the OSC1 oscillator circuit) kept active and the BBRAM data maintained by supplying RTCVDD. The RTC provides the WAKEUP and #STBY pins used for controlling this function.

The #STBY pin is used to disconnect the circuits driven with RTCVDD (RTC, OSC1, and BBRAM) from the other circuits driven with LVDD, PLLVDD, HVDD, and AVDD (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 RTCVDD circuits from the system allowing the system power (LVDD, PLLVDD, HVDD, AVDD) 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 (LVDD, PLLVDD, HVDD, AVDD) on and off. Note that leakage currents flow to the RTCVDD 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 8.5.1 shows an example of system standby/wakeup circuit using the WAKEUP and #STBY pins.

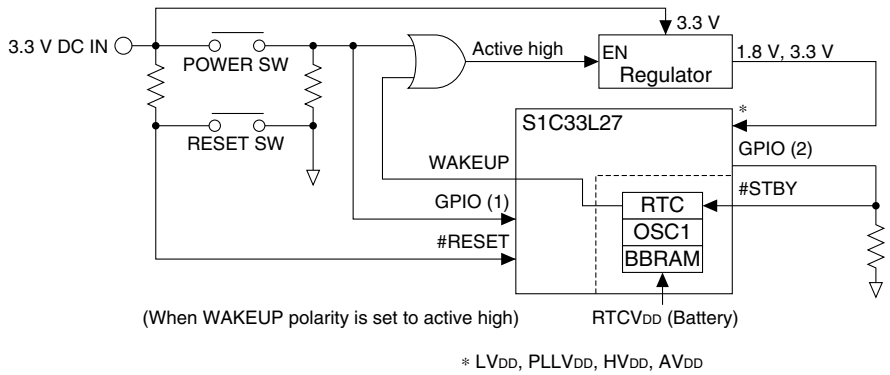


Figure 8.5.1 Example of System Standby/Wakeup Circuit

Selecting the WAKEUP signal polarity

Use WUP_POL/RTC_WAKEUP register to select the WAKEUP output level when it is asserted by an RTC interrupt or software control.

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 8.4) 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/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/RTC_WAKEUP register.
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 8.5.1 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)

When a software reset is performed (RTCRST → 1 → 0), WUP_CTL and WUP_POL are reset to 0 to set the WAKEUP signal to 0.

Control procedures

The following shows some power control procedures using the system standby/wakeup circuit shown in Figure 8.5.1. The description below assumes that the power (3.3 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 1.8 V (and 3.3 V) voltage is supplied to the S1C33L27 LV_{DD}, $PLLV_{DD}$, HV_{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 BBRAM 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 BBRAM.
(7) Clear the key located in the BBRAM (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 S1C33L27 into battery backup mode.
(3) Copy the data required to be saved into the BBRAM. In addition to this, write a key for indicating that the backup data is valid (e.g. 0xaa) to the specific location in the BBRAM.
(4) Set the RTC interrupt conditions and enable the interrupt. (when restarting the system using an RTC interrupt)
(5) Write 0x0 to the RTC_WAKEUP register to set the WAKEUP pin to output 0 (low).
(6) 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.

(7) (After the POWER SW is turned off if it is still on,) The regulator stops generating 1.8 V (and 3.3 V) and the power of the S1C33L27 except the RTCVDD 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 1.8 V (and 3.3 V) voltage is supplied to the S1C33L27 LVDD, PLLVDD, HVDD, and AVDD 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 RTCVDD 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/RTC_INTSTAT register to 0.
(7) Read the key from the specific BBRAM 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 BBRAM.
(8) Clear the key located in the BBRAM (e.g. write a value such as 0x00).
(9) Execute other processing.

8.6 Details of Control Registers

Table 8.6.1 RTC Register List

Address	Register name		Function
0x300a00	RTC_INTSTAT	RTC Interrupt Status Register	Indicates RTC interrupt status.
0x300a01	RTC_INTMODE	RTC Interrupt Mode Register	Sets up RTC interrupt modes.
0x300a02	RTC_CNTL0	RTC Control 0 Register	Controls the RTC.
0x300a03	RTC_CNTL1	RTC Control 1 Register	
0x300a04	RTC_SEC	RTC Second Register	Second counter data
0x300a05	RTC_MIN	RTC Minute Register	Minute counter data
0x300a06	RTC_HOUR	RTC Hour Register	Hour counter data
0x300a07	RTC_DAY	RTC Day Register	Day counter data
0x300a08	RTC_MONTH	RTC Month Register	Month counter data
0x300a09	RTC_YEAR	RTC Year Register	Year counter data
0x300a0a	RTC_WEEK	RTC Days of Week Register	Days of week counter data
0x300a0f	RTC_WAKEUP	RTC Wakeup Configuration Register	Sets up RTC wakeup conditions.

The following describes each RTC register.

Notes: • When data is written to the register, the “Reserved” bits must always be written as 0 and not 1.

• 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 8.3.5, “Counter Hold and Busy Flag,” and Section 8.3.7, “Counter Read.”

RTC Interrupt Status Register (RTC_INTSTAT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
RTC Interrupt Status Register (RTC_INTSTAT)	0x300a00 (8 bits)	D7–1	–	reserved	–				–	–	0 when being read.
		D0	RTCIRQ	Interrupt status	1	Occurred	0	Not occurred	X (0)	R/W	Reset by writing 1.

Init.: ( ) indicates the value set after a software reset (RTCRST → 1 → 0) is performed.

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 (software reset value)
1 (W): Resets this bit to 0
0 (W): Has no effect

This bit is set at cyclic interrupt intervals set up by RTCT[2:0]/RTC_INTMODE register. When RTC interrupts have been enabled by RTCIEN/RTC_INTMODE register at this time, an interrupt request is sent to the ITC.

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.

RTC Interrupt Mode Register (RTC_INTMODE)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
RTC Interrupt Mode Register (RTC_INTMODE)	0x300a01 (8 bits)	D7–5	–	reserved	–		–	–	0 when being read.
		D4–2	RTCT[2:0]	RTC interrupt cycle setup	RTCT[2:0]	Cycle	X (0x1)	R/W	
					0x7	reserved			
					0x6	1/128 second			
					0x5	1/256 second			
					0x4	1/512 second			
					0x3	1 hour			
					0x2	1 minute			
					0x1	1 second			
					0x0	1/64 second			
		D1	RTCIMD	reserved	1		X (1)	R/W	Always set to 1.
		D0	RTCIEN	RTC interrupt enable	1 Enable	0 Disable	X (0)	R/W	

Init.: ( ) indicates the value set after a software reset (RTCRST → 1 → 0) is performed.

D[7:5] Reserved

D[4:2] RTCT[2:0]: RTC Interrupt Cycle Setup Bits

These bits select the RTC interrupt cycle.

Table 8.6.2 Interrupt Cycle Settings

RTCT[2:0]	Interrupt cycle
0x7	Reserved
0x6	1/128 second
0x5	1/256 second
0x4	1/512 second
0x3	1 hour
0x2	1 minute
0x1	1 second
0x0	1/64 second

(Default: indeterminate, software reset: 0x1)

RTCIRQ/RTC_INTSTAT register is set by a count-up pulse of the interrupt cycle counter selected. When RTC interrupts are enabled by RTCIEN, an interrupt request is sent to the ITC.
RTCT[2:0] should be set while RTC interrupts are disabled. (These bits may also be set simultaneously when RTC interrupts are enabled.)

D1 RTCIMD: Reserved (Always be sure to set to 1.)

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 (software reset value)

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 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.

RTC Control 0 Register (RTC_CNTL0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Control 0 Register (RTC_CNTL0)	0x300a02 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	RTC24H	24H/12H mode select	1 24H 0 12H	X (0)	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2	RTCADJ	30-second adjustment	1 Adjust 0 –	X (0)	R/W	
		D1	RTCSTP	Divider run/stop control	1 Stop 0 Run	X (0)	R/W	
		D0	RTCRST	Software reset	1 Reset 0 –	X (0)	R/W	

Init.: ( ) indicates the value set after a software reset (RTCRST → 1 → 0) is performed.

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 (software reset value)

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) (software reset value)

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: Divider Run/Stop Control Bit

This bit starts or stops the divider. It also indicates divider operating status.

1 (W): Stops divider

0 (W): Starts divider

1 (R/W): Divider/counters are idle

0 (R/W): Divider/counters are operating (software reset value)

Setting this bit to 0 starts the divider; setting it to 1 stops the divider.
The value read from this bit is 0 when the divider/counters are operating, and 1 when the counters are idle.
This bit starts/stops the divider 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 divider stops while carry of a counter is taking place, the count value may be corrupted. Therefore, see Section 8.3.5 to ensure that carry is not taking place when the divider is stopped. This is not required when, for example, the contents of all counters are newly set again.

D0 RTCRST: Software Reset Bit

This bit resets the divider and output signals.
1 (R/W): Reset
0 (R/W): Negate reset (software reset value)

To perform software reset, write 1 to RTCRST and then write 0.
The software reset clears the 32 kHz to 2 Hz divider bits, negates the interrupt request and WAKEUP signals, and initializes some control bits.
When setting up the RTC, first perform software reset using RTCRST.

RTC Control 1 Register (RTC_CNTL1)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
RTC Control 1 Register (RTC_CNTL1)	0x300a03 (8 bits)	D7–3	–	reserved	–				–	–	0 when being read.
		D2	RTCRDHLD	Read buffer enable	1	Enable	0	Disable	X (0)	R/W	
		D1	RTCBSY	Counter busy flag	1	Busy	0	R/W possible	X (0)	R	
		D0	RTCHLD	Counter hold control	1	Hold	0	Running	X (0)	R/W	

Init.: ( ) indicates the value set after a software reset (RTCRST → 1 → 0) is performed.

D[7:3] Reserved

D2 RTCRDHLD: Read Buffer Enable Bit

This bit enables or disables the read buffer.
1 (R/W): Enabled
0 (R/W): Disabled (software reset value)

In order to prevent carry over during reading counters, the RTC includes a read buffer to hold counter data. Before reading counter data, set RTCRDHLD to 1 to load the current counter data to the read buffer. While RTCRDHLD is set to 1, the buffered data is read out from the counter registers. Be sure to reset RTCRDHLD to 0 after the buffered data is read out. This operation does not affect the counters. The counters keeps counting while RTCRDHLD is set to 1.

D1 RTCBSY: Counter Busy Flag Bit

This flag indicates whether 1 is being carried over to the next-digit counter.
1 (R): Busy (while carry is taking place)
0 (R): Accessible for read/write (software reset value)

Attempting a write or stop operation may corrupt the counter values if 1 is being carried over. Therefore, this bit should be checked to confirm that the counters are not in a carry (busy) state before writing data to the counter registers.

When a value of 0 is read from RTCBSY after writing 1 to RTCHLD, it means that carry is not taking place. In this state, counter data can be written to.
After 1 is written to RTCHLD, the counters stop operating. So RTCBSY is fixed at 0, as carry will not take place. In this case, the counter hold function is also actuated, with a carry over of 1 to the 1-second counter disabled in hardware. The divider (counter for less than one second) continues operating.
Write data to the counter registers. After writing data, reset RTCHLD to 0.
If 1 is being carried over when data is being written to a counter in the hold state, 1 second is automatically added to correct the counter values when RTCHLD is reset to 0. This correction is only effective for 1 second and no correction is conducted on the carry encountered in the second time and on. In this case, the timekeeping data gets out of order. Therefore, be sure to reset RTCHLD to 0 as soon as possible after completing the required write operation.

When a value of 1 is read from RTCBSY after writing 1 to RTCHLD, it means that carry is taking place. In this case, writing 1 to RTCHLD is ignored and RTCHLD retains 0. A period of 4 ms per second is required for a carry over of 1 to the counters. In this case, repeat writing 1 to RTCHLD and checking RTCBSY, or write 1 to RTCHLD and check RTCBSY after waiting for 4 ms.

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 (software reset value)

For the operation of this bit, see the description of RTCBSY above.

RTC Second Register (RTC_SEC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Second Register (RTC_SEC)	0x300a04 (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	

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Note: Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

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.

RTC Minute Register (RTC_MIN)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Minute Register (RTC_MIN)	0x300a05 (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	

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Note: Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

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.

RTC Hour Register (RTC_HOUR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Hour Register (RTC_HOUR)	0x300a06 (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 to 9	X (*)	R/W	

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

• Rewriting RTC24H/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/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/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.

RTC Day Register (RTC_DAY)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Day Register (RTC_DAY)	0x300a07 (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	

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

• Rewriting RTC24H/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.

RTC Month Register (RTC_MONTH)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Month Register (RTC_MONTH)	0x300a08 (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	

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

• Rewriting RTC24H/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.

RTC Year Register (RTC_YEAR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Year Register (RTC_YEAR)	0x300a09 (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	

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

• Rewriting RTC24H/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.

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)	0x300a0a (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			

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

Notes: • Data should not be read from or written to the counters while 1 is being carried over. (See Section 8.3.5, "Counter Hold and Busy Flag," and Section 8.3.7, "Counter Read.")

• Rewriting RTC24H/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

These bits are the septenary counter (that counts from 0 to 6) bits 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 8.6.3 lists the basic correspondence.

Table 8.6.3 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, software reset: previous value retained)

RTC Wakeup Configuration Register (RTC_WAKEUP)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
RTC Wakeup Configuration Register (RTC_WAKEUP)	0x300a0f (8 bits)	D7–2	–	reserved	–				–	–	0 when being read.
		D1	WUP_CTL	WAKEUP control	1	Active	0	Inactive	X (0)	R/W	
		D0	WUP_POL	WAKEUP polarity select	1	Active low	0	Active high	X (0)	R/W	

Init.: ( ) indicates the value set after a software reset (RTCRST → 1 → 0) is performed.

D[7:2] Reserved

D1 WUP_CTL: WAKEUP Control Bit

This bit controls the WAKEUP output.

1 (R/W): Active

0 (R/W): Inactive (software reset value)

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 Select Bit

This bit selects the active level of the WAKEUP output signal.

1 (R/W): Active low

0 (R/W): Active high (software reset value)

9 SRAM Controller (SRAMC)

The S1C33L27 includes a bus controller that controls access to external memories. The bus controller consists of an SRAM controller (SRAMC) for controlling the SRAM, an SDRAM controller (SDRAMC) for controlling the SDRAM, and a data queue buffer (DQB) for efficiently reading from external memories.
The following describes the SRAMC. For information on the SDRAMC and DQB, see the "SDRAM Controller (SDRAMC)" chapter.

9.1 SRAMC Overview

The SRAM controller (SRAMC) is a module for controlling the external bus. It can output up to six chip enable signals and configure the access cycle and the connected device type for respective areas assigned to each chip enable signal. When the CPU or DMAC accesses an external memory space, the SRAMC reads/writes from/to the connected memory or I/O device according to the defined access conditions.
The features of the SRAMC are described below.

- 26-bit address bus (maximum 64M-byte address space)
- 8-bit or 16-bit selectable data bus
- Up to six chip enable signals are available for connecting external devices.
- Supports connections to a Flash ROM, SRAM, and other devices such as an LCD driver.
- Programmable bus access wait cycle (0 to 15 cycles)
- Little endian access
- Supports memory mapped I/O devices.
- Supports either A0 or BS (bus strobe) access type.
- Supports external wait requests via the #WAIT pin.

9.2 SRAMC Pins

Table 9.2.1 lists the pins used by the SRAMC.

Table 9.2.1 SRAMC Pin List

Pin name	I/O	Qty	Function
D[15:0]	I/O	16	External data bus D[15:0]
A0/#BSL	O	1	External address bus A0 / Bus strobe (low byte) signal output
A[25:1]	O	25	External address bus A[25:1]
#CE10, #CE9, #CE8, #CE7, #CE5, #CE4	O	6	Chip enable signal outputs
#RD	O	1	Read signal output
#WRL	O	1	Write (low byte) signal output
#WRH/#BSH	O	1	Write (high byte) signal / Bus strobe (high byte) signal output
#WAIT	I	1	External wait request signal input

Notes: • Some of the bus control pins listed above are shared with general-purpose I/O ports and they may be 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 bit.
For details on how to switch over the pin functions, see the "I/O Ports (GPIO)" chapter.

• The bus control signals can be pulled up or forcibly driven low via software. For more information, see the "I/O Ports (GPIO)" chapter.

9.3 SRAMC Operating Clock

The SRAMC operates with SYSCLK supplied from the CMU.
BCLK is also used for accessing the SRAMC control registers.
For more information on the clock control, see the "Clock Management Unit (CMU)" chapter.

9.4 External Memory Areas

The SRAMC supports an external memory space, which is divided into 14 areas as shown in Figure 9.4.1.

			#CE4	#CE5	#CE7	#CE8	#CE9	#CE10
Area 22	0xffff ffff 0x8000 0000	External memory 2G (64M) bytes (*2)					●	
Area 21	0x7fff ffff 0x4000 0000	External memory 1G (64M) bytes (*2)				●		
Area 20	0x3fff ffff 0x2000 0000	External memory 512M (64M) bytes (*2)						●
Area 19 (*1)	0x1fff ffff 0x1000 0000	External memory 256M (64M) bytes (*2)			●			
Area 18	0x0fff ffff 0x0c00 0000	Reserved						
Area 17	0x0bff ffff 0x0800 0000	Reserved						
Area 16	0x07ff ffff 0x0600 0000	External memory 32M bytes		●				
Area 15	0x05ff ffff 0x0400 0000	External memory 32M bytes		●				
Area 14	0x03ff ffff 0x0300 0000	External memory 16M bytes	●					
Area 13	0x02ff ffff 0x0200 0000	External memory 16M bytes						●
Area 12	0x01ff ffff 0x0180 0000	Reserved						
Area 11	0x017f ffff 0x0100 0000	Reserved						
Area 10	0x00ff ffff 0x00c0 0000	External memory 4M bytes						●
Area 9	0x00bf ffff 0x0080 0000	External memory 4M bytes					●	
Area 8	0x007f ffff 0x0060 0000	External memory 2M bytes				●		
Area 7 (*1)	0x005f ffff 0x0040 0000	External memory 2M bytes			●			
Area 6	0x003f ffff 0x0030 0000	(Reserved area for internal peripheral modules)						
Area 5	0x002f ffff 0x0020 0000	External memory 1M bytes		●				
Area 4	0x001f ffff 0x0010 0000	External memory 1M bytes	●					

*1 Usable as an SDRAM area.
*2 Since the address bus is 26-bit wide, valid area for each area is 64M bytes from the top.

Figure 9.4.1 S1C33L27 External Memory Space

Areas 4, 5, 7 to 10, 13 to 16, and 19 to 22 comprise an external memory area accessible from the SRAMC, to which external memory devices may be connected. The access conditions can be set by area, including the device type and size as well as the number of wait cycles.

9.4.1 Chip Enable Signals

The SRAMC provides maximum 26 bits of an external address bus, 16 bits of an external data bus, and 6 chip-enable pins (#CE4, #CE5, #CE7 to #CE10), allowing access to a maximum 366M-byte address space.

Two or more areas are assigned to each chip-enable signal. Table 9.4.1.1 shows the relationship between the chip-enable pins and corresponding areas.

Table 9.4.1.1 Chip Enable Pins and Corresponding Areas

#CE pin	Corresponding area	Available area capacity for a series of adjacent addresses					
		Area	Capacity	Area	Capacity	Area	Capacity
#CE4	Areas 4 and 14	Area 4	1MB	Area 14	16MB	–	–
#CE5	Areas 5, 15, and 16	Area 5	1MB	Areas 15 + 16	64MB	–	–
#CE7	Areas 7 and 19	Area 7	2MB	Area 19	64MB	–	–
#CE8	Areas 8 and 21	Area 8	2MB	Area 21	64MB	–	–
#CE9	Areas 9 and 22	Area 9	4MB	Area 22	64MB	–	–
#CE10	Areas 10, 13, and 20	Area 10	4MB	Area 13	16MB	Area 20	64MB

The #CE*x* signal becomes active when an address in any corresponding area is accessed.

9.4.2 Area Condition Settings

Bus access conditions can be set by area for each #CE*x* signal. Therefore, the same conditions are set for two or more areas accommodated by the respective #CE*x* signals.
This section describes the parameters to be set individually for each #CE*x* area and the relevant control bits.
The SRAMC control registers are initialized by an initial reset. These registers should be set up in software to suit the external device configuration or specification as required.

Note: Letter '*x*' in the control bit and #CE names denotes a #CE area number (4, 5, or 7 to 10).

Endian mode

The S1C33L27 supports little endian mode only.

Device type

The device size can be selected from 8 bits and 16 bits. Additionally, for a 16-bit device, the device type can also be selected from the A0 (default) or BSL modes.
For selection, use CE*x*TYPE[1:0]/SRAMC_TYPE register.

Table 9.4.2.1 Device Type Selections

CE*x*TYPE[1:0]	Device type	Pins to be used
0x3–0x2	8-bit device	A[25:0], D[7:0], #CE*x*, #RD, #WRL
0x1	16-bit BSL device	A[25:1], D[15:0], #CE*x*, #RD, #WRL, #BSL(A0), #BSH
0x0	16-bit A0 device	A[25:1], D[15:0], #CE*x*, #RD, #WRL, #WRH

(Default: 0x0)

#CE setup time

The setup time for #CE*x* signals (from a #CE*x* falling edge to the read/write signal falling edge) can be set to within the range from 1 to 4 cycles. Use CE*x*SETUP[1:0] in the SRAMC_TMG47 and SRAMC_TMG810 registers for settings.

Table 9.4.2.2 #CE Setup Time Settings

CE*x*SETUP[1:0]	Setup time
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x3)

#CE hold time

The hold time for #CE*x* signals (from a read/write signal rising edge to the #CE*x* rising edge) can be set to within the range from 1 to 4 cycles. Use CE*x*HOLD[1:0] in the SRAMC_TMG47 and SRAMC_TMG810 registers for settings.

Table 9.4.2.3 #CE Hold Time Settings

CE*x*HOLD[1:0]	Hold time
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x3)

Static wait cycles

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. According to the specifications of the connected device, set an appropriate wait cycle using CE*x*WAIT[3:0] in the SRAMC_TMG47 and SRAMC_TMG810 registers. If CE*x*WAIT[3:0] is set to 0, no static wait cycle is inserted. In this case, the minimum read/write pulse width will be one cycle.

Table 9.4.2.4 Static Wait Cycle Settings

CE*x*WAIT[3:0]	Static wait cycle	Read/write cycle
0xf	15 cycles	16 cycles (+ #WAIT)
0xe	14 cycles	15 cycles (+ #WAIT)
:	:	:
0x1	1 cycle	2 cycles (+ #WAIT)
0x0	0 cycles	1 cycle (+ #WAIT)

(Default: 0xf)

The area to which an SRAM device is connected allows dynamic wait control using the #WAIT pin in addition to the static wait control.

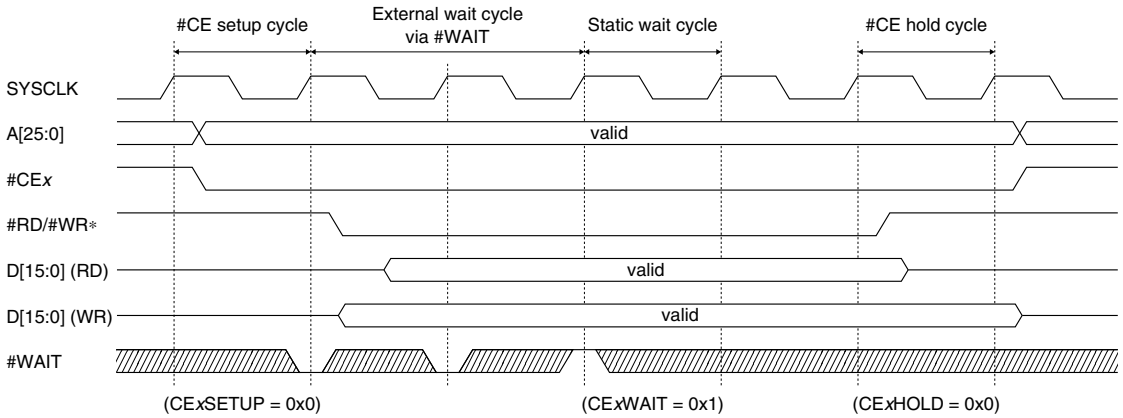


Figure 9.4.2.1 Example of Timing Parameter Settings

9.5 Connection of External Devices and Bus Operation

9.5.1 Connecting External Devices

The following shows examples of connecting the S1C33L27 and SRAM.

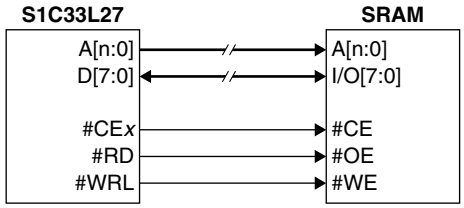


Figure 9.5.1.1 Example of 8-bit SRAM Connection

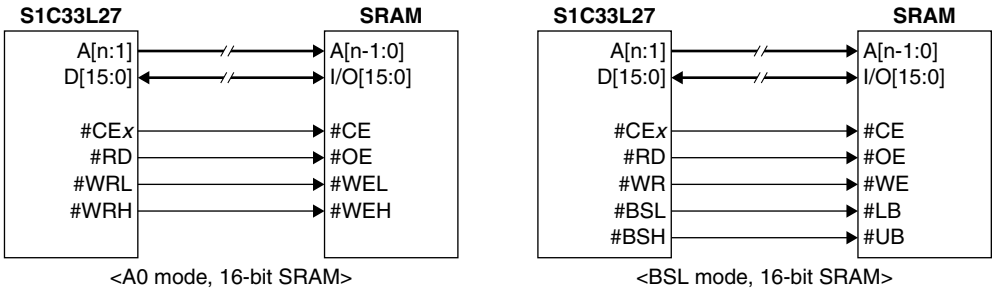


Figure 9.5.1.2 Example of 16-bit SRAM Connection

9.5.2 Data Configuration in Memory

The S1C33L27 SRAMC handles byte (8-bit), halfword (16-bit), and word (internal 32-bit) data. To access data in a memory, addresses aligned to the boundary of the data size must be specified. Specifying other addresses generates address misaligned exceptions.

Instructions (e.g., stack manipulating and branch instructions) that rewrite the contents of the stack pointer (SP) or program counter (PC) forcibly alter the address specified to a boundary address to prevent address misaligned exceptions. For details of address misaligned exceptions, refer to the C33 PE Core Manual.

Table 9.5.2.1 shows where each type of data is located in a memory.

Table 9.5.2.1 Data Locations in Memory

Data type	Location
Byte	Byte boundary (all addresses)
Halfword	Halfword boundary (A0 = 0)
Word	Word boundary (A[1:0] = 0b00)

All halfword and word data in a memory are accessed in little endian mode. To increase memory efficiency, try locating the same type of data at continuous addresses to reduce blank areas created by positioning at boundary addresses as much as possible.

9.5.3 External Bus Operation

The internal data bus size in the S1C33L27 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 9.5.3.1 shows bus operations in A0 and BSL modes.
For details on how to connect memory, see Section 9.5.1, "Connecting External Devices."

Table 9.5.3.1 Bus Operations

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	Byte	W	*	*	#WRL	–	D[7:0]	–	–	–	1
		R	*	*	#RD	–	D[7:0]	–	–	–	1
	Halfword	W	*	0	#WRL	–	D[7:0]	–	–	–	1st
			*	1		–	D[15:8]		–	–	2nd
		R	*	0	#RD	–	D[7:0]	–	–	–	1st
			*	1		–	D[15:8]		–	–	2nd
	Word	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	Byte	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
	Halfword	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
	Word	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

9.6 Bus Access Timing Charts

9.6.1 SRAM Read/Write Timing with No External #WAIT

1. SRAM read/write timings with no static wait cycles

[Example settings]
Device size: 16 bits
Number of static wait cycles: 0 cycles
#CE setup/hold time: 1 cycle

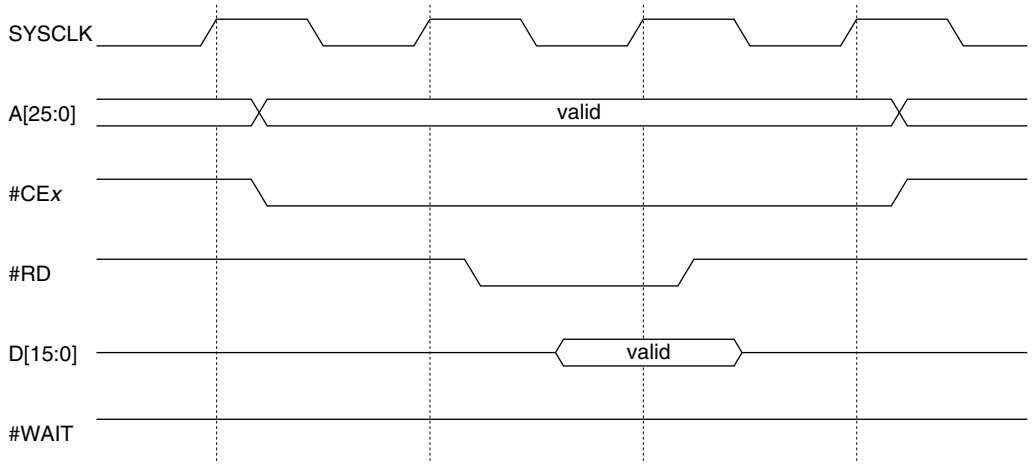


Figure 9.6.1.1 SRAM Read Timing with No Static Wait Cycle

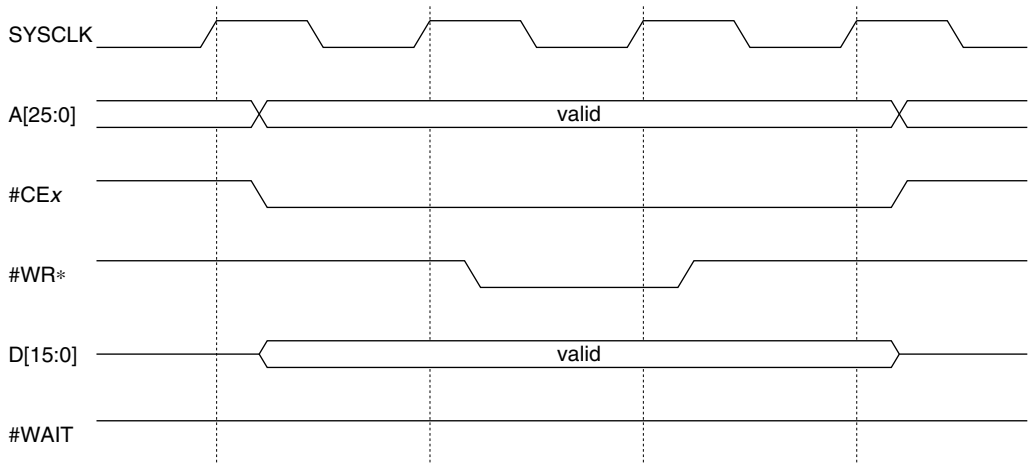


Figure 9.6.1.2 SRAM Write Timing with No Static Wait Cycle

2. SRAM read/write timings with static wait cycles

[Example settings]
Device size: 16 bits
Number of static wait cycles: 2 cycles
#CE setup/hold time: 1 cycle

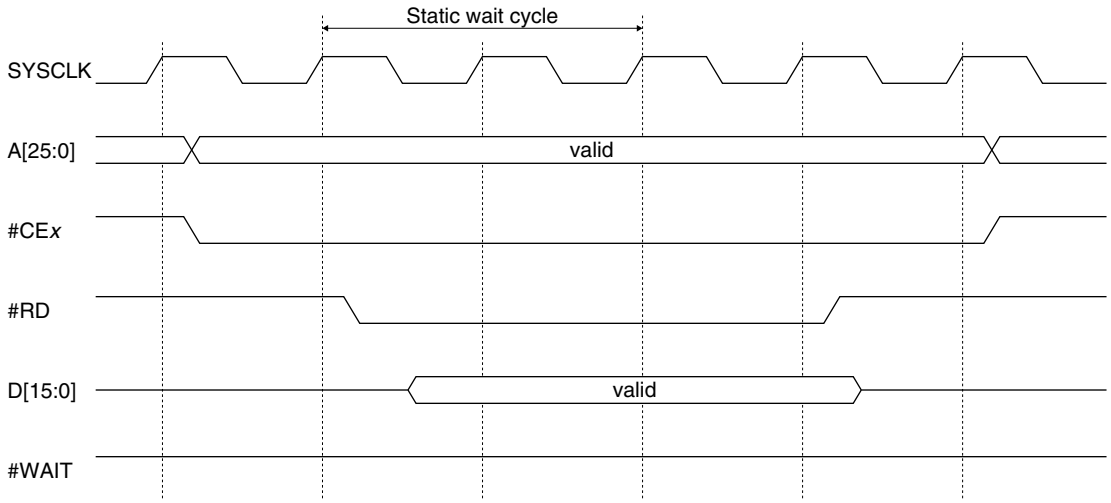


Figure 9.6.1.3 SRAM Read Timing with Static Wait Cycle

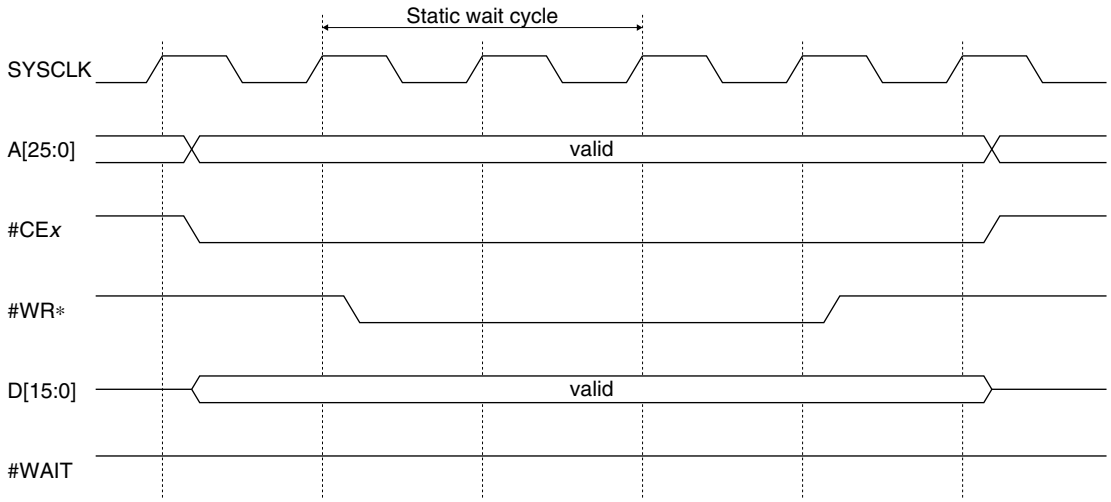


Figure 9.6.1.4 SRAM Write Timing with Static Wait Cycle

9.6.2 SRAM Read/Write Timings with External #WAIT

Wait cycles can be inserted from the #WAIT pin only for SRAM-type devices.
The external #WAIT signal is sampled on the rising edges of SYSCLK after the #CE setup cycles end and no later than 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
Number of static wait cycles: 0 cycles (see Note below)
#CE setup/hold time: 1 cycle (see Note below)

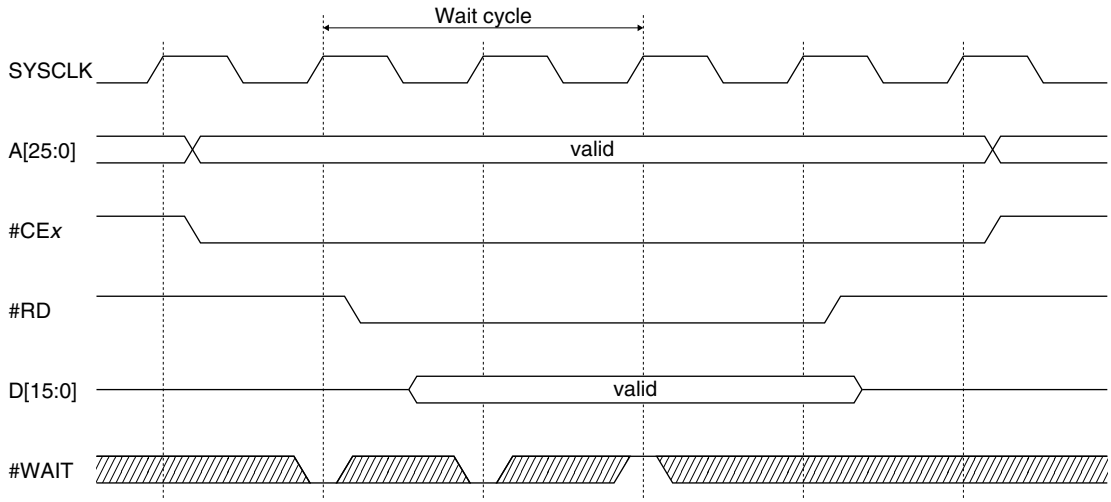


Figure 9.6.2.1 SRAM Read Timing with External #WAIT

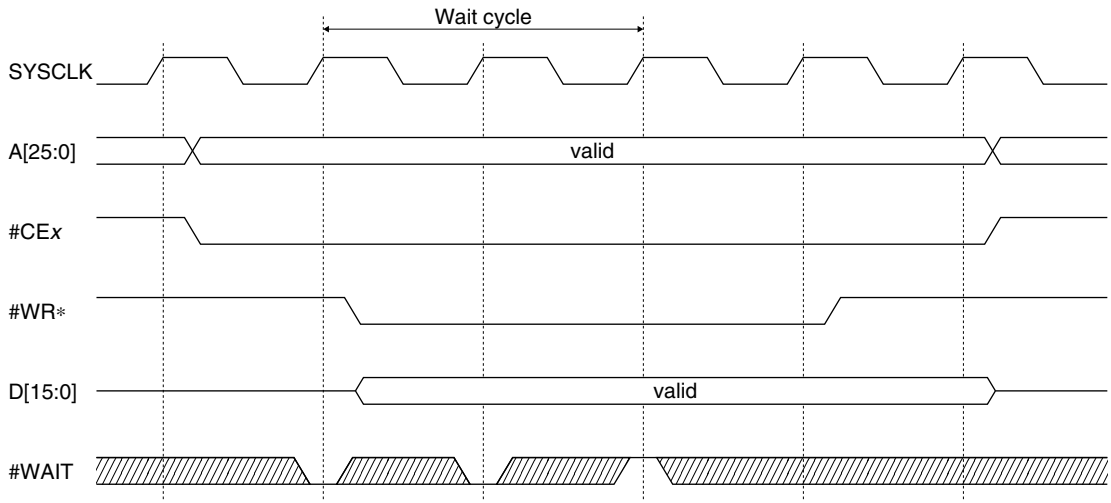


Figure 9.6.2.2 SRAM Write Timing with External #WAIT

Note: Figures 9.6.2.1 and 9.6.2.2 assume a very low operating speed for convenience of explanation. When actually using an external wait request, to allow for a delay due to noise filter of the #WAIT pin, be sure to set the #CE setup cycles (CE*x*SETUP[1:0]) or static wait cycles (CE*x*WAIT[3:0]) as follows:

- When using #WAIT, set CE*x*SETUP[1:0] to 0x1 or higher, or set CE*x*WAIT[3:0] to 0x1 or higher.
- When a #CE*x* signal is used to generate a #WAIT signal, set the conditions no lower than the following.

  CE*x*SETUP[1:0] = 0x1 and CE*x*WAIT[3:0] = 0x3, or
  CE*x*SETUP[1:0] = 0x2 and CE*x*WAIT[3:0] = 0x2, or
  CE*x*SETUP[1:0] = 0x3 and CE*x*WAIT[3:0] = 0x1, or
  CE*x*SETUP[1:0] = 0x4 and CE*x*WAIT[3:0] = 0x0

- When a #RD/#WRH/#WRL signal is used to generate a #WAIT signal, set the conditions no lower than the following.

  CE*x*WAIT[3:0] = 0x4

When settings are other than the listed above, external wait is ineffective.

9.7 Control Register Details

Table 9.7.1 List of SRAMC Registers

Address	Register name		Function
0x302220	SRAMC_TMG47	#CE[7:4] Access Timing Configuration Register	Set #CE[7:4] access conditions
0x302224	SRAMC_TMG810	#CE[10:8] Access Timing Configuration Register	Set #CE[10:8] access conditions
0x302228	SRAMC_TYPE	#CE[10:4] Device Configuration Register	Set #CE[10:4] device types

The following describes each SRAMC register.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

#CE[7:4] Access Timing Configuration Register (SRAMC_TMG47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
#CE[7:4] Access Timing Configuration Register (SRAMC_ TMG47) **#CE[7:4] Access Timing Configuration Register (SRAMC_ TMG47)**	**0x302220** (32 bits) **0x302220** (32 bits)	D31–30	**CE7SETUP [1:0]**	#CE7 setup cycle	CE7SETUP[1:0] 0x3 0x2 0x1 0x0	Setup cycle 4 cycles 3 cycles 2 cycles 1 cycle	0x3	R/W	
		D29–28	**CE7HOLD [1:0]**	#CE7 hold cycle	CE7HOLD[1:0] 0x3 0x2 0x1 0x0	Hold cycle 4 cycles 3 cycles 2 cycles 1 cycle	0x3	R/W	
		D27–24	**CE7WAIT [3:0]**	#CE7 static wait cycle	CE7WAIT[3:0] 0xf 0xe : 0x1 0x0	Wait cycle 15 cycles 14 cycles : 1 cycle 0 cycles	0xf	R/W	
		D23–16	**–**	reserved	–		–	–	1 when being read.
		D15–14	**CE5SETUP [1:0]**	#CE5 setup cycle	CE5SETUP[1:0] 0x3 : 0x0	Setup cycle 4 cycles : 1 cycle	0x3	R/W	
		D13–12	**CE5HOLD [1:0]**	#CE5 hold cycle	CE5HOLD[1:0] 0x3 : 0x0	Hold cycle 4 cycles : 1 cycle	0x3	R/W	
		D11–8	**CE5WAIT [3:0]**	#CE5 static wait cycle	CE5WAIT[3:0] 0xf : 0x0	Wait cycle 15 cycles : 0 cycles	0xf	R/W	
		D7–6	**CE4SETUP [1:0]**	#CE4 setup cycle	CE4SETUP[1:0] 0x3 : 0x0	Setup cycle 4 cycles : 1 cycle	0x3	R/W	

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
#CE[7:4] Access Timing Configuration Register (SRAMC_ TMG47)	**0x302220** (32 bits)	D5–4	**CE4HOLD [1:0]**	#CE4 hold cycle	CE4HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D3–0	**CE4WAIT [3:0]**	#CE4 static wait cycle	CE4WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					:	:			
					0x0	0 cycles			

See the descriptions of the SRAMC_TMG810 register.

#CE[10:8]Access Timing Configuration Register (SRAMC_TMG810)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
#CE[10:8] Access Timing Configuration Register (SRAMC_ TMG810)	**0x302224** (32 bits)	D31–24	**–**	reserved	–		–	–	1 when being read.
		D23–22	**CE10SETUP [1:0]**	#CE10 setup cycle	CE10SETUP[1:0]	Setup cycle	0x3	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D21–20	**CE10HOLD [1:0]**	#CE10 hold cycle	CE10HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D19–16	**CE10WAIT [3:0]**	#CE10 static wait cycle	CE10WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					0xe	14 cycles			
					:	:			
					0x1	1 cycle			
					0x0	0 cycles			
		D15–14	**CE9SETUP [1:0]**	#CE9 setup cycle	CE9SETUP[1:0]	Setup cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D13–12	**CE9HOLD [1:0]**	#CE9 hold cycle	CE9HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D11–8	**CE9WAIT [3:0]**	#CE9 static wait cycle	CE9WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					:	:			
					0x0	0 cycles			
		D7–6	**CE8SETUP [1:0]**	#CE8 setup cycle	CE8SETUP[1:0]	Setup cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D5–4	**CE8HOLD [1:0]**	#CE8 hold cycle	CE8HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D3–0	**CE8WAIT [3:0]**	#CE8 static wait cycle	CE8WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					:	:			
					0x0	0 cycles			

The SRAMC_TMG47 and the SRAMC_TMG810 registers are used to set the SRAM access timing for each #CE area.

Letter '*x*' in the control bit and #CE names denotes a #CE area number (4, 5, or 7 to 10).

D[31:30], D[23:22], D[15:14], D[7:6]

CE*x*SETUP[1:0]: #CE*x* Setup Cycle Bits

Configures the #CE*x* signal setup time (#CE*x* falling edge to read/write signal falling edge).

Table 9.7.2 #CE Setup Time Settings

CE*x*SETUP[1:0]	Setup time
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x3)

D[29:28], D[21:20], D[13:12], D[5:4]

CE*x*HOLD[1:0]: #CE*x* Hold Cycle Bits

Configures the #CE*x* signal hold time (read/write signal rising edge to #CE*x* rising edge).

Table 9.7.3 #CE Hold Time Settings

CE*x*HOLD[1:0]	Hold time
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x3)

D[27:24], D[19:16], D[11:8], D[3:0]

CE*x*WAIT[3:0]: #CE*x* Static Wait Cycle Bits

Configures the #CE*x* signal 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. If CE*x*WAIT[3:0] is set to 0, no static wait cycle is inserted. In this case, the minimum read/write pulse width will be one cycle.

Table 9.7.4 Static Wait Cycle Settings

CE*x*WAIT[3:0]	Static wait cycle	Read/write cycle
0xf	15 cycles	16 cycles (+ #WAIT)
0xe	14 cycles	15 cycles (+ #WAIT)
:	:	:
0x1	1 cycle	2 cycles (+ #WAIT)
0x0	0 cycles	1 cycle (+ #WAIT)

(Default: 0xf)

The area to which an SRAM device is connected allows dynamic wait control using the #WAIT pin in addition to the static wait control.

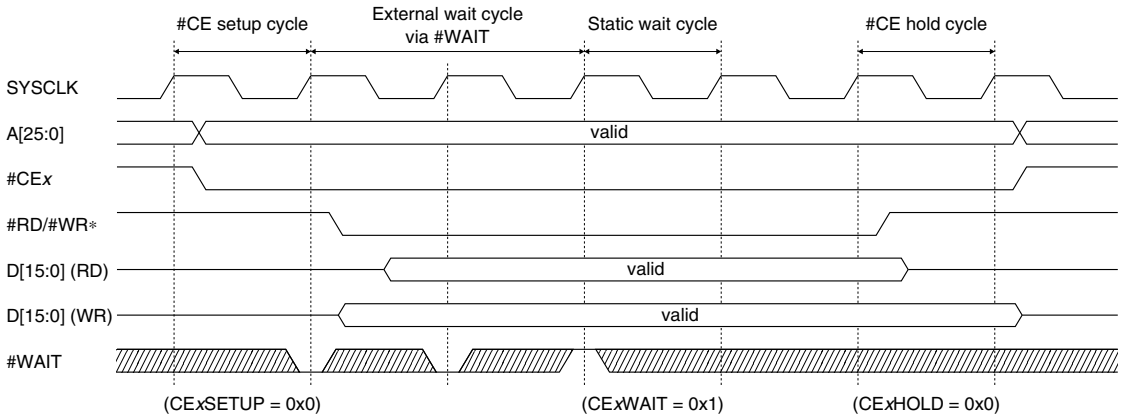


Figure 9.7.1 Example of Timing Parameter Settings

#CE[10:4] Device Configuration Register (SRAMC_TYPE)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
#CE[10:4] Device Configuration Register (SRAMC_TYPE)	0x302228 (32 bits)	D31–14	–	reserved	–		–	–	0 when being read.
		D13–12	CE10TYPE [1:0]	#CE10 device type	CE*x*TYPE[1:0]	Device type	0x0	R/W	
					0x3–0x2	8-bit device			
		D11–10	CE9TYPE [1:0]	#CE9 device type	0x1	16-bit BSL type	0x0	R/W	
					0x0	16-bit A0 type			
		D9–8	CE8TYPE [1:0]	#CE8 device type			0x0	R/W	
		D7–6	CE7TYPE [1:0]	#CE7 device type			0x0	R/W	
		D5–4	–	reserved	–		–	–	0 when being read.
		D3–2	CE5TYPE [1:0]	#CE5 device type	CE*x*TYPE[1:0]	Device type	0x0	R/W	
					0x3–0x2	8-bit device			
		D1–0	CE4TYPE [1:0]	#CE4 device type	0x1	16-bit BSL type	0x0	R/W	
					0x0	16-bit A0 type			

D[31:14], D[5:4]

Reserved

D[13:12], D[11:10], D[9:8], D[7:6], D[3:2], D[1:0]

CE*x*TYPE[1:0]: #CE*x* Device Type Bits

Selects the device type for each #CE area.

Table 9.7.5 Device Type Selections

CE*x*TYPE[1:0]	Device type	Pins to be used
0x3–0x2	8-bit device	A[25:0], D[7:0], #CE*x*, #RD, #WRL
0x1	16-bit BSL device	A[25:1], D[15:0], #CE*x*, #RD, #WRL, #BSL(A0), #BSH
0x0	16-bit A0 device	A[25:1], D[15:0], #CE*x*, #RD, #WRL, #WRH

(Default: 0x0)

10 SDRAM Controller (SDRAMC)

The S1C33L27 includes a bus controller that controls access to external memories. The bus controller consists of an SRAM controller (SRAMC) for controlling the SRAM, an SDRAM controller (SDRAMC) for controlling the SDRAM, and a data queue buffer (DQB) for efficiently reading from external memories.
The following describes the SDRAMC and DQB. For information on the SRAMC, see the "SRAM Controller (SRAMC)" chapter.

10.1 SDRAMC Overview

The SDRAM controller (SDRAMC) allows up to 64MB of SDRAM to be connected to Areas 7 and 19.
The main features of the SDRAMC are outlined below.

- Supports direct connection of an SDRAM from minimum 16M bits (2MB) to maximum 512M bits (64MB).
- The operating clock frequency: Maximum 60 MHz (same as the CPU clock frequency)
- Data bus width: 16 bits (16-bit SDRAM × 1 or 8-bit SDRAM × 2)
- CAS latency: Can be set to 1, 2 or 3.
- Supports burst transfers (burst length: 2).
- Supports 2- or 4-bank SDRAM (BA1 and BA0 outputs).
  Row address range: 2K (SDA10–SDA0), 4K (SDA11–SDA0), or 8K (SDA12–SDA0)
  Column address range: 256 (SDA7–SDA0), 512 (SDA8–SDA0), or 1K (SDA9–SDA0)
- Supports byte writes with the DQML and DQMH pins.
- Includes a programmable 12-bit auto refresh counter.
  Necessary refreshing enabled irrespective of the clock frequency used.
- Provided with intelligent self-refresh mode for low-power operation.
- Supports the EMRS (Extended Mode Register Set) command to program drive strength, temperature compensated self refresh, and partial array self refresh, in addition to MRS (Mode Register Set).

10.2 SDRAMC Pins

Table 10.2.1 lists the pins used by the SDRAMC.

Table 10.2.1 SDRAMC Pin List

Pin name	I/O	Qty	Function
A[15:14]/SDBA[1:0]	O	2	Bank select signal output
A[13:1]/SDA[12:0]	O	13	Address signal output
D[15:0]	I/O	16	Data signal I/O
SDCKE	O	1	SDRAM clock enable signal output
SDCLK	O	1	SDRAM clock output
#SDCS	O	1	SDRAM chip enable signal output
A20/#SDRAS	O	1	SDRAM row address strobe signal output
A19/#SDCAS	O	1	SDRAM column address strobe signal output
A18/#SDWE	O	1	SDRAM write signal output
A16/DQML	O	1	SDRAM data (to select low-order byte) input/output mask signal output
A17/DQMH	O	1	SDRAM data (to select high-order byte) input/output mask signal output

Notes: • Some of the bus control pins listed above are shared with general-purpose I/O ports and they may be configured for I/O ports at initial reset. Before the SDRAMC signals assigned to these pins can be used, the functions of these pins must be switched for the SDRAMC by setting each corresponding port function select bit.
For details on how to switch over the pin functions, see the "I/O Ports (GPIO)" chapter.

• The bus control signals can be pulled up or forcibly driven low via software. For more information, see the "I/O Ports (GPIO)" chapter.

10.3 SDRAMC Operating Clock

The SDRAMC operates with SYSCLK supplied from the CMU. This clock is used as SDCLK.
BCLK is also used for accessing the SDRAMC control registers.
For more information on the clock control, see the "Clock Management Unit (CMU)" chapter.

10.4 Configuration of SDRAM

10.4.1 SDRAM Area

A #CE7 area (Area 7 or Area 19) is reserved for the SDRAMC. Note that the #CE7 area is configured for an SRAM area controlled with the SRAMC as the SDRAMC is disabled at initial reset. Therefore, to use the SDRAM, the #CE7 area must be configured for the SDRAM area by setting SDON/SDRAMC_INIT register to 1.

Note: Setting SDON to 1 overrides the external SRAM access conditions for the #CE7 area set in the SRAMC.

10.4.2 SDRAM Size and Access Condition Settings

Table 10.4.2.1 lists the conditions related to SDRAM size and timing parameters that the SDRAMC can accommodate.

Table 10.4.2.1 SDRAM Setup Items

Item	Setting contents	Control bit settings
SDRAM address configuration	32M × 16 bits × 1	ADDRC[2:0]/SDRAMC_CFG register = 0x7
	16M × 16 bits × 1	ADDRC[2:0]/SDRAMC_CFG register = 0x3
	8M × 16 bits × 1	ADDRC[2:0]/SDRAMC_CFG register = 0x2
	4M × 16 bits × 1	ADDRC[2:0]/SDRAMC_CFG register = 0x1
	1M × 16 bits × 1 (default)	ADDRC[2:0]/SDRAMC_CFG register = 0x0 (default)
	16M × 8 bits × 2	ADDRC[2:0]/SDRAMC_CFG register = 0x6
	8M × 8 bits × 2	ADDRC[2:0]/SDRAMC_CFG register = 0x5
CAS latency	3, 2, or 1	CAS[1:0]/SDRAMC_APP register = 0x3, 0x2 or 0x1
Burst length	2 (fixed)	–
t_{RP}, t_{RCD}	1 (default) to 4 cycles	T24NS[1:0]/SDRAMC_CFG register = 0x0 (default) to 0x3
t_{RAS}	1 (default) to 8 cycles	T60NS[2:0]/SDRAMC_CFG register = 0x0 (default) to 0x7
t_{RC}, t_{RFC}, t_{XSR}	1, 5, 9, or 13 cycles	T80NS[3:0]/SDRAMC_CFG register = 0x0, 0x4, 0x8, or 0xc

SDRAM address configuration

Use ADDRC[2:0]/SDRAMC_CFG register to select the SDRAM size and chip configuration. This selection also sets up the bank size, column address size (page size), and row address size.

Table 10.4.2.2 SDRAM Size Selections and SDRAM Address

ADDRC[2:0]	0x0 (default)	0x1	0x2	0x3	0x7	0x4	0x5	0x6
SDRAM device	16-bit device					Two 8-bit devices		
Capacity (M bit)	16	64	128	256	512	–	64 × 2	128 × 2
Data width	16 bits							
Row size	2048	4096	4096	8192	8192	–	4096	4096
Column size	256	256	512	512	1024	–	512	1024
Number of banks	2	4	4	4	4	–	4	4
A25	–	–	–	–	R10	–	–	–
A24	–	–	–	R12	R12	–	–	R11
A23	–	–	R11	R11	R11	–	R11	R10
A22	–	R11	R9	R9	R9	–	R9	R9
A21	–	R8	R8	R8	R8	–	R8	R8
A20	R7	R7	R7	R7	R7	–	R7	R7
A19	R6	R6	R6	R6	R6	–	R6	R6
A18	R5	R5	R5	R5	R5	–	R5	R5
A17	R4	R4	R4	R4	R4	–	R4	R4
A16	R3	R3	R3	R3	R3	–	R3	R3
A15	R2	R2	R2	R2	R2	–	R2	R2
A14	R1	R1	R1	R1	R1	–	R1	R1
A13	R0	R0	R0	R0	R0	–	R0	R0
A12	R10	R10	R10	R10	B1	–	R10	B1
A11	R9	R9	B0	B0	B0	–	B0	B0
A10	R8	B1	B1	B1	C9	–	B1	C9
A9	B0	B0	C8	C8	C8	–	C8	C8
A8	C7	C7	C7	C7	C7	–	C7	C7
A7	C6	C6	C6	C6	C6	–	C6	C6
A6	C5	C5	C5	C5	C5	–	C5	C5
A5	C4	C4	C4	C4	C4	–	C4	C4
A4	C3	C3	C3	C3	C3	–	C3	C3
A3	C2	C2	C2	C2	C2	–	C2	C2
A2	C1	C1	C1	C1	C1	–	C1	C1
A1	C0	C0	C0	C0	C0	–	C0	C0
A0	DQM	DQM	DQM	DQM	DQM	–	DQM	DQM

When reading/writing byte data, the SDRAMC decodes A0/#BSL and WRH/BSH into DQML and DQMH. Upper address bits that are not used (depending on memory size) are all fixed to 0.
Figures 10.4.2.1 and 10.4.2.2 show examples of how to connect SDRAMs.

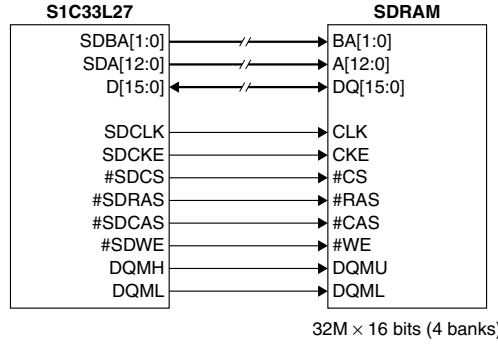


Figure 10.4.2.1 64MB SDRAM Connection Example

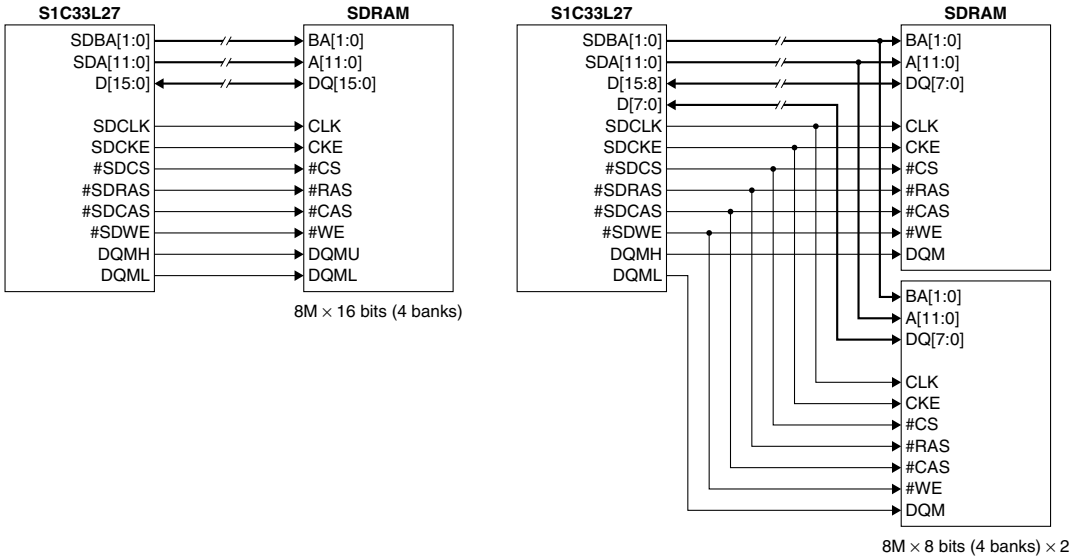


Figure 10.4.2.2 16MB SDRAM Connection Example

Timing setup

The following parameters can be configured according to the SDRAM specifications.

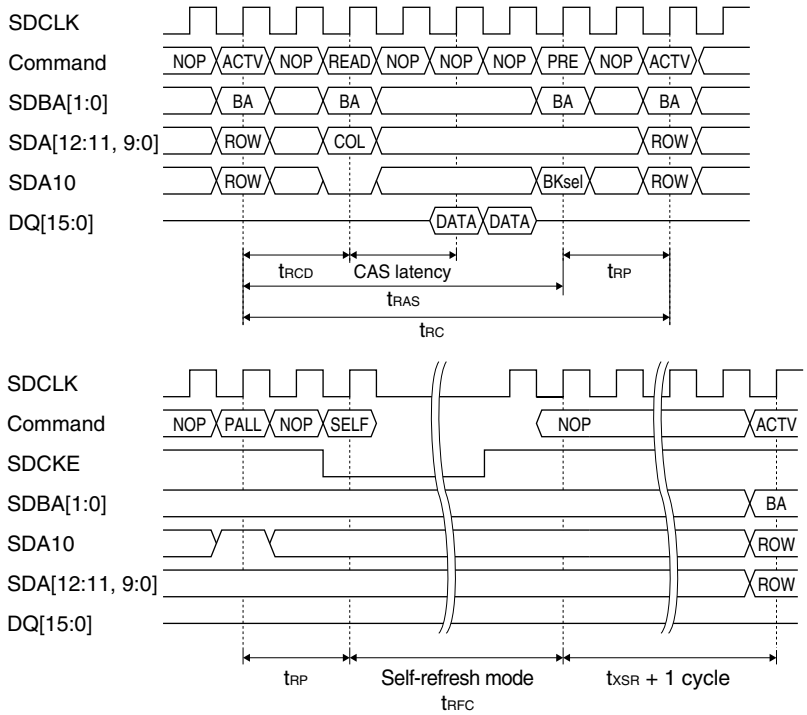


Figure 10.4.2.3 SDRAM Timing Parameters

(1) CAS Latency

CAS latency refers to the number of SDCLK clock cycles that run until data is output from the SDRAM after the READ command is issued. In this SDRAM interface, CAS latency can be set from 1 to 3 using CAS[1:0]/SDRAMC_APP register.

Table 10.4.2.3 CAS Latency Settings

CAS[1:0]	CAS latency
0x3	3
0x2	2
0x1	1
0x0	Reserved

(Default: 0x2)

(2) tRC, tRFC, tXSR

tRC: ACTIVE to ACTIVE command cycle time
tRFC: Auto-refresh cycle time
tXSR: Self-refresh end to ACTIVE command period

These timing parameters can be set from 1 to 16 cycles in SDCLK using T80NS[3:0]/SDRAMC_CFG register.

Table 10.4.2.4 tRC, tRFC, and tXSR Settings

T80NS[3:0]	tRC, tRFC, tXSR
0xc	13 cycles
0x8	9 cycles
0x4	5 cycles
0x0	1 cycle
Other	Reserved

(Default: 0xe)

(3) tRAS

tRAS: ACTIVE to PRECHARGE command period

This timing parameter can be set from 1 to 8 cycles in SDCLK using T60NS[2:0]/SDRAMC_CFG register.

Table 10.4.2.5 tRAS Settings

T60NS[2:0]	tRAS
0x7	8 cycles
0x6	7 cycles
:	:
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

(4) tRP, tRCD

tRP: PRECHARGE to ACTIVE command period
tRCD: ACTIVE to READ/WRITE delay time

These timing parameters can be set from 1 to 4 cycles in SDCLK using T24NS[1:0]/SDRAMC_CFG register.

Table 10.4.2.6 tRP and tRCD Settings

T24NS[1:0]	tRP, tRCD
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

10.5 Control and Operation of SDRAM Interface

10.5.1 Initializing SDRAM

To use the SDRAM, it must be initialized by following the procedure below.

1. Initializing the SDRAMC registers
   Set up the SDRAMC registers in the following order:

   (1) SDRAMC_CFG register
   Set the SDRAM size, address configuration and access timing parameters.

   (2) SDRAMC_REF register
   Set the auto-refresh and self-refresh counters.

   (3) SDRAMC_INIT register
   Set SDON to 1 to enable SDRAMC.

   (4) SDRAMC_APP register
   Set the CAS latency.

2. Waiting after SDRAM power-on
   After the power to the SDRAM is turned on, a NOP state (#SDCS = 1) must be maintained at least for a certain length of time (e.g., 100 µs or 200 µs). Because this duration varies with each SDRAM, refer to the specifications of the SDRAM being used.

3. Executing an SDRAM initial sequence
   In order to initialize the SDRAM, the PALL (Precharge All banks), REF (Auto Refresh), and MRS (Mode Register Set) commands must be executed sequentially. Note that the initialization sequence depends on the SDRAM used.
   Example 1: PALL → REF → REF → MRS (→ EMRS)
   Example 2: PALL → MRS → REF → REF (→ REF → REF → REF → REF → REF → REF)

   Refer to the specifications of the SDRAM to be used for the initialization sequence.
   Each command can be executed separately using the control bit shown below.

   To execute the PALL (Precharge All) command:
   Write 0x12 to the SDRAMC_INIT register to set INIPRE to 1.
   Then write any data to any address in the SDRAM. This dummy write causes the PALL command to be sent to the SDRAM.

   To execute the REF (Auto Refresh) command:
   Write 0x11 to the SDRAMC_INIT register to set INIREF to 1.
   Then write any data to any address in the SDRAM. This dummy write causes the REF command to be sent to the SDRAM.

   When executing the REF command twice or more, insert the nop instruction between the executions.
   REF command execution → nop instruction execution → REF command execution (→ REF → nop → REF...)

   The SDRAM timing parameters set in the SDRAMC_CFG register is disabled when this initialization sequence is executed. Therefore, enough number of nop instructions must be executed to satisfy the SDRAM timings.

   To execute the MRS/EMRS (Mode Register Set/Extended Mode Register Set) command:
   Write 0x14 to the SDRAMC_INIT register to set INIMRS to 1.
   Then write any data to a specific address of SDRAM shown below according to the CAS latency (MRS) or extended mode parameters (EMRS).

Table 10.5.1.1 Data Write Address to Execute the MRS/EMRS Command

CPU address	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1
SDRAM address	BA1	BA0	SDA12	SDA11	SDA10	SDA9	SDA8	SDA7	SDA6	SDA5	SDA4	SDA3	SDA2	SDA1	SDA0
MRS	Mode		Reserved			WB	Test mode		CAS latency			BT	Burst length		
CAS latency = 1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1
CAS latency = 2	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1
CAS latency = 3	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1
EMRS	Mode		Reserved							DS	TCSR		PASR		
	1	0	0	0	0	0	0	0	0	See the SDRAM specification.					

For example, to execute an MRS command with CAS latency = 2, write any value to address 0x10000442 (when the SDRAM is mapped to Area 19) after writing 0x14 to the SDRAMC_INIT register.

Notes:
- The CAS latency specified in the MRS command must be the same as the setting for CAS[1:0]/SDRAMC_APP register.
- After the initial sequence commands are executed, the command enable bit must be set to 0. Write 0x10 to the SDRAMC_INIT register after the last initialization command has been executed.
- The self-refresh function must be disabled until the SDRAM has finished initialization.

4. Checking if the SDRAM has been initialized

INIDO/SDRAMC_INIT register is reset to 0 after power-on, and is set to 1 upon completion of the SDRAM initialization sequence shown above. Make sure that INIDO is set to 1 before the SDRAM is accessed.
In addition to being reset at power-on, INIDO is reset to 0 by writing 0 to SDON/SDRAMC_INIT register.

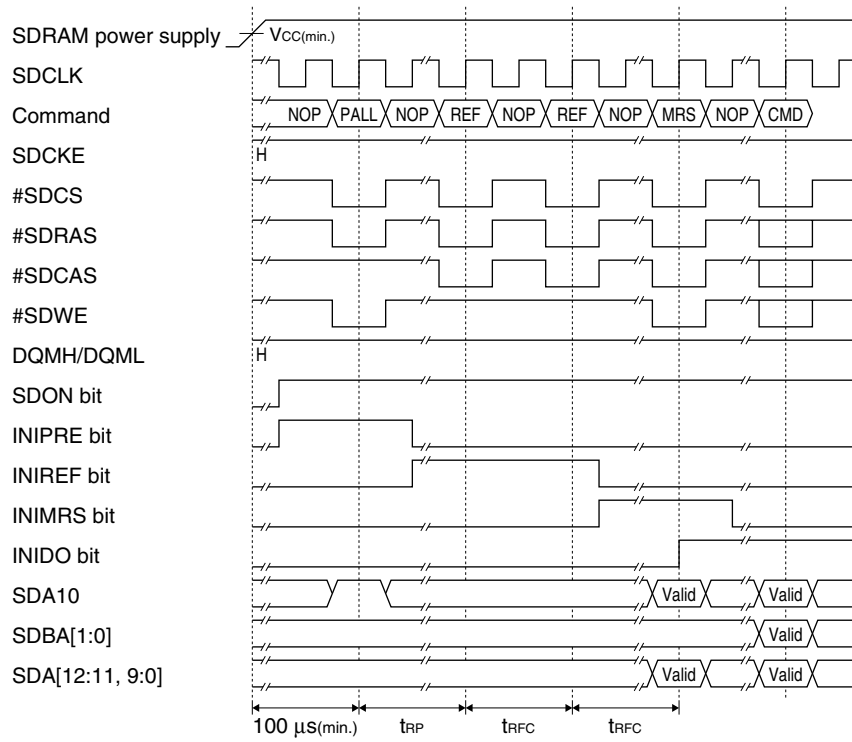


Figure 10.5.1.1 SDRAM Power-on and Initialization

10.5.2 SDRAM Commands

The SDRAM is controlled by commands that are comprised of a combination of high or low logic level signals. The table below lists the commands output by the SDRAMC.

Table 10.5.2.1 List of the Supported SDRAM Commands

Command	SDCKE	DQM (DQMH/L)	SDBA[1:0]	SDA10	SDA[12:11] SDA[9:0]	#SDCS	#SDRAS	#SDCAS	#SDWE
Deselect	H	–	–	–	–	H	–	–	–
Bank Active (ACTV)	H	–	V	V	V	L	L	H	H
Specified Bank Precharge (PRE)	H	–	V	L	–	L	L	H	L
Precharge All banks (PALL)	H	–	–	H	–	L	L	H	L
Write (WRIT)	H	–	V	L	V	L	H	L	L
Read (READ)	H	–	V	L	V	L	H	L	H
Mode Register Set (MRS)	H	–	–	V	V	L	L	L	L
No Operation (NOP)	H	–	–	–	–	L	H	H	H
Auto-Refresh (REF)	H	–	–	–	–	L	L	L	H
Self-Refresh (SELF)	H → L	–	–	–	–	L	L	L	H
End Self-Refresh	L → H	–	–	–	–	H	–	–	–
Data Write/Output Enable	H	L	–	–	–	–	–	–	–
Data Write/Output Disable	H	H	–	–	–	–	–	–	–

V = Valid, – = Optional/Unknown, L = Low level, H = High level

Because all of these commands are output by the SDRAM controller as necessary, they do not need to be controlled by the user program, except for initializing the SDRAM.

10.5.3 SDRAM Bus Operations

The external data bus of the S1C33L27 is sized to 16 bits. Depending on the device size and data size of the instruction executed, two or more bus operations may occur. The table below shows bus operations in the SDRAM area.

Table 10.5.3.1 Bus Operations

Device size	Data size	R/W	A1	A0	Little endian			Access count
					Valid signal	D[15:8] pins	D[7:0] pins	
16 bits	Byte	W	*	0	DQML	–	D[7:0]	1
			*	1	DQMH	D[7:0]	–	1
		R	*	0	DQMH/L	–	D[7:0]	1
			*	1		D[7:0]	–	1
	Halfword	W	*	*	DQMH/L	D[15:0]		1
		R	*	*	DQMH/L	D[15:0]		1
	Word	W	0	*	DQMH/L	D[15:0]		1st
			1	*		D[31:16]		2nd
		R	0	*	DQMH/L	D[15:0]		1st
			1	*		D[31:16]		2nd

10.5.4 Read/Write Cycles

Read cycle

The SDRAMC always reads data from the SDRAM in bursts. The burst length is fixed to 2. Figure 10.5.4.1 shows an example of timing chart when reading out 2-word data from the same row address.

Parameter setting example: CAS latency = 2, t_{RCD} = 2 cycles, t_{RAS} = 4 cycles, t_{RP} = 2 cycles

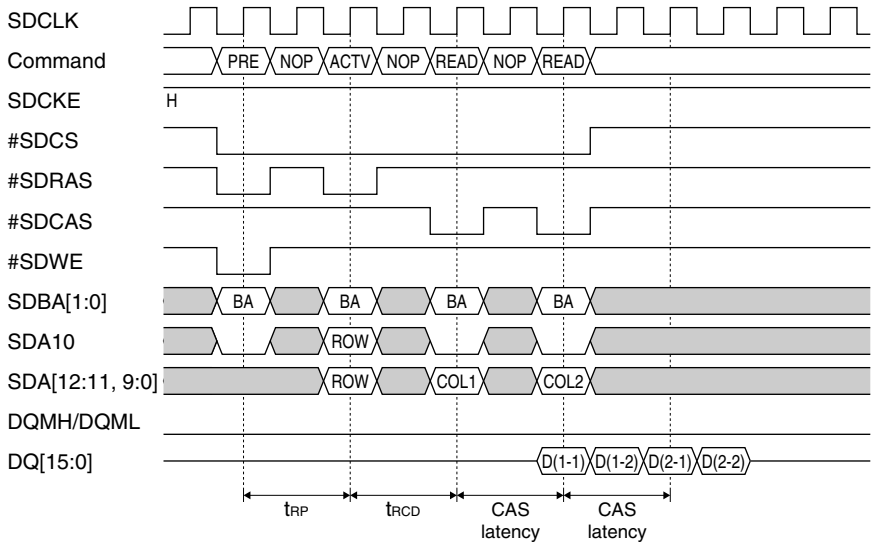


Figure 10.5.4.1 Burst Read in the Same Page

Figure 10.5.4.2 shows an example of a timing chart in cases where the row address is changed during burst read.

Parameter setting example: CAS latency = 2, t_{RCD} = 2 cycles, t_{RAS} = 4 cycles, t_{RP} = 2 cycles

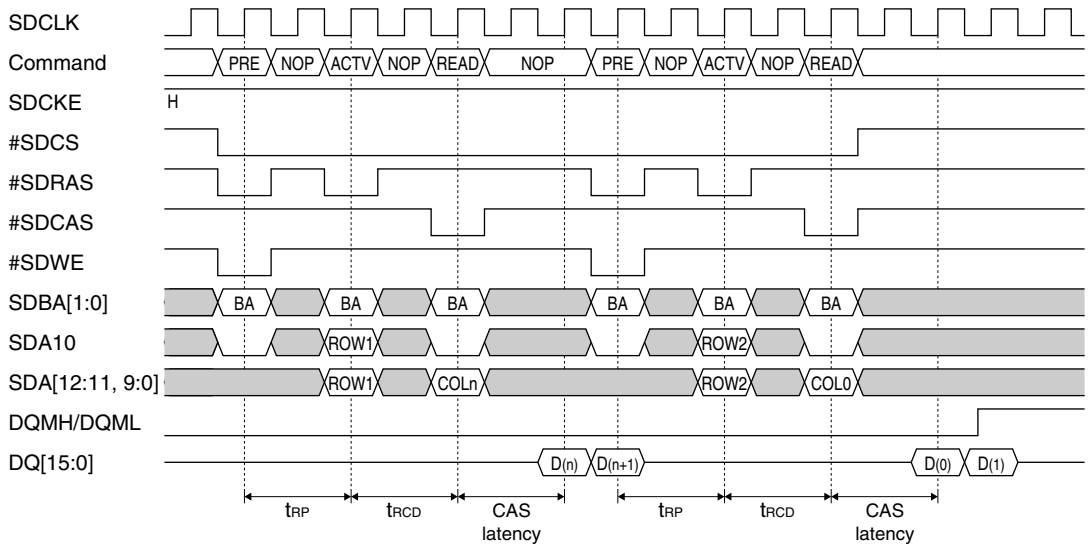


Figure 10.5.4.2 Changing Row Address During Burst Read

Write cycle

When writing to the SDRAM, data are always written in a single operation.

Parameter setting example: CAS latency = 2, t_{RCD} = 2 cycles, t_{RAS} = 4 cycles, t_{RP} = 2 cycles

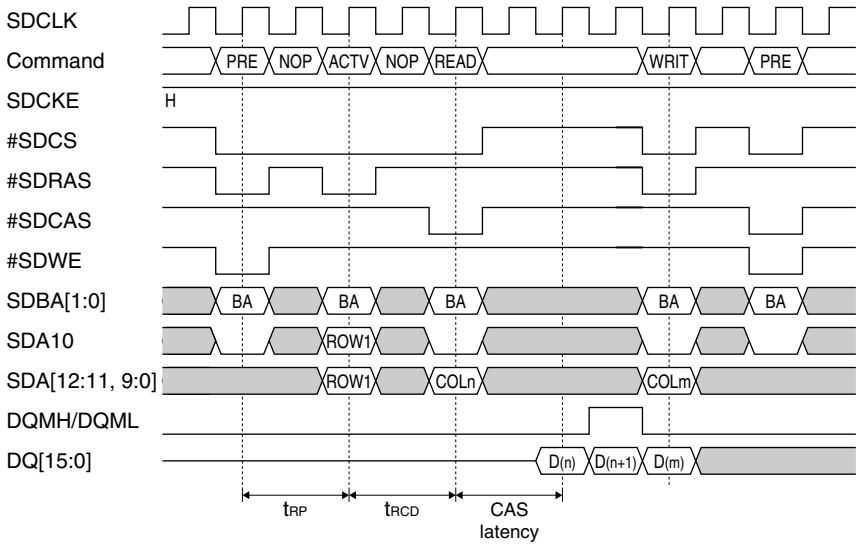


Figure 10.5.4.3 Burst Read to Single Write (same page)

Bank interleaved access

Multiple banks (up to four banks) can be activated at the same time. This makes it possible to access the SDRAM successively, one bank after another without issuing the ACTV (Active) command.

Parameter setting example: CAS latency = 2, t_{RCD} = 2 cycles, t_{RAS} = 4 cycles, t_{RP} = 2 cycles

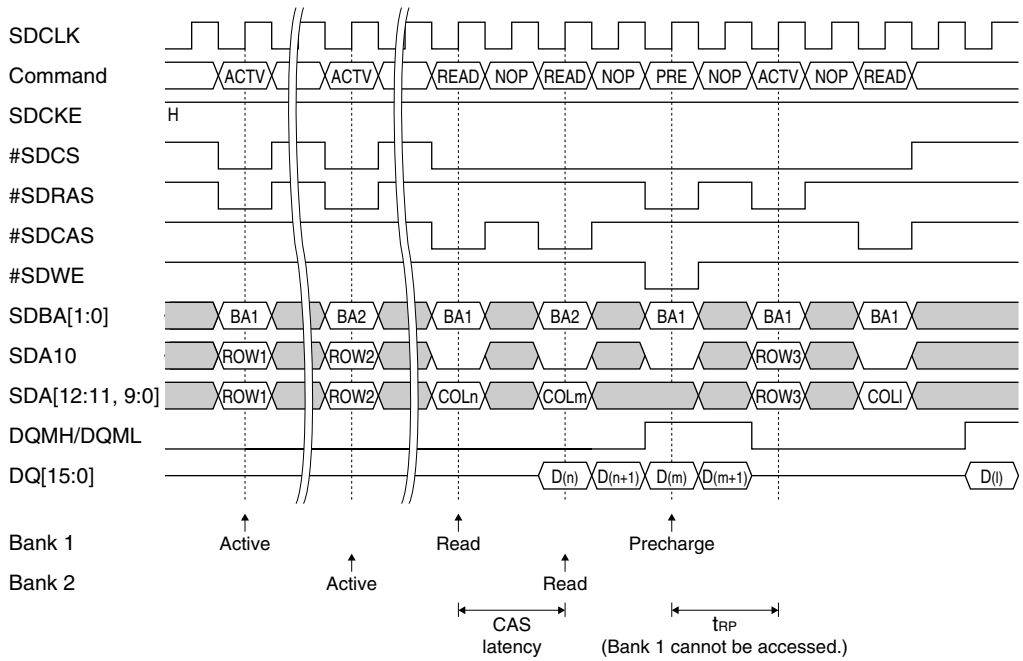


Figure 10.5.4.4 Bank Interleaved Access

10.5.5 SDRAM Refresh

The SDRAMC supports two SDRAM refresh modes: auto-refresh and self-refresh.

Auto-refresh

The SDRAMC includes a 12-bit auto refresh counter. This counter continues counting on the SDCLK clock edges, and when a specified count is reached, commands are sent to the SDRAM that precharges and auto-refreshes all banks. The counter is reset at that time, and starts counting for the next refresh period. The counter is also reset by self-refresh.

The auto-refresh period is determined by the SDCLK (SYSCLK) clock frequency and the count value set in AURCO[11:0]/SDRAMC_REF register.

AURCO[11:0] should be set to the appropriate value meeting the specifications of the SDRAM. The count value is obtained by the equation below.

$$AURCO \leq \frac{RFP}{ROWS} \times f_{CLK} - BL - CL - 2 \times t_{RP} - t_{RCD} - 3$$

RFP: Maximum refresh period [s]
ROWS: Row address size
f_{CLK}: SDCLK clock frequency [Hz]
BL: Burst length (= 2)
CL: CAS latency
t_{RP}: PRECHARGE command period [Number of cycles]
t_{RCD}: ACTIVE to READ/WRITE delay time [Number of cycles]

If RFP = 64 ms, ROWS = 4,096, f_{CLK} = 20 MHz, CL = 2, t_{RP} = 2, and t_{RCD} = 2, for example, the value to set is calculated as follows:

$$AURCO \leq \frac{0.064}{4{,}096} \times 20{,}000{,}000 - 2 - 2 - 2 \times 2 - 2 - 3 = 299$$

Therefore, set a value equal to or less than 299 (0x12b) to AURCO[11:0].

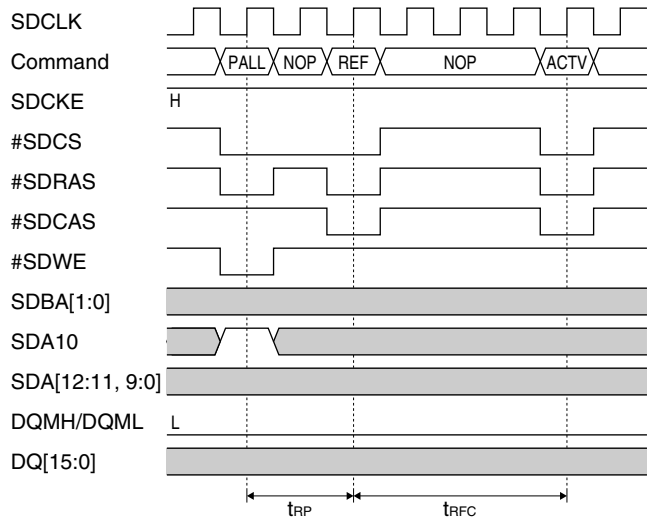


Figure 10.5.5.1 Auto-Refresh

Self-refresh

Self-refresh uses the self-refresh function of the SDRAM and does not require any clocks during the refresh period, thus helping to reduce power consumption. This self-refresh function is also used for data retention in the power-down mode. To cause the SDRAM to be self-refreshed, set SELEN/SDRAMC_REF register to 1. This enables the SDRAMC to send the self-refresh command (which sets the SDCKE output to low) to the SDRAM.

The command is actually sent a certain time after accessing or auto-refreshing the SDRAM, so the SDRAMC contains a self-refresh counter to count this time. The counter counts on SDCLK clock edges, and when the designated count is reached, the SDRAMC sends the self-refresh command to the SDRAM. When an SDRAM access or auto-refresh command is issued, the counter is reset and starts counting again. The counter value can be specified in a range of 1 to 127 using the SELCO[6:0]/SDRAMC_REF register. Do not set the counter to 0 when the self-refresh function is enabled.
If an access to the SDRAM occurs while the SDRAM is in self-refresh mode, SDCKE is reset to high, thereby self-refresh mode is canceled. After the SDRAM access has finished, the SDRAMC sends another self-refresh command when the designated count is reached again. When the auto-refresh command is issued or an SDRAM access occurs, the counter will restart if the self-refresh command has not been sent to the SDRAM. Therefore, the self-refresh counter value to be set must be smaller than the auto-refresh counter value.

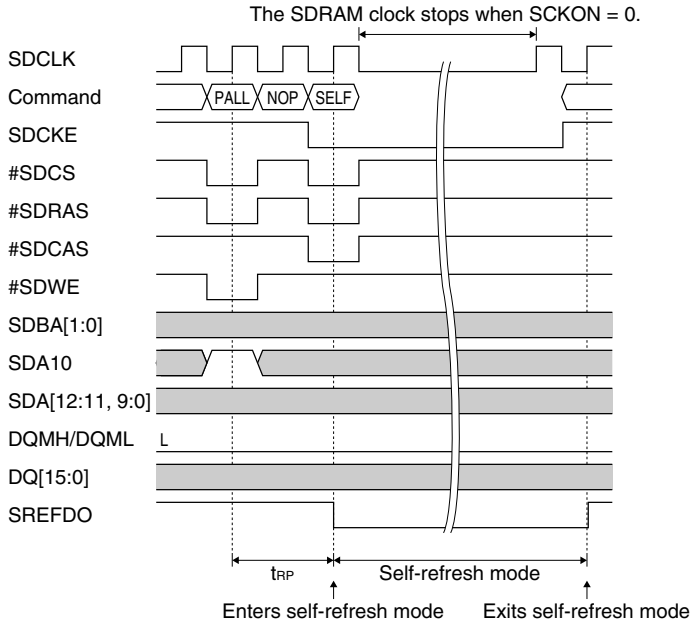


Figure 10.5.5.2 Self-Refresh

During self-refresh (while SDCKE = low), the SREFDO/SDRAMC_REF register remains 0. Therefore, it is possible to determine whether or not self-refresh is in operation by reading this status bit.
Furthermore, SDRAM clock output during self-refresh can be turned off in order to reduce power consumption by setting the SCKON/SDRAMC_REF register to 0.

To cancel self-refresh mode

Perform the following procedure to cancel self-refresh mode:

1. Disable self-refresh mode by clearing SELEN to 0.
2. Read data from an SDRAM address and then write the read value to the same address (any address can be specified).
3. Read SREFDO to check if self-refresh mode is canceled (SREFDO = 1).

Note that auto-refresh mode cannot be canceled.

Note: Be sure to avoid setting SDON/SDRAMC_INIT register to 0 (SDRAMC disabled) during self-refreshing. Before disabling the SDRAMC, always make sure the SDRAMC is not in self-refresh mode.

10.5.6 Power-Down Mode

The S1C33L27 supports two power-down modes for the C33 PE Core (HALT and SLEEP).

HALT mode

The SDRAM clock will be supplied in HALT mode, if it is not disabled in normal mode.
The SDRAM clock can be stopped before executing the halt instruction by controlling the CMU. To maintain data in the SDRAM during HALT status with no SDRAM clock supplied, place the SDRAM in self-refresh mode by setting SELEN/SDRAMC_REF register to 1 before disabling the SDRAM clock supply.

SLEEP mode

In SLEEP mode, the SDRAM can be turned off to reduce power consumption by the following procedure:

1. If the CPU runs with the program stored in the SDRAM, it must be changed to a program located in the built-in RAM or a memory other than the SDRAM.
2. Turn the SDRAM power off.
3. Switch the ports used for the SDRAM to general-purpose I/O ports.
4. Drive the data and address buses to low.
5. Set SDON/SDRAMC_INIT register to 0 to disable the SDRAMC.
6. Execute the slp instruction.

Perform the following procedure when the CPU wakes up from SLEEP status:

1. The CPU wakes up from SLEEP status.
2. Configure the port functions for the SDRAM.
3. Release the data and address buses from forced low driving.
4. Turn the SDRAM power on.
5. Wait at least 100 or 200 µs for the SDRAM to be stabilized, according to the SDRAM specifications.
6. Set SDON/SDRAMC_INIT register to 1 to enable the SDRAMC.
7. Initialize the SDRAMC.

10.6 Data Queue Buffer (DQB)

The bus controller of the S1C33L27 also includes a data queue buffer (DQB) to increase the C33 PE Core memory access performance.

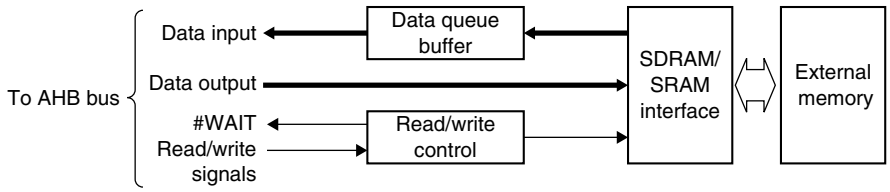


Figure 10.6.1 Data Queue Buffer

The DQB is a 4 × 16-bit buffer used as an instruction/data buffer for reading from the SDRAM or the external memories of the SRAMC.
The DQB acts as a pure read buffer for storing all data read from the SDRAM or external memories of the SRAMC, regardless of whether the target is an instruction or data. Note that the DQB cannot be disabled.
Table 10.6.1 lists the DQB status corresponding to the bus operation for the SDRAM or the external memories of SRAMC.

Table 10.6.1 Correspondence between DQB Status and Bus Operation

Bus operation	DQB status
CPU instruction fetch	Enabled
CPU vector fetch	Enabled
CPU data read	Enabled
CPU data write	Disabled
CPU stack read	Enabled
CPU stack write	Disabled
DMAC data read	Enabled
DMAC data write	Disabled

If the CPU executes writing to the address of data buffered in the DQB, the buffered data concerned are flushed. Initial resetting also resets and empty the DQB.

10.7 Control Register Details

Table 10.7.1 List of SDRAMC Registers

Address	Register name		Function
0x302200	SDRAMC_INIT	SDRAM Initialization Register	Enable SDRAMC and control SDRAM initialization
0x302204	SDRAMC_CFG	SDRAM Configuration Register	Set SDRAM size and timing parameters
0x302208	SDRAMC_REF	SDRAM Refresh Control Register	Control SDRAM refresh
0x302210	SDRAMC_APP	SDRAM Application Configuration Register	Set CAS latency

The following describes each SDRAMC register.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

SDRAM Initialization Register (SDRAMC_INIT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SDRAM Initialization Register (SDRAMC_INIT)	**0x302200** (32 bits)	D31–5	**–**	reserved	–				–	–	0 when being read.
		D4	**SDON**	SDRAM controller enable	1	Enable	0	Disable	0	R/W	
		D3	**INIDO**	SDRAM initialization status	1	Finished	0	Busy	0	R	
		D2	**INIMRS**	MRS command enable for init.	1	Enable	0	Disable	0	R/W	
		D1	**INIPRE**	PALL command enable for init.	1	Enable	0	Disable	0	R/W	
		D0	**INIREF**	REF command enable for init.	1	Enable	0	Disable	0	R/W	

D[31:5] Reserved

D4 SDON: SDRAM Controller Enable Bit

Enables the SDRAM controller.

1 (R/W): Enabled

0 (R/W): Disabled (default)

When SDON is set to 1, the SDRAM controller activates and outputs the SDRAM clock from the SDCLK pin. Before setting SDON to 1, be sure to start SDRAMC clock supply to the SDRAM controller.

Note: Be sure to avoid setting SDON to 0 (SDRAMC disabled) during self-refreshing. Before disabling the SDRAMC, always make sure the SDRAMC is not in self-refresh mode.

D3 INIDO: SDRAM Initialization Status Bit

Indicates that the SDRAM has finished initialization (Mode Register Set).

1 (R): Initialization finished

0 (R): Before initialization (default)

INIDO is set to 1 when the initialization sequence is completed. Make sure that INIDO is set to 1 before the SDRAM is accessed.

D2 INIMRS: MRS Command Enable for Initialization Bit

Enables to output the MRS (Mode Register Set) command for initializing the SDRAM.

1 (R/W): Enabled

0 (R/W): Disabled (default)

In order to initialize the SDRAM, the PALL (Precharge All), REF (Auto-Refresh), and MRS (Mode Register Set) commands must be executed sequentially. Note that the initialization sequence depends on the SDRAM used. Refer to the specification of the SDRAM to be used for the initialization sequence.
Example 1: PALL → REF → REF → MRS (→ EMRS)
Example 2: PALL → MRS → REF → REF (→ REF → REF → REF → REF → REF → REF)

To execute the MRS/EMRS (Mode Register Set/Extended Mode Register Set) command, write 0x14 to this register to set INIMRS to 1. Then write any data to a specific address of SDRAM shown below according to the CAS latency (MRS) or extended mode parameters (EMRS).

Table 10.7.2 Data Write Address to Execute the MRS/EMRS Command

CPU address	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1
SDRAM address	BA1	BA0	SDA12	SDA11	SDA10	SDA9	SDA8	SDA7	SDA6	SDA5	SDA4	SDA3	SDA2	SDA1	SDA0
MRS	Mode		Reserved			WB	Test mode		CAS latency			BT	Burst length		
CAS latency = 1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1
CAS latency = 2	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1
CAS latency = 3	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1
EMRS	Mode		Reserved							DS	TCSR		PASR		
	1	0	0	0	0	0	0	0	0	See the SDRAM specification.					

For example, to execute an MRS command with CAS latency = 2, write any value to address 0x10000442 (when the SDRAM is mapped to Area 19) after writing 0x14 to the SDRAMC_INIT register.

Note: The CAS latency specified in the MRS command must be the same as the setting for CAS[1:0]/SDRAMC_APP register.

D1 INIPRE: PALL Command Enable for Initialization Bit

Enables to output the PALL (Precharge All) command for initializing the SDRAM.
1 (R/W): Enabled
0 (R/W): Disabled (default)

To execute the PALL (Precharge All) command, write 0x12 to this register to set INIPRE to 1. Then write any data to any address in the SDRAM. This dummy write is required as the trigger to send the PALL command to the SDRAM. See INIMRS for the initialization sequence.

D0 INIREF: REF Command Enable for Initialization Bit

Enables to output the REF (Auto-Refresh) command for initializing the SDRAM.
1 (R/W): Enabled
0 (R/W): Disabled (default)

To execute the REF (Auto-Refresh) command, write 0x11 to this register to set INIREF to 1. Then write any data to any address in the SDRAM. This dummy write is required as the trigger to send the REF command to the SDRAM. See INIMRS for the initialization sequence.

When executing the REF command twice or more, insert the nop instruction between the executions.
REF command execution → nop instruction execution → REF command execution (→ REF → nop → REF...)

Notes:
- The SDRAM timing parameters set in the SDRAMC_CFG register is disabled when the initialization sequence is executed. Therefore, enough number of nop instructions must be executed to satisfy the SDRAM timings.
- After the initial sequence commands are executed, the command enable bit must be set to 0. Write 0x10 to the SDRAMC_INIT register after the last initialization command has been executed.
- The self-refresh function must be disabled until the SDRAM has finished initialization.

SDRAM Configuration Register (SDRAMC_CFG)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
SDRAM Configuration Register (SDRAMC_CFG)	0x302204 (32 bits)	D31–14	–	reserved	–		–	–	0 when being read.
		D13–12	T24NS[1:0]	Number of SDRAM tRP and tRCD cycles	T24NS[1:0]	# of cycles	0x0	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D11	–	reserved	–		–	–	0 when being read.
		D10–8	T60NS[2:0]	Number of SDRAM tRAS cycles	T60NS[2:0]	# of cycles	0x0	R/W	
					0x7	8 cycles			
					0x6	7 cycles			
					:	:			
					0x1	2 cycles			
					0x0	1 cycle			
		D7–4	T80NS[3:0]	Number of SDRAM tRC, tRFC and txSR cycles	T80NS[3:0]	# of cycles	0xe	R/W	
					0xc	13 cycles			
					0x8	9 cycles			
					0x4	5 cycles			
					0x0	1 cycle			
					Other	reserved			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	ADDRC[2:0]	SDRAM address configuration	ADDRC[2:0]	Configuration	0x0	R/W	Do not set to 0x4.
					0x7	512M bits			
					0x6	128M bits x 2			
					0x5	64M bits x 2			
					0x4	reserved			
					0x3	256M bits			
					0x2	128M bits			
					0x1	64M bits			
					0x0	16M bits			

D[31:14] Reserved

D[13:12] T24NS[1:0]: Number of SDRAM tRP and tRCD Cycles Bits

Sets the tRP and tRCD SDRAM timing parameters.

- tRP PRECHARGE to ACTIVE command period
- tRCD ACTIVE to READ/WRITE delay time

Table 10.7.3 tRP and tRCD Settings

T24NS[1:0]	tRP, tRCD
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

D11 Reserved

D[10:8] T60NS[2:0]: Number of SDRAM tRAS Cycles Bits

Sets the tRAS SDRAM timing parameter.

- tRAS ACTIVE to PRECHARGE command period

Table 10.7.4 tRAS Settings

T60NS[2:0]	tRAS
0x7	8 cycles
0x6	7 cycles
:	:
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

D[7:4] T80NS[3:0]: Number of SDRAM tRC, tRFC, and txSR Cycles Bits

Sets the tRC, tRFC and txSR SDRAM timing parameters.

- tRC ACTIVE to ACTIVE command cycle time
- tRFC Auto-refresh cycle time
- txSR Self-refresh end to ACTIVE command period

Table 10.7.5 t_{RC}, t_{RFC}, and t_{XSR} Settings

T80NS[3:0]	t_{RC}, t_{RFC}, t_{XSR}
0xc	13 cycles
0x8	9 cycles
0x4	5 cycles
0x0	1 cycle
Other	Reserved

(Default: 0xe)

D3 Reserved

D[2:0] ADDRC[2:0]: SDRAM Address Configuration Bits

Selects SDRAM size and chip configuration. This selection also sets up the bank size, column address size (page size), and row address size.

Table 10.7.6 SDRAM Size Selections

ADDRC[2:0]	Number of banks	Row size	Column size	SDRAM chip configuration	Memory size
0x7	4	8K	1K	32M × 16 bits × 1	64M bytes
0x6	4	4K	1K	16M × 8 bits × 2	32M bytes
0x5	4	4K	512	8M × 8 bits × 2	16M bytes
0x4	Reserved (Do not set ADDRC[2:0] to 0x4.)				
0x3	4	8K	512	16M × 16 bits × 1	32M bytes
0x2	4	4K	512	8M × 16 bits × 1	16M bytes
0x1	4	4K	256	4M × 16 bits × 1	8M bytes
0x0	2	2K	256	1M × 16 bits × 1	2M bytes

(Default: 0x0)

SDRAM Refresh Register (SDRAMC_REF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SDRAM Refresh Control Register (SDRAMC_REF)	0x302208 (32 bits)	D31–26	–	reserved	–	–	–	0 when being read.
		D25	SREFDO	SDRAM self-refresh status	1 Finished 0 Busy	0	R	
		D24	SCKON	SDRAM clock during self-refresh	1 Enable 0 Disable	0	R/W	
		D23	SELEN	SDRAM self-refresh enable	1 Enable 0 Disable	0	R/W	
		D22–16	SELCO[6:0]	SDRAM self-refresh counter	0x0 to 0x7f	0x7f	R/W	
		D15–12	–	reserved	–	–	–	0 when being read.
		D11–0	AURCO[11:0]	SDRAM auto-refresh counter	0x0 to 0xfff	0x8c	R/W	

D[31:26] Reserved

D25 SREFDO: SDRAM Self-Refresh Status Bit

Indicates the SDRAM self-refresh status.

1 (R): Self-refresh has finished

0 (R): Self-refresh mode (default)

SREFDO is set to 0 while the SDRAM is placed into self-refresh mode. Otherwise, SREFDO is set to 1. Before entering the SLEEP mode, always be sure to read this bit using a program stored elsewhere (i.e., not in the SDRAM) to confirm that the SDRAM is in self-refresh mode.

D24 SCKON: SDRAM Clock During Self-Refresh Bit

Specifies whether to stop the SDRAM clock during self-refresh or not.

1 (R/W): Enabled (output continued)

0 (R/W): Disabled (output disabled) (default)

Writing 0 to SCKON disables the SDRAM clock output from the SDCLK pin while the SDRAM is in self-refresh mode. This helps to reduce the current consumption.

If SCKON is 1, the SDRAM clock is always output from the SDCLK pin even if the SDRAM is in self-refresh mode.

D23 SELEN: SDRAM Self-Refresh Enable Bit

Enables the self-refresh control function of the SDRAM.

1 (R/W): Enabled

0 (R/W): Disabled (Default)

Writing 1 to SELEN enables the SDRAMC to start self-refreshing the SDRAM (by setting SDCKE output low). Note that self-refreshing of the SDRAM actually begins a certain time after accessing or auto-refreshing the SDRAM. The duration of this elapsed time is defined by the number of clock cycles in SELCO[6:0].

To cancel self-refresh mode, perform the following procedure:

1. Disable self-refresh mode by clearing SELEN to 0.
2. Read data from an SDRAM address and then write the read value to the same address (any address can be specified).
3. Read SREFDO to check if self-refresh mode is canceled (SREFDO = 1).

D[22:16] SELCO[6:0]: SDRAM Self-Refresh Counter Bits

Sets the value for the self-refresh counter. (Default: 0x7f)

If SELEN is set to 1 (self-refresh enabled), the self-refresh counter starts counting up on the SDCLK clock edges beginning with 0 after accessing or auto-refreshing the SDRAM. When the count specified here is reached, the SDCKE output is pulled low, causing the SDRAM to start self-refreshing. If an access to the SDRAM occurs during self-refresh, SDCKE returns high, thereby self-refresh mode is canceled.

D[15:12] Reserved

D[11:0] AURCO[11:0]: SDRAM Auto-Refresh Counter Bits

Sets the auto-refresh counter value. (Default: 0x8c)

The auto-refresh counter counts up on the SDCLK clock edges beginning with 0, and when the count specified here is reached, the SDRAM controller sends an auto-refresh command. The counter is reset at that point, and starts counting the next refresh period. The counter is also reset by self-refresh.

The value calculated from the equation below is the maximum count that can be set.

$$AURCO \leq \frac{RFP}{ROWS} \times f_{CLK} - BL - CL - 2 \times t_{RP} - t_{RCD} - 3$$

RFP: Maximum refresh period [s]
ROWS: Row address size
f_{CLK}: SDCLK clock frequency [Hz]
BL: Burst length (= 2)
CL: CAS latency
t_{RP}: PRECHARGE command period [Number of cycles]
t_{RCD}: ACTIVE to READ/WRITE delay time [Number of cycles]

SDRAM Application Configuration Register (SDRAMC_APP)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
SDRAM Application Configuration Register (SDRAMC_APP)	0x302210 (32 bits)	D31–6	–	reserved	–		–	–	0 when being read.
		D5	–	reserved	–		0	–	Do not set to 1.
		D4	–	reserved	–		–	–	0 when being read.
		D3–2	CAS[1:0]	CAS latency setup	CAS[1:0]	CAS latency	0x2	R/W	
					0x3	3			
					0x2	2			
					0x1	1			
					0x0	reserved			
		D1	–	reserved	–		–	–	0 when being read.
		D0	–	reserved	–		0	–	Do not set to 1.

D[31:6] Reserved

D5 Reserved (Do not set to 1.)

D4 Reserved

D[3:2] CAS[1:0]: CAS Latency Setup Bits

Sets the CAS latency.

CAS latency refers to the number of SDCLK clock cycles counted until data is output from the SDRAM after issuing the READ command.

Table 10.7.7 CAS Latency Settings

CAS[1:0]	CAS latency
0x3	3
0x2	2
0x1	1
0x0	Reserved

(Default: 0x2)

D1 Reserved

D0 Reserved (Do not set to 1.)

10.8 Precautions

If the operating clock (SYSCLK) is stopped while the SDRAM is being accessed, a system failure may occur due to stoppage of the SDRAM operation in uncontrolled status. The following operations stop SYSCLK. Do not perform these operations when the SDRAM may be accessed.
- Placing the S1C33L27 into SLEEP status
- Disabling the clock supply to the SDRAMC module

Besides from the CPU, the SDRAM can be accessed from the DMAC (if DMA transfers are enabled toward the SDRAM). In this case, before performing the above operations, stop the DMAC to disable its access to the SDRAM.

11 Host Interface (HIF)

11.1 HIF Overview

The S1C33L27 includes a host interface (HIF) for connecting to an external host processor. This allows use as a controller such as an LCDC or as a sub-processor. The HIF includes an AHB interface and allows access by an external host processor to all S1C33L27 internal peripheral circuits as well as system memory (excluding RAM in Area 0).
The main features are listed below.

- Supports 80-based processors.
- Data bus: 8 bits (HIF_D[7:0]) or 16 bits (HIF_D[15:0]) selectable
  Successive reads and writes enable byte, half-word, and word access to the inside of the S1C33L27. (Either little-endian or big-endian format can be selected.)
- Address bus: 4 bits (HIF_A[3:0]), specifies the I/O (HIF register) address.
  The S1C33L27 internal address can be specified using an address register that allows selection of fixed address or automatic address increment.
- Bus control signals: Chip enable (#HIF_CS), Read (#HIF_RD), Write (#HIF_WR)
- A wait request signal (#HIF_WAIT) can be output to the external host processor.
- An interrupt request signal (#HIF_INT) can be output to the external host processor.
- The external host processor can generate interrupts to the C33 PE Core by setting a register (selection of maskable interrupts or non-maskable interrupts (NMI) enabled).
- Incorporates eight semaphore registers and four scratchpad registers that can be accessed from both the external host processor and the C33 PE core.
- Supports booting from the external host processor. (see Appendix)

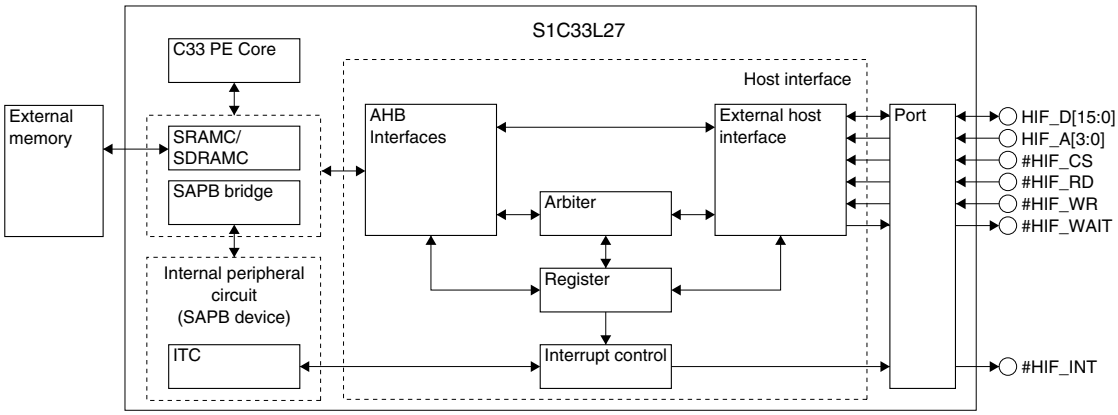


Figure 11.1.1 HIF Module Block Diagram

Note: The priorities of the AHB bus masters are set as follows:
1. LCDC → 2. DMAC → 3. HIF → 4. C33 PE (cache)

11.2 HIF Pins

Table 11.2.1 lists the HIF input/output pins.

Table 11.2.1 List of HIF Pins

Pin name	I/O	Qty	Function
HIF_D[15:0]/HIF_D[7:0]	I/O	16/8	External host processor data bus D[15:0] (16-bit I/F mode) or D[7:0] (8-bit I/F mode)
HIF_A[3:0]	I	4	External host processor address bus A[3:0]
#HIF_CS	I	1	External host processor chip select signal input
#HIF_RD	I	1	External host processor read signal input
#HIF_WR	I	1	External host processor write signal input
#HIF_WAIT	O	1	Wait request signal output
#HIF_INT	O	1	Interrupt request signal output

The HIF input/output pins are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as HIF input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

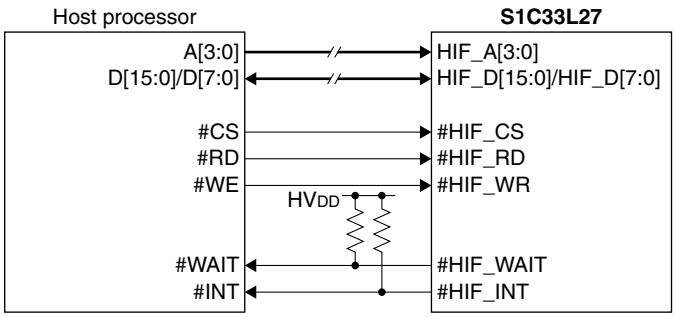


Figure 11.2.1 Host Processor (80-series) Connection Example

11.3 HIF Registers

11.3.1 HIF Register Assignment

The HIF module incorporates registers that allow access only from an external host processor and shared registers that enable access from both an external host processor and the C33 PE Core.
The external host processor views the S1C33L27 as a single I/O device and accesses the registers in the HIF module using a chip enable signal and a 4-bit address (A[3:0]). The S1C33L27 system memory and internal peripheral modules other than the HIF module are accessed via registers in the HIF module.
The HIF registers viewed by the external host processor are assigned to addresses 0x0 to 0xf. Two of these registers are used as an index and as a data port to access additional special registers.
The shared registers (including special registers) viewed by the C33 PE Core are assigned to addresses 0x300901 to 0x300919.

Table 11.3.1 HIF Registers

Host address	Index (*1)	Register	Function	S1C33L27 Internal address
0x0	–	HIF Direct Interface Byte Data (HIF_DRIF_B) Register	Data direct read/write register (for C33 byte access only)	Cannot be accessed
0x1	–	HIF Direct Interface Half Word Data (HIF_DRIF_HW) Register	Data direct read/write register (for C33 half-word access only)	
0x2	–	HIF Direct Interface Word Data (HIF_DRIF_W) Register	Data direct read/write register (for C33 word access only)	
0x3	–	HIF Indirect Interface Data (HIF_IDIF_DAT) Register	Data indirect read/write register	
0x4	–	HIF Indirect Interface Control (HIF_IDIF_CTRL) Register	Data indirect read/write control	
0x6	–	HIF Control (HIF_CTRL) Register	Setting of access conditions, interrupt permission to the host	
0x8	–	HIF Address (HIF_ADDR0) Register 0	Address bits A[7:0] (8-bit I/F mode) Address bits A[15:0] (16-bit I/F mode)	
0x9	–	HIF Address (HIF_ADDR1) Register 1	Address bits A[15:8] (8-bit I/F mode) Address bits A[31:16] (16-bit I/F mode)	
0xa	–	HIF Address (HIF_ADDR2) Register 2	Address bits A[23:16] (8-bit I/F mode)	
0xb	–	HIF Address (HIF_ADDR3) Register 3	Address bits A[31:24] (8-bit I/F mode)	
0xc	–	HIF Special Register Index (HIF_SP_IDX) Register	Designation of special registers	
0xd		HIF Special Register Data (HIF_SP_DAT) Register	Special register data	
	0x1	HIF Extended Control (HIF_EXTCTRL1) Register 1 (*2)	Interrupt mode settings	0x300901
	0x8	HIF Semaphore (HIF_SMPH0) Register 0 (*2)	Semaphore 0	0x300908
	0x9	HIF Semaphore (HIF_SMPH1) Register 1 (*2)	Semaphore 1	0x300909
	0xa	HIF Semaphore (HIF_SMPH2) Register 2 (*2)	Semaphore 2	0x30090a
	0xb	HIF Semaphore (HIF_SMPH3) Register 3 (*2)	Semaphore 3	0x30090b
	0xc	HIF Semaphore (HIF_SMPH4) Register 4 (*2)	Semaphore 4	0x30090c
	0xd	HIF Semaphore (HIF_SMPH5) Register 5 (*2)	Semaphore 5	0x30090d
	0xe	HIF Semaphore (HIF_SMPH6) Register 6 (*2)	Semaphore 6	0x30090e
	0xf	HIF Semaphore (HIF_SMPH7) Register 7 (*2)	Semaphore 7	0x30090f
	0x10	HIF Scratchpad (HIF_SCRATCH0) Register 0 (*2)	Scratchpad register 0	0x300910
	0x11	HIF Scratchpad (HIF_SCRATCH1) Register 1 (*2)	Scratchpad register 1	0x300911
	0x12	HIF Scratchpad (HIF_SCRATCH2) Register 2 (*2)	Scratchpad register 2	0x300912
	0x13	HIF Scratchpad (HIF_SCRATCH3) Register 3 (*2)	Scratchpad register 3	0x300913
0xe	–	HIF Flag (HIF_FLAG0) Register 0	Slave flags	0x300918
0xf	–	HIF Flag (HIF_FLAG1) Register 1	Host flags	0x300919

Registers that can only be accessed by the host processor (all other registers are shared registers)

*1 Index of special registers that the host specifies with the HIF_SP_IDX register (0xc) (specified registers are accessed by the host through the HIF_SP_DAT register (0xd)). "–" indicates that these addresses are directly accessible by the host addresses.

*2 Special register

The registers will be described in detail on the following pages.

If a shared register has been accessed by both the external host processor and the C33 PE Core simultaneously, the arbiter gives the host priority; access by the C33 PE Core is placed on hold until access by the host has been completed (the CPU enters the waiting state).

11.3.2 Semaphore Registers

The HIF module incorporates eight semaphore registers (each one a one-bit semaphore) that can be accessed by both the external host processor and C33 PE Core to enable easy program setting/browsing of the status of use of a specified resource (up to eight).

To access a semaphore register from the external host processor, first write the index (0x8 to 0xf) specifying the semaphore register to the HIF_SP_IDX register (host address 0xc), then access the HIF_SP_DAT register (host address 0xd).

To access a semaphore register from the C33 PE Core, perform access in the same manner normally used to access an I/O register.

Table 11.3.2.1 Semaphores

Semaphore	Index to set at HIF_SP_IDX	S1C33 internal address
SMPH0/HIF_SMPH0 register	0x8	0x300908
SMPH1/HIF_SMPH1 register	0x9	0x300909
SMPH2/HIF_SMPH2 register	0xa	0x30090a
SMPH3/HIF_SMPH3 register	0xb	0x30090b
SMPH4/HIF_SMPH4 register	0xc	0x30090c
SMPH5/HIF_SMPH5 register	0xd	0x30090d
SMPH6/HIF_SMPH6 register	0xe	0x30090e
SMPH7/HIF_SMPH7 register	0xf	0x30090f

A semaphore (SMPH*x*) is assigned to bit 0 of each register.
The module (external host processor or S1C33L27 internal program) that will use the resource reads the corresponding SMPH*x*. If SMPH*x* was 0 (can be used), the resource is used and processing continues. SMPH*x* automatically changes to 1 (in use) when read. Accordingly, if SMPH*x* was 1, this indicates that another module is using the resource. In this case, resource use is placed on hold until SMPH*x* reverts to 0. When a module has finished using a resource, 1 must be written to SMPH*x* to reset it to 0.

11.3.3 Scratchpad Registers

The HIF module incorporates four 8-bit scratchpad registers. These registers have no effect on the operation of the host interface and can be used for temporary storage of data or for communication between the external host processor and the C33 PE Core.
To access a scratchpad from an external host processor, first write the index (0x10 to 0x13) specifying the scratchpad register to the HIF_SP_IDX register (host address 0xc), then access the HIF_SP_DAT register (host address 0xd).
To access a scratchpad register from the C33 PE Core, perform access in the same manner normally used to access an I/O register.

Table 11.3.3.1 Scratchpad Registers

Scratchpad register	Index to set at HIF_SP_IDX	S1C33 internal address
HIF_SCRATCH0 register	0x10	0x300910
HIF_SCRATCH1 register	0x11	0x300911
HIF_SCRATCH2 register	0x12	0x300912
HIF_SCRATCH3 register	0x13	0x300913

11.3.4 Flag Registers

The following flags are assigned to the two flag registers (HIF_FLAG0 register and HIF_FLAG1 register).

SFLAG[3:0]

The SFLAG[3:0]/HIF_FLAG0 register bits are general-purpose flags that can be set/reset by the C33 PE Core. Only reading and resetting can be performed from the external host processor; settings cannot be made. These flags can be used for various purposes, such as reporting the status of the S1C33L27 to the external host processor.
To manipulate the flag using the C33 PE Core, write 1 (set) or 0 (reset).
To reset the flag from the external host processor, write 1 (see the note below).

HFLAG[2:0]

The HFLAG[2:0]/HIF_FLAG1 register bits are general-purpose flags that can be set/reset by the external host processor. Only reading and resetting can be performed from the C33 PE core; settings cannot be made. These flags can be used for various purposes, such as reporting the status of the external host processor to the S1C33L27.
To manipulate the flag, write 1 (set) or 0 (reset) from the external host processor.
The flags can also be reset by writing 1 from the C33 PE Core. However, note that write protection for the HIF_FLAG1 register must be cleared (see the note below).

In addition to the above flags, SINT/HIF_FLAG0 register for generating a software interrupt from the S1C33L27 to the external host processor and HINT/HIF_FLAG1 register for generating an S1C33L27 HIF interrupt or an NMI from the external host processor are also provided. For more information on these flags, see Section 11.6, "Interrupts."

Notes: • To prevent errant operations resulting from unnecessary overwrites, the default setting for the HIF_FLAG0 (D[4:0]) and HIF_FLAG1 (D[3:0]) registers (host addresses 0xe and 0xf) are the write-protected state. To overwrite these two registers, the write protection must be cleared by writing 0x96 to the HIF_FLAG0 register (host address 0xe). (This can only be done from the external host processor.)
To enable the write protection of these two registers, write 0xff to the HIF_FLAG0 register (host address 0xe). (This can only be done from the external host processor.)

- To reset the SFLAG[3:0] flags and SINT, be sure to write PROTECT to 0 when writing to the HIF_FLAG0 register. The flags will not be reset if PROTECT is 1.

11.4 Accessing the S1C33L27 from the Host Processor

Described below are the settings and procedures required to access the S1C33L27's internal peripheral circuits and system memory from the external host processor.

11.4.1 Setting Access Conditions

Selecting the data bus width (8 bits/16 bits)

Select the data bus width of the host interface using HIFSEL/HIF_CTRL register. Writing 0 to HIFSEL configures the data bus to 8 bits (HIF_D[7:0]); writing 1 to HIFSEL configures the data bus to 16 bits (HIF_D[15:0]). The HIF register bit width is also set to 8- or 16 bits according to this selection (except the special registers).

Enabling AHB bus access

The HIF module incorporates an AHB interface and functions as an AHB bus master. The external host processor accesses the S1C33L27's internal peripheral circuits and system memory via the HIF registers and the AHB interface.

Since the AHB interface is initially disabled, it must be enabled by writing 1 to AHB_EN/HIF_CTRL register before the external host processor can access the S1C33L27.

Selecting endian mode

The HIF module provides an 8- or 16-bit data bus as an interface to an external host processor. Although data transfers to/from the external host processor are performed in 8/16-bit units, byte access, half-word access, and word access are enabled inside the S1C33L27 (for detailed information on this control method, see Section 11.4.2, "Direct Interface," and Section 11.4.3, "Indirect Interface").

To perform half-word or word access, select the endian mode using ENDIAN/HIF_CTRL register. Set ENDIAN to 0 (default) for little endian (transfer with the low-order byte first) and to 1 for big endian (transfer with the high-order byte first).

Depending on the data bus width and internal access size, the external host processor performs reads/writes once, twice, or four times. The following table shows the access data bits for each instance according to the endian mode selected.

Table 11.4.1.1 Relationship Between Addresses and Data Bits According to Selected Endian Mode

Data bus width	Internal access size	Internal address low-order bit		Internal read/write bit		Access number of times/Number
		A1	A0	Little endian	Big endian	
8 bits	Byte	*	*	D[7:0]	D[7:0]	One time
	Half-word	*	0	D[7:0]	D[15:8]	First time
		*	1	D[15:8]	D[7:0]	Second time
	Word	0	0	D[7:0]	D[31:24]	First time
		0	1	D[15:8]	D[23:16]	Second time
		1	0	D[23:16]	D[15:8]	Third time
		1	1	D[31:24]	D[7:0]	Fourth time

Data bus width	Internal access size	Internal address low-order bit		Internal read/write bit		Access number of times/Number
		A1	A0	Little endian	Big endian	
16 bits	Byte	*	*	D[7:0]	D[7:0]	One time
	Half-word	*	0	D[15:0]	D[15:0]	One time
	Word	0	0	D[15:0]	D[31:16]	First time
		1	0	D[31:16]	D[15:0]	Second time

Setting the address and selecting address control mode

The S1C33L27 memory address accessed by the external host processor is set to the HIF_ADDR*x* registers (host address 0x8 to 0xb).

Table 11.4.1.2 Address Settings

Data bus width	Register	Register address (host address)	C33 address bits specified with the register
8 bits	HIF_ADDR0 register	0x8	A[7:0]
	HIF_ADDR1 register	0x9	A[15:8]
	HIF_ADDR2 register	0xa	A[23:16]
	HIF_ADDR3 register	0xb	A[31:24]
16 bits	HIF_ADDR0 register	0x8	A[15:0]
	HIF_ADDR1 register	0x9	A[31:16]

Both the direct interface mode and indirect interface mode (described later) use these registers.

Notes: • When accessing the S1C33L27 in half-word size, set a half-word boundary address (A0 = 0). When accessing in word size, set a word boundary address (A[1:0] = 0b00).

• If an access attempt is made for an address of a memory not actually installed, an invalid write or invalid read will occur.

The post-increment access function can be used if the external host processor accesses the S1C33L27 more than once in succession. ADDR_MD[1:0]/HIF_CTRL register is used for this selection.

Table 11.4.1.3 Address Control Mode

ADDR_MD[1:0]	Address control mode
0x3–0x2	Reserved
0x1	Post increment
0x0	Fixed address

(Default: 0x0)

If ADDR_MD[1:0] is 0x0 (default), the address set in the HIF_ADDR*x* register will not change, even following access.
If ADDR_MD[1:0] is set to 0x1, the low-order 16 bits of the address set in the HIF_ADDR*x* registers are incremented by the amount of the access size after access.
When the access is byte access: +1
When the access is half-word access: +2
When the access is word access: +4

This enables successive access to the memory area without altering the HIF_ADDR*x* register via software.

Note: The post-increment function is enabled only for the low-order 16 bits of the address (A[15:0]). Accordingly, successive accesses that exceed the 64-KB boundary address (0x*XXXX*ffff) are not possible.

11.4.2 Direct Interface

Using the direct interface function enables access to the address set at the HIF_ADDR*x* register (host address 0x8 to 0x9 or 0xb) simply by reading/writing to a data register. However, the external host processor must have an external wait request input pin to connect to the S1C33L27 #HIF_WAIT pin (active low).

The S1C33L27 provides three different data registers corresponding to the internal access size. Use the register matching the access size and following procedure for read/write operations.

8-bit interface

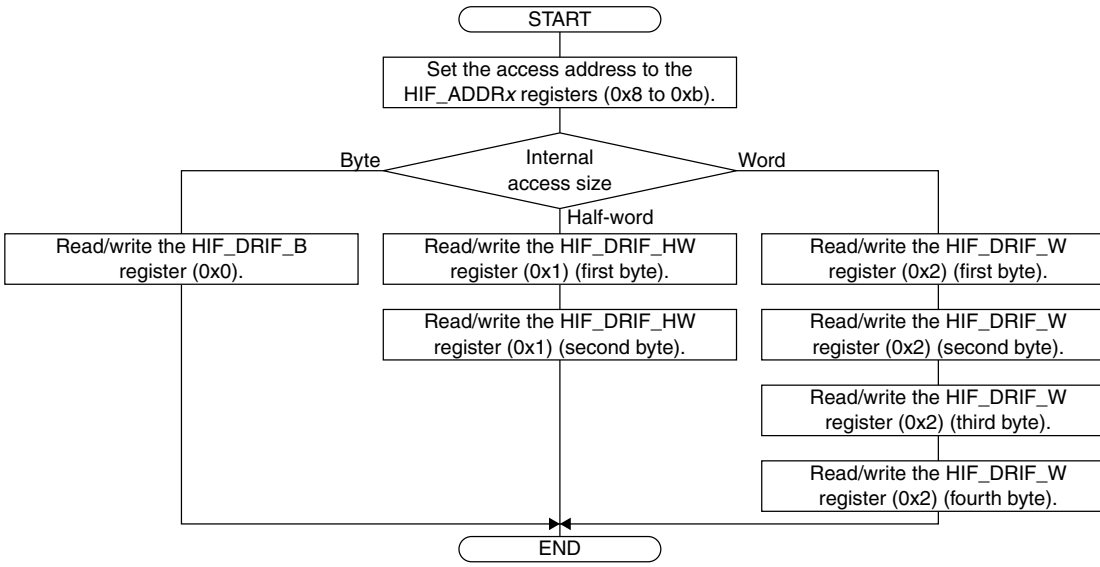


Figure 11.4.2.1 Reading/Writing Using the Direct Interface (8-bit interface)

Byte access (8-bit interface)

Read/write the HIF_DRIF_B register (host address 0x0) once.

Writing data

1. Set the address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
2. Write data D[7:0] to the HIF_DRIF_B register.
   The data sent from the host is written to the write data buffer, and a byte write bus operation is executed with the HIF module serving as the bus master.

Reading data

1. Set the address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
2. Read the HIF_DRIF_B register.
   A byte read bus operation is executed with the HIF module serving as the bus master. Once the read values are loaded into the read data buffer, the byte data is output to HIF_D[7:0].

Half-word access (8-bit interface)

Read/write the HIF_DRIF_HW register (host address 0x1) twice.

Writing data

1. Set the half-word address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
2. Write data D[7:0] (when little endian has been selected) or D[15:8] (when big endian has been selected) to the HIF_DRIF_HW register. The data sent from the host is written to the write data buffer.
3. Write data D[15:8] (when little endian has been selected) or D[7:0] (when big endian has been selected) to the HIF_DRIF_HW register. The data is written to the write data buffer, and a half-word write bus operation is executed with the HIF module serving as the bus master.

Reading data

1. Set the half-word address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers register (only when required).
2. Read the first byte data from the HIF_DRIF_HW register.
   A half-word read bus operation is executed with the HIF module serving as the bus master. When the half-word read value is loaded into the read data buffer, the data D[7:0] (when little endian has been selected) or D[15:8] (when big endian has been selected) is output to HIF_D[7:0].

3. Read the next byte data from the HIF_DRIF_HW register.
   The data D[15:8] (when little endian has been selected) or D[7:0] (when big endian has been selected) in the read data buffer is output to HIF_D[7:0].

Word access (8-bit interface)

Read/write the HIF_DRIF_W register (host address 0x2) four times.

Writing data

1. Set the word address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
2. Write data D[7:0] (when little endian has been selected) or D[31:24] (when big endian has been selected) to the HIF_DRIF_W register. The data sent from the host is written to the write data buffer.
3. Write data D[15:8] (when little endian has been selected) or D[23:16] (when big endian has been selected) to the HIF_DRIF_W register. The data sent from the host is written to the write data buffer.
4. Write data D[23:16] (when little endian has been selected) or D[15:8] (when big endian has been selected) to the HIF_DRIF_W register. The data sent from the host is written to the write data buffer.
5. Write data D[31:24] (when little endian has been selected) or D[7:0] (when big endian has been selected) to the HIF_DRIF_W register. The data is written to the write data buffer, and a word write bus operation is executed with the HIF module serving as the bus master.

Reading data

1. Set the word address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
2. Read the first byte data from the HIF_DRIF_W register.
   A word read bus operation is executed with the HIF module serving as the bus master. When the word read value is loaded into the read data buffer, the data D[7:0] (when little endian has been selected) or D[31:24] (when big endian has been selected) is output to HIF_D[7:0].
3. Read the second byte data from the HIF_DRIF_W register.
   The data D[15:8] (when little endian has been selected) or D[23:16] (when big endian has been selected) in the read data buffer is output to HIF_D[7:0].
4. Read the third byte data from the HIF_DRIF_W register.
   The data D[23:16] (when little endian has been selected) or D[15:8] (when big endian has been selected) in the read data buffer is output to HIF_D[7:0].
5. Read the fourth byte data from the HIF_DRIF_W register.
   The data D[31:24] (when little endian has been selected) or D[7:0] (when big endian has been selected) in the read data buffer is output to HIF_D[7:0].

#HIF_WAIT becomes active while data is being read/written, and the host enters wait status.

16-bit interface

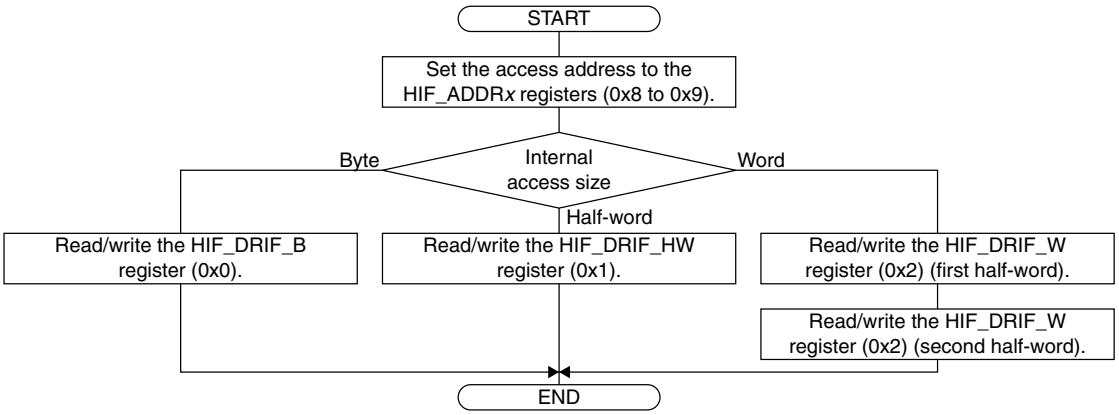


Figure 11.4.2.2 Reading/Writing Using the Direct Interface (16-bit interface)

Byte access (16-bit interface)

Read/write the HIF_DRIF_B register (host address 0x0) once.

Writing data

1. Set the address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Write data D[7:0] to the low-order 8 bits of the HIF_DRIF_B register. Note that the high-order 8 bits D[15:8] of the HIF_DRIF_B register are ineffective.
   The data sent from the host is written to the write data buffer, and a byte write bus operation is executed with the HIF module serving as the bus master.

Reading data

1. Set the address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Read the HIF_DRIF_B register.
   A byte read bus operation is executed with the HIF module serving as the bus master. Once the read values are loaded into the read data buffer, the byte data is output to HIF_D[7:0]. Note that the high-order 8 bits D[15:8] of the HIF_DRIF_B register are ineffective. The HIF_D[15:8] pins output 0.

Half-word access (16-bit interface)

Read/write the HIF_DRIF_HW register (host address 0x1) once.

Writing data

1. Set the half-word address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Write data D[15:0] to the HIF_DRIF_HW register.
   The data sent from the host is written to the write data buffer, and a half-word write bus operation is executed with the HIF module serving as the bus master.

Reading data

1. Set the half-word address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Read half-word data from the HIF_DRIF_HW register.
   A half-word read bus operation is executed with the HIF module serving as the bus master. Once the read values are loaded into the read data buffer, the half-word data is output to HIF_D[15:0].

Word access (16-bit interface)

Read/write the HIF_DRIF_W register (host address 0x2) twice.

Writing data

1. Set the word address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Write data D[15:0] (when little endian has been selected) or D[31:16] (when big endian has been selected) to the HIF_DRIF_W register. The data sent from the host is written to the write data buffer.
3. Write data D[31:16] (when little endian has been selected) or D[15:0] (when big endian has been selected) to the HIF_DRIF_W register. The data is written to the write data buffer, and a word write bus operation is executed with the HIF module serving as the bus master.

Reading data

1. Set the word address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Read the first half-word data from the HIF_DRIF_W register.
   A word read bus operation is executed with the HIF module serving as the bus master. When the word read value is loaded into the read data buffer, the data D[15:0] (when little endian has been selected) or D[31:16] (when big endian has been selected) is output to HIF_D[15:0].
3. Read the next half-word data from the HIF_DRIF_W register.
   The data D[31:16] (when little endian has been selected) or D[15:0] (when big endian has been selected) in the read data buffer is output to HIF_D[15:0].

#HIF_WAIT becomes active while data is being read/written, and the host enters wait status.

11.4.3 Indirect Interface

The indirect interface function is used when a processor lacking an external wait request input is connected. A data read/write register (HIF_IDIF_DAT register (host address 0x3)) and a bus operation control register (HIF_IDIF_CTRL register (host address 0x4)) differing from those for the direct interface are provided. The same registers as those used for the direct interface, HIF_ADDR*x* registers (host address 0x8 to 0xb), are used for address settings. When using the indirect interface, use the following procedure for read/write operations.

Writing data via 8-bit interface

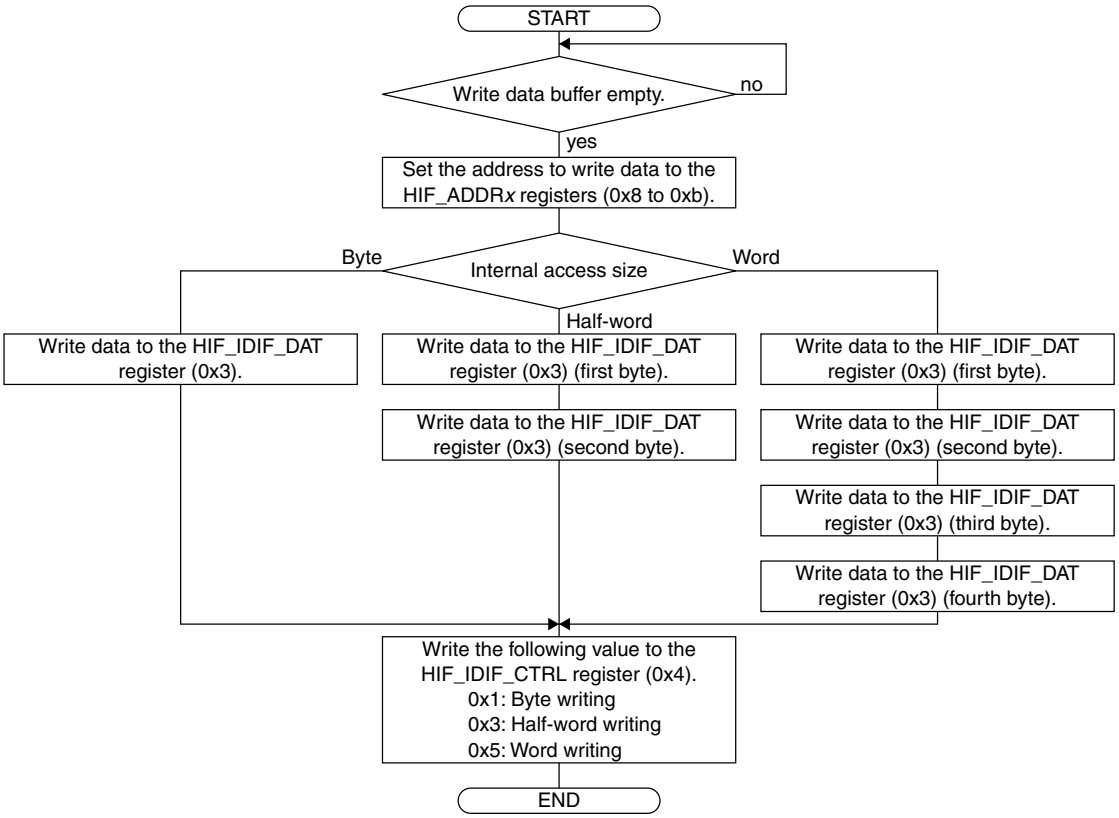


Figure 11.4.3.1 Writing Data Using the Indirect Interface (8-bit interface)

1. Read the HIF_IDIF_CTRL register (host address 0x4). Check the WR_BF value.
   The WR_BF flag indicates the status of the write data buffer. The flag bit is 1 (data full) if the prior data writing operation has not been completed and 0 (data empty) if it has been completed.
   Wait until the WR_BF flag bit becomes 0.
2. Set the address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
3. Write data to the HIF_IDIF_DAT register.
   If byte access is used, write once.
   If half-word access is used, write twice according to the endian mode setting.
   If word access is used, write four times according to the endian mode setting.
   The data is written to the write data buffer.
4. Write the byte data that has been set as follows to the HIF_IDIF_CTRL register.
   D[7:3] = 0b00000
   D[2:1] (ACCESS_MD[1:0]) = 0b00 (byte), 0b01 (half-word), or 0b10 (word)
   D0 (RW_SEL) = 1 (write)

   ACCESS_MD[1:0]/HIF_IDIF_CTRL register specifies the internal access size.
   RW_SEL/HIF_IDIF_CTRL register specifies reading or writing.

When this data is written to the HIF_IDIF_CTRL register, the HIF module writes the data in the write data buffer to the specified address.

Reading data via 8-bit interface

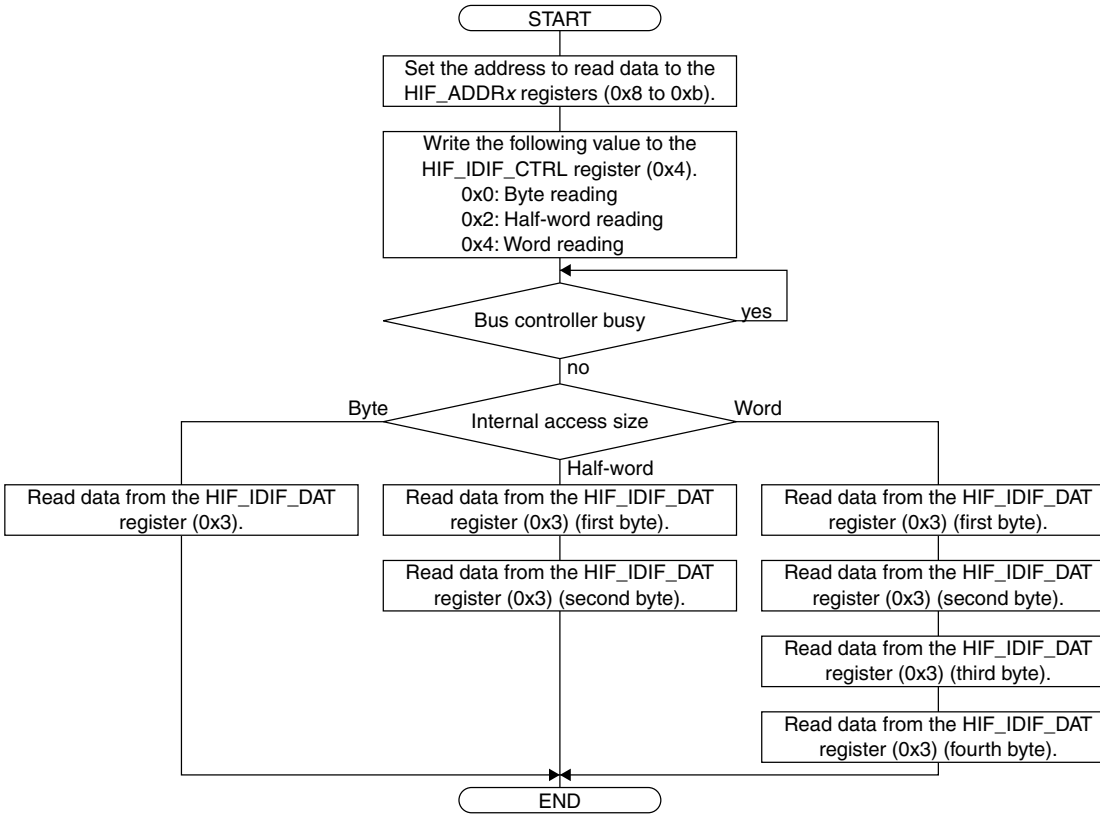


Figure 11.4.3.2 Reading Data Using the Indirect Interface (8-bit interface)

1. Set the address to be accessed to the HIF_ADDR0 to HIF_ADDR3 registers (only when required).
2. Write the byte data that has been set as follows to the HIF_IDIF_CTRL register.
   D[7:3] = 0b00000
   D[2:1] (ACCESS_MD[1:0]) = 0b00 (byte), 0b01 (half-word), or 0b10 (word)
   D0 (RW_SEL) = 0 (read)

   When this data is written to the HIF_IDIF_CTRL register, the HIF module reads the data from the specified address and stores it in the read data buffer.
3. Read the HIF_IDIF_CTRL register (host address 0x4). Check the BUSY value.
   The BUSY flag indicates the status of the bus controller. The flag bit is 1 (busy) if data is being accessed, and 0 (ready) when the bus operation has been completed.
   Wait until the BUSY flag becomes 0.
4. Read data from the HIF_IDIF_DAT register.
   If byte access is used, read once.
   If half-word access is used, read twice according to the endian mode setting.
   If word access is used, read four times according to the endian mode setting.

Writing data via 16-bit interface

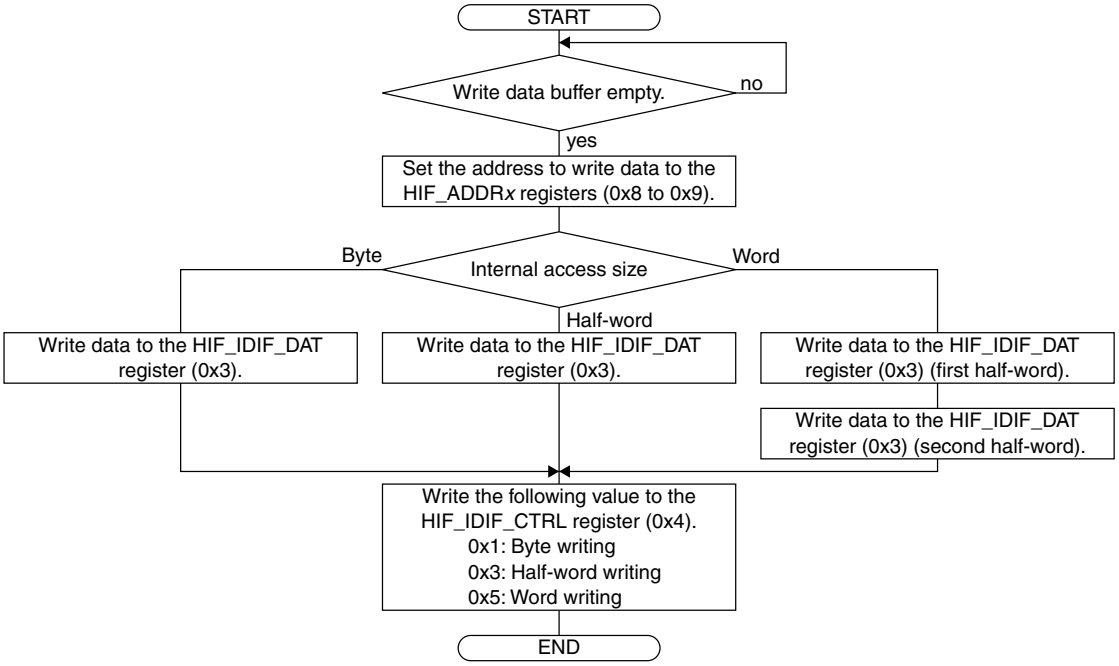


Figure 11.4.3.3 Writing Data Using the Indirect Interface (16-bit interface)

1. Read the HIF_IDIF_CTRL register (host address 0x4). Check the WR_BF value.
   The WR_BF flag indicates the status of the write data buffer. The flag bit is 1 (data full) if the prior data writing operation has not been completed and 0 (data empty) if it has been completed.
   Wait until the WR_BF flag bit becomes 0.

2. Set the address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).

3. Write data to the HIF_IDIF_DAT register.
   If byte access is used, write once. Data must be written to the low-order 8 bits of the HIF_IDIF_DAT register.
   If half-word access is used, write once.
   If word access is used, write twice according to the endian mode setting.
   The data is written to the write data buffer.

4. Write the data that has been set as follows to the HIF_IDIF_CTRL register.
   D[15:3] = 0b0000000000000
   D[2:1] (ACCESS_MD[1:0]) = 0b00 (byte), 0b01 (half-word), or 0b10 (word)
   D0 (RW_SEL) = 1 (write)

   ACCESS_MD[1:0]/HIF_IDIF_CTRL register specifies the internal access size.
   RW_SEL/HIF_IDIF_CTRL register specifies reading or writing.
   When this data is written to the HIF_IDIF_CTRL register, the HIF module writes the data in the write data buffer to the specified address.

Reading data via 16-bit interface

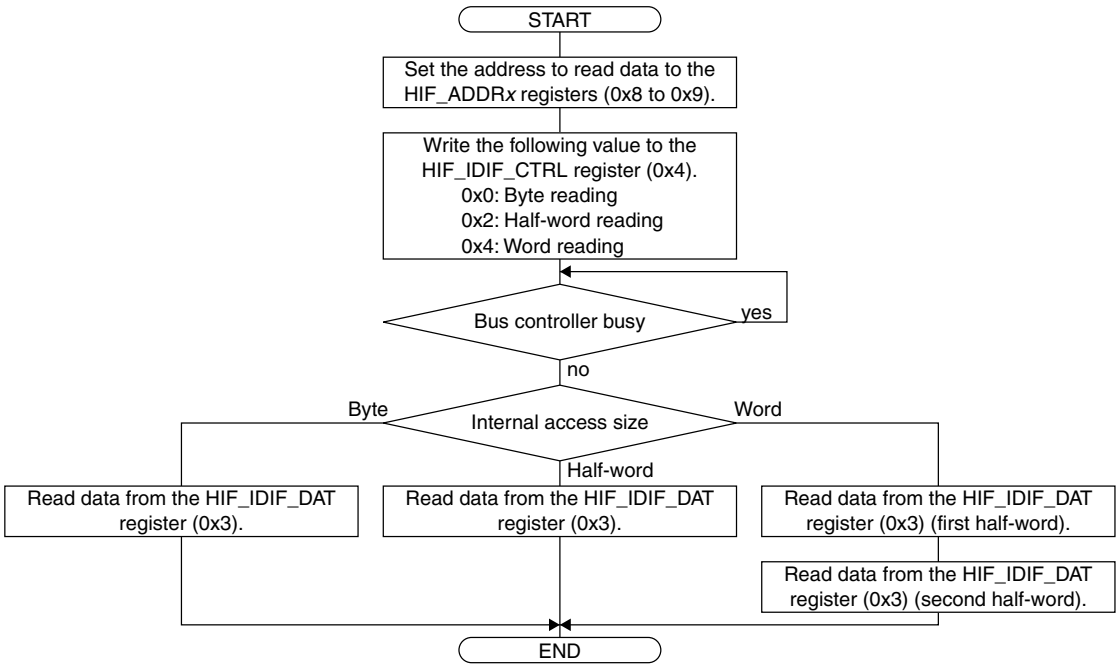


Figure 11.4.3.4 Reading Data Using the Indirect Interface (16-bit interface)

1. Set the address to be accessed to the HIF_ADDR0 and HIF_ADDR1 registers (only when required).
2. Write the data that has been set as follows to the HIF_IDIF_CTRL register.
   D[15:3] = 0b0000000000000
   D[2:1] (ACCESS_MD[1:0]) = 0b00 (byte), 0b01 (half-word), or 0b10 (word)
   D0 (RW_SEL) = 0 (read)

   When this data is written to the HIF_IDIF_CTRL register, the HIF module reads the data from the specified address and stores it in the read data buffer.
3. Read the HIF_IDIF_CTRL register (host address 0x4). Check the BUSY value.
   The BUSY flag indicates the status of the bus controller. The flag bit is 1 (busy) if data is being accessed, and 0 (ready) when the bus operation has been completed.
   Wait until the BUSY flag becomes 0.
4. Read data from the HIF_IDIF_DAT register.
   If byte access is used, read once. The low-order 8 bits of the HIF_IDIF_DAT register are effective as read data. The high-order 8 bits are read as 0.
   If half-word access is used, read once.
   If word access is used, read twice according to the endian mode setting.

11.5 Timing Chart

The HIF is controlled by PCLK (SYSCLK) supplied from the CMU.
The following diagram is a read/write timing chart.

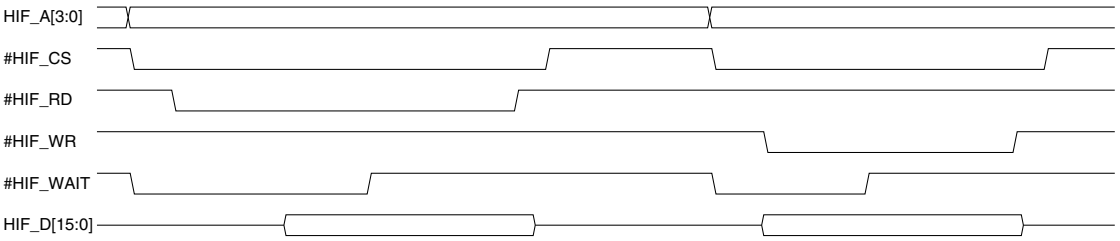


Figure 11.5.1 HIF Register Reading/Writing by the Host

11.6 Interrupts

The HIF module can output interrupt requests sent to an external host processor and to the ITC.

Generating interrupts to the external host processor

The HIF can output an interrupt request signal to the external host processor in either of the following cases:
1. When an interrupt request is generated from the ITC to the C33 PE Core
2. When the SINT flag is set by the S1C33L27 program (software interrupt)

The factor generating an interrupt can be selected using INTSEL/HIF_CTRL register.
The SINT flag interrupt request is selected if INTSEL is 0 (default), and the ITC interrupt request is selected if INTSEL is 1.

SINT

If INTSEL is 0, and 1 is written from the C33 PE Core to SINT/HIF_FLAG0 register, the #HIF_INT signal becomes active, enabling an interrupt request to be issued to an external host processor. 0 can also be written to perform a reset operation. In this case the #HIF_INT signal will become inactive.
The external host processor can read and reset the SINT flag only; settings cannot be made. Write 1 to reset the SINT flag from the external host processor. If an interrupt request was received by the external host processor, be sure to clear this flag (make the #HIF_INT signal inactive).

ITC interrupt request signal

If INTSEL is set to 1, the interrupt signal output by the ITC to the C33 PE Core is output to an external host processor as a #HIF_INT signal. Accordingly, the #HIF_INT signal becomes active when any of the interrupts authorized by the ITC is generated. The S1C33L27 internal interrupt control method is the same as that used when generating an interrupt to the C33 PE Core.

Interrupts from the external host processor

Writing 1 from the external host processor to the HINT/HIF_FLAG1 register flag bit enables an HIF interrupt request to be issued to the ITC or an NMI to be generated. A reset operation can also be performed by writing 0.
The C33 PE Core can read and reset the HINT flag only; settings cannot be made. Write 1 to reset the HINT flag bit from the C33 PE Core. If an interrupt request was received, be sure to clear this flag.
Whether to generate a maskable HIF interrupt or an NMI by setting HINT is determined by the value set for HINT_MD/HIF_EXTCTRL1 register. When HINT_MD is set to 0 (default), an NMI will be generated; when it is set to 1, an HIF interrupt request will be issued to the ITC.

For specific information on interrupt processing, see the “Interrupt Controller (ITC)” chapter.

11.7 Control Register Details

11.7.1 Host Processor Special-Purpose Registers

Table 11.7.1.1 List of Host Processor Special-Purpose Registers

Address	Register name		Function
Host 0x0	HIF_DRIF_B	HIF Direct Interface Byte Data Register	Byte data direct read/write register
Host 0x1	HIF_DRIF_HW	HIF Direct Interface Half Word Data Register	Half word data direct read/write register
Host 0x2	HIF_DRIF_W	HIF Direct Interface Word Data Register	Word data direct read/write register
Host 0x3	HIF_IDIF_DAT	HIF Indirect Interface Data Register	Data indirect read/write register
Host 0x4	HIF_IDIF_CTRL	HIF Indirect Interface Control Register	Control data indirect reading/writing
Host 0x6	HIF_CTRL	HIF Control Register	Set access conditions
Host 0x8	HIF_ADDR0	HIF Address Register 0	Address bits A[7:0] (8-bit I/F) or A[15:0] (16-bit I/F)
Host 0x9	HIF_ADDR1	HIF Address Register 1	Address bits A[15:8] (8-bit I/F) or A[31:16] (16-bit I/F)
Host 0xa	HIF_ADDR2	HIF Address Register 2	Address bits A[23:16] (8-bit I/F)
Host 0xb	HIF_ADDR3	HIF Address Register 3	Address bits A[31:24] (8-bit I/F)
Host 0xc	HIF_SP_IDX	HIF Special Register Index Register	Specify special register
Host 0xd	HIF_SP_DAT	HIF Special Register Data Register	Special register data

The host processor special-purpose registers are described individually below. All these registers cannot be accessed from the C33 PE Core.

Notes: • These registers are all configured as 8-bit registers when HIF is set to 8-bit interface, or as 16-bit registers when HIF is set to 16-bit interface.

• When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

HIF Direct Interface Byte Data Register (HIF_DRIF_B)

Notes: • This register is used only with the direct interface.

• To use the direct interface, the host processor must have an external wait request input pin, and this pin must be connected to the S1C33L27 #HIF_WAIT pin (active low).
If a host processor lacking an external wait request input is used, read/write using the HIF_IDIF_DAT register (host address 0x3) and the HIF_IDIF_CTRL register (host address 0x4) for the indirect interface.

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Byte Data Register (HIF_DRIF_B)	**Host 0x0** (8 bits)	D7–0	**B_DAT[7:0]**	Byte access data	0x0–0xff	X	R/W	

D[7:0] B_DAT[7:0]: Byte Access Data Bits

This register is used to read/write data when accessing the S1C33L27 in byte size via the direct interface. (Default: undefined)

Reading Data

When the external host processor reads this register, a bus operation to read byte data from the address set at the HIF_ADDR*x* registers (0x8 to 0xb) is executed. Once the read values have been loaded into the read data buffer, the byte data is output to HIF_D[7:0].

Writing Data

When the external host processor writes data to this register, a byte write bus operation is executed for the address set at the HIF_ADDR*x* registers (0x8 to 0xb), with the HIF module serving as the bus master.

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Byte Data Register (HIF_DRIF_B)	Host 0x0 (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	B_DAT[7:0]	Byte access data	0x0–0xff	X	R/W	

D[15:8] Reserved

D[7:0] B_DAT[7:0]: Byte Access Data Bits

This register is used to read/write data when accessing the S1C33L27 in byte size via the direct interface. (Default: undefined)

Reading Data

When the external host processor reads this register, a bus operation to read byte data from the address set at the HIF_ADDR*x* registers (0x8 and 0x9) is executed. Once the read values have been loaded into the read data buffer, the byte data is output to HIF_D[7:0]. HIF_D[15:8] outputs 0.

Writing Data

When the external host processor writes data to this register, a byte write bus operation is executed for the address set at the HIF_ADDR*x* registers (0x8 and 0x9), with the HIF module serving as the bus master.

HIF Direct Interface Half Word Data Register (HIF_DRIF_HW)

Notes: • This register is used only with the direct interface.

• To use the direct interface, the host processor must have an external wait request input pin, and this pin must be connected to the S1C33L27 #HIF_WAIT pin (active low).
If a host processor lacking an external wait request input is used, read/write using the HIF_IDIF_DAT register (host address 0x3) and the HIF_IDIF_CTRL register (host address 0x4) for the indirect interface.

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Half Word Data Register (HIF_DRIF_HW)	Host 0x1 (8 bits)	D7–0	HW_DAT [7:0]	Half word access data	0x0–0xff	X	R/W	

D[7:0] HW_DAT[7:0]: Half Word Access Data Bits

This register is used to read/write data when accessing the S1C33L27 in half-word size via the direct interface. (Default: undefined)
Access this register twice in succession to read/write half-word data.

Reading Data

When this register is read for the first time, a bus operation to read half-word data from the address set at the HIF_ADDR*x* registers (0x8 to 0xb) is executed. Once the read half-word values have been loaded into the read data buffer, the first byte data is output to HIF_D[7:0].
When the register is read the second time, the second byte data is read from the read data buffer and then output to HIF_D[7:0].

Writing Data

Write data to this register in the sequence of first byte, then second byte. The data is temporarily stored in the write data buffer and collated as half-word data.
When the second byte has been written, a half-word write bus operation is executed for the address set at the HIF_ADDR*x* registers (0x8 to 0xb), with the HIF module serving as the bus master.
The sequence of the first and second bytes will differ, depending on the endian mode specified at ENDIAN/HIF_CTRL register.

Table 11.7.1.2 Half-Word Data Byte Arrangement (for 8-bit interface)

	Little endian	Big endian
First byte	D[7:0]	D[15:8]
Second byte	D[15:8]	D[7:0]

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Half Word Data Register (HIF_DRIF_HW)	**Host 0x1** (16 bits)	D15–0	**HW_DAT [15:0]**	Half word access data	0x0–0xffff	X	R/W	

D[15:0] HW_DAT[15:0]: Half Word Access Data Bits

This register is used to read/write data when accessing the S1C33L27 in half-word size via the direct interface. (Default: undefined)

Reading Data

When the external host processor reads this register, a bus operation to read half-word data from the address set at the HIF_ADDR*x* registers (0x8 and 0x9) is executed. Once the read values have been loaded into the read data buffer, the half-word data is output to HIF_D[15:0].

Writing Data

When the external host processor writes data to this register, a half-word write bus operation is executed for the address set at the HIF_ADDR*x* registers (0x8 and 0x9), with the HIF module serving as the bus master.

HIF Direct Interface Word Data Register (HIF_DRIF_W)

Notes: • This register is used only with the direct interface.

• To use the direct interface, the host processor must have an external wait request input pin, and this pin must be connected to the S1C33L27 #HIF_WAIT pin (active low).
If a host processor lacking an external wait request input is used, read/write using the HIF_IDIF_DAT register (host address 0x3) and the HIF_IDIF_CTRL register (host address 0x4) for the indirect interface.

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Word Data Register (HIF_DRIF_W)	**Host 0x2** (8 bits)	D7–0	**W_DAT[7:0]**	Word access data	0x0–0xff	X	R/W	

D[7:0] W_DAT[7:0]: Word Access Data Bits

This register is used to read/write data when accessing the S1C33L27 in word size via the direct interface. (Default: undefined)
Access this register four times in succession to read/write word data.

Reading Data

When this register is read for the first time, a bus operation to read word data from the address set at the HIF_ADDR*x* registers (0x8 to 0xb) is executed. Once the read word values have been loaded into the read data buffer, the first byte data is output to HIF_D[7:0].
When the register is read the second, third, and fourth times, the second, third, and fourth byte data are read from the read data buffer and then output to HIF_D[7:0].

Writing Data

Write data to this register in the sequence of first byte through fourth byte. The data is temporarily stored in the write data buffer and collated as word data.
When the fourth byte has been written, a word write bus operation is executed for the address set at the HIF_ADDR*x* registers (0x8 to 0xb), with the HIF module serving as the bus master.

The sequence of the first through fourth bytes will differ, depending on the endian mode specified at the ENDIAN/HIF_CTRL register.

Table 11.7.1.3 Word Data Byte Arrangement (for 8-bit interface)

	Little endian	Big endian
First byte	D[7:0]	D[31:24]
Second byte	D[15:8]	D[23:16]
Third byte	D[23:16]	D[15:8]
Fourth byte	D[31:24]	D[7:0]

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Word Data Register (HIF_DRIF_W)	**Host 0x2** (16 bits)	D15–0	**W_DAT [15:0]**	Word access data	0x0–0xffff	X	R/W	

D[15:0] W_DAT[15:0]: Half Word Access Data Bits

This register is used to read/write data when accessing the S1C33L27 in word size via the direct interface. (Default: undefined)

Access this register twice in succession to read/write word data.

Reading Data

When this register is read for the first time, a bus operation to read word data from the address set at the HIF_ADDR*x* registers (0x8 and 0x9) is executed. Once the read word values have been loaded into the read data buffer, the first half-word data is output to HIF_D[15:0].

When the register is read the second time, the second half-word data is read from the read data buffer and then output to HIF_D[15:0].

Writing Data

Write data to this register in the sequence of first half-word, then second half-word. The data is temporarily stored in the write data buffer and collated as word data.

When the second half-word has been written, a word write bus operation is executed for the address set at the HIF_ADDR*x* registers (0x8 and 0x9), with the HIF module serving as the bus master.

The sequence of the first and second half-words will differ, depending on the endian mode specified at ENDIAN/HIF_CTRL register.

Table 11.7.1.4 Word Data Half-Word Arrangement (for 16-bit interface)

	Little endian	Big endian
First half-word	D[15:0]	D[31:16]
Second half-word	D[31:16]	D[15:0]

HIF Indirect Interface Data Register (HIF_IDIF_DAT)

Note: This register is used only with the indirect interface.

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Indirect Interface Data Register (HIF_IDIF_DAT)	**Host 0x3** (8 bits)	D7–0	**IDIF_DAT [7:0]**	Indirect interface data	0x0–0xff	X	R/W	

D[7:0] IDIF_DAT[7:0]: Indirect Interface Data Bits

This register is used to read/write data when accessing the S1C33L27 via the indirect interface. (Default: undefined)

If byte access is used, read/write this register once.

If half-word access is used, read/write this register twice.

If word access is used, read/write this register four times.

Reading Data

When a read command is written to the HIF_IDIF_CTRL register (host address 0x4), the HIF module reads data at the specified size (byte, half-word or word) from the address specified at the HIF_ADDR*x* registers (0x8 to 0xb) and stores it in the read data buffer. During this bus operation, BUSY/HIF_IDIF_CTRL register is set to 1. Once the read data is stored in the read data buffer, BUSY is reset to 0.
After confirming that the BUSY bit is 0, read this register for the number of times required by the selected access size.

Writing Data

After checking that the write data buffer is empty (WR_BF/HIF_IDIF_CTRL register = 0), write data to this register for the number of times required by the selected access size. The data is temporarily stored in the write data buffer and collated as data of the selected access size.
Next, write a write command to the HIF_IDIF_CTRL register (host address 0x4). This results in the execution of a data write bus operation to the address set at the HIF_ADDR*x* registers (0x8 to 0xb), with the HIF module serving as the bus master.

When performing half-word or word access, confirm that the byte data read/write sequences match the endian mode specified at ENDIAN/HIF_CTRL register (see Tables 11.7.1.2 and 11.7.1.3).

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Indirect Interface Data Register (HIF_IDIF_DAT)	**Host 0x3** (16 bits)	D15–0	**IDIF_DAT [15:0]**	Indirect interface data	0x0–0xffff	X	R/W	

D[15:0] IDIF_DAT[15:0]: Indirect Interface Data Bits

This register is used to read/write data when accessing the S1C33L27 via the indirect interface. (Default: undefined)
If byte access is used, read/write this register once. Note that the low-order 8 bits are effective.
If half-word access is used, read/write this register once.
If word access is used, read/write this register twice.

Reading Data

When a read command is written to the HIF_IDIF_CTRL register (host address 0x4), the HIF module reads data at the specified size (byte, half-word or word) from the address specified at the HIF_ADDR*x* registers (0x8 and 0x9) and stores it in the read data buffer. During this bus operation, BUSY/HIF_IDIF_CTRL register is set to 1. Once the read data is stored in the read data buffer, BUSY is reset to 0.
After confirming that the BUSY bit is 0, read this register for the number of times required by the selected access size.

Writing Data

After checking that the write data buffer is empty (WR_BF/HIF_IDIF_CTRL register = 0), write data to this register for the number of times required by the selected access size. The data is temporarily stored in the write data buffer and collated as data of the selected access size.
Next, write a write command to the HIF_IDIF_CTRL register (host address 0x4). This results in the execution of a data write bus operation to the address set at the HIF_ADDR*x* registers (0x8 and 0x9), with the HIF module serving as the bus master.

When performing word access, confirm that the half-word data read/write sequences match the endian mode specified at ENDIAN/HIF_CTRL register (see Table 11.7.1.4).

HIF Indirect Interface Control Register (HIF_IDIF_CTRL)

Note: This register is used only with the indirect interface.

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Indirect Interface Control Register (HIF_IDIF_CTRL)	Host 0x4 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	WR_BF	Write buffer full flag	1 Full 0 Empty	0	R	
		D3	BUSY	Bus controller busy flag	1 Busy 0 Ready	0	R	
		D2–1	ACCESS_MD[1:0]	Access mode select	ACCESS_MD[1:0] Mode 0x3–0x2 Word 0x1 Half word 0x0 Byte	0x0	R/W	
		D0	RW_SEL	Read/write cycle select	1 Write 0 Read	0	R/W	

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Indirect Interface Control Register (HIF_IDIF_CTRL)	Host 0x4 (16 bits)	D15–5	–	reserved	–	–	–	0 when being read.
		D4	WR_BF	Write buffer full flag	1 Full 0 Empty	0	R	
		D3	BUSY	Bus controller busy flag	1 Busy 0 Ready	0	R	
		D2–1	ACCESS_MD[1:0]	Access mode select	ACCESS_MD[1:0] Mode 0x3–0x2 Word 0x1 Half word 0x0 Byte	0x0	R/W	
		D0	RW_SEL	Read/write cycle select	1 Write 0 Read	0	R/W	

D[15:5] Reserved (16-bit interface)

D[7:5] Reserved (8-bit interface)

D4 WR_BF: Write Buffer Full Flag Bit

Indicates the write data buffer status.

1 (R): Full

0 (R): Empty (default)

When the first data is written to the HIF_IDIF_DAT register (host address 0x3), WR_BF is set to 1. After data is written to the specified memory address, this bit is reset to 0. When the indirect interface is used to write data from the external host processor, confirm that this bit is 0 before writing data to the HIF_IDIF_DAT register (host address 0x3).

D3 BUSY: Bus Controller Busy Flag Bit

Indicates the HIF internal bus controller status.

1 (R): Busy

0 (R): Ready (default)

This bit is set to 1 (busy) while the HIF is accessing data and it reverts to 0 (ready) when the bus operation has been completed. When the indirect interface is used to perform data reading, confirm that this bit is 0 before reading the HIF_IDIF_DAT register (host address 0x3).

D[2:1] ACCESS_MD[1:0]: Access Mode Select Bits

Indicates the S1C33L27 internal access size.

Table 11.7.1.5 Access Size Specification

ACCESS_MD[1:0]	Access size
0x3 or 0x2	Word
0x1	Half-word
0x0	Byte

(Default: 0x0)

D0 RW_SEL: Read/Write Cycle Select Bit

Selects whether to execute data reads or data writes.

1 (R/W): Write

0 (R/W): Read (default)

When a write is performed to this register, the bus operation specified at D[2:0] is executed.

Table 11.7.1.6 Operation by Writing to HIF_DIF_CTRL Register

RW_SEL	ACCESS_MD[1:0]	Bus operation
1	0x3 or 0x2	Word writing
	0x1	Half-word writing
	0x0	Byte writing
0	0x3 or 0x2	Word reading
	0x1	Half-word reading
	0x0	Byte reading

HIF Control Register (HIF_CTRL)

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Control Register (HIF_CTRL)	Host 0x6 (8 bits)	D7	HIFSEL	HIF data width select	1 16 bits 0 8 bits	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	INTSEL	Interrupt source select	1 ITC_INT 0 SINT	0	R/W	
		D3–2	ADDR_MD [1:0]	Address control mode select	ADDR_MD[1:0] Mode 0x3–0x2 reserved 0x1 Post increment 0x0 Fixed	0x0	R/W	
		D1	ENDIAN	Endian mode select	1 Big endian 0 Little endian	0	R/W	
		D0	AHB_EN	AHB interface enable	1 Enable 0 Disable	0	R/W	

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Control Register (HIF_CTRL)	Host 0x6 (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7	HIFSEL	HIF data width select	1 16 bits 0 8 bits	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	INTSEL	Interrupt source select	1 ITC_INT 0 SINT	0	R/W	
		D3–2	ADDR_MD [1:0]	Address control mode select	ADDR_MD[1:0] Mode 0x3–0x2 reserved 0x1 Post increment 0x0 Fixed	0x0	R/W	
		D1	ENDIAN	Endian mode select	1 Big endian 0 Little endian	0	R/W	
		D0	AHB_EN	AHB interface enable	1 Enable 0 Disable	0	R/W	

D[15:8] Reserved (16-bit interface)

D7 HIFSEL: HIF Data Width Select Bit

Selects the host interface data bus width.
1 (R/W): 16 bits
0 (R/W): 8 bits (default)

Writing 0 to HIFSEL configures the host interface to 8-bit data bus (HIF_D[7:0]); writing 1 to HIFSEL configures to 16-bit data bus (HIF_D[15:0]). The HIF register bit width is also set to 8- or 16 bits according to this selection (except the special registers).

D[6:5] Reserved

D4 INTSEL: Interrupt Source Select Bit

Selects the interrupt source of the S1C33L27 to generate an interrupt to the external host processor.
1 (R/W): ITC interrupt request output
0 (R/W): SINT control (default)

When INTSEL is 0, writing 1 to SINT/HIF_FLAG0 register asserts the #HIF_INT signal, enabling an interrupt request to be issued to an external host processor. When INTSEL is set to 1, the interrupt signal output by the ITC to the C33 PE Core is also output to #HIF_INT. Accordingly, an interrupt is also generated to an external host processor at the same time an interrupt is generated inside the S1C33L27.

D[3:2] ADDR_MD[1:0]: Address Control Mode Select Bits

Specifies the address control method after a memory access operation.

Table 11.7.1.7 Transfer Address Control

ADRR_MD[1:0]	Transfer address control		
	Byte access	Half-word access	Word access
0x3 or 0x2	Reserved		
0x1	Increment (+1)	Increment (+2)	Increment (+4)
0x0	Fixed address	Fixed address	Fixed address

(Default: 0x0)

When ADDR_MD[1:0] is 0x0, the address set in the HIF_ADDR0 to 3 registers (host address 0x8 to 0xb) will not be changed after a read or write operation.
When ADDR_MD[1:0] is set to 0x1, the address set in the HIF_ADDR0 to 3 registers is incremented by the amount of the access size after a read or write operation, enabling successive reads or writes of that address without rewriting the HIF_ADDR0 to 3 registers.

Note: The post-increment function is enabled only for the low-order 16 bits of the address (A[15:0]). Accordingly, successive accesses that exceed the 64-KB boundary address (0x*XXXX*ffff) are not possible.

D1 ENDIAN: Endian Mode Select Bit

This bit selects the endian mode when half-word or word access is performed.
1 (R/W): Big endian
0 (R/W): Little endian (Default)

Depending on the internal access size, the external host processor performs reads/writes once (byte), twice (half-word), or four times (word). The following table shows the access data bits for each instance for the endian mode selected.

Table 11.7.1.8 Relationship Between Address and Data Bits According to Endian Mode Selected

Data bus width	Internal access size	Internal address low-order bit		Internal read/write bit		Access number of times/Number
		A1	A0	Little endian	Big endian	
8 bits	Byte	*	*	D[7:0]	D[7:0]	One time
	Half-word	*	0	D[7:0]	D[15:8]	First time
		*	1	D[15:8]	D[7:0]	Second time
	Word	0	0	D[7:0]	D[31:24]	First time
		0	1	D[15:8]	D[23:16]	Second time
		1	0	D[23:16]	D[15:8]	Third time
		1	1	D[31:24]	D[7:0]	Fourth time
16 bits	Byte	*	*	D[7:0]	D[7:0]	One time
	Half-word	*	0	D[15:0]	D[15:0]	One time
	Word	0	0	D[15:0]	D[31:16]	First time
		1	0	D[31:16]	D[15:0]	Second time

D0 AHB_EN: AHB Interface Enable Bit

This bit permits access by the external host processor of the AHB slave device.
1 (R/W): Permitted
0 (R/W): Disabled (Default)

Setting AHB_EN to 1 enables the external host processor to access the internal memory in Area 3, internal peripheral circuits/memory in Area 6, and the external memory connected to the S1C33L27.

HIF Address Register *x* (HIF_ADDR*x*)

In 8-bit interface mode (HIF_ADDR0–HIF_ADDR3)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Address Register 0 (HIF_ADDR0)	**Host 0x8** (8 bits)	D7–0	**ADDR [7:0]**	S1C33L27 internal address A[7:0]	0x0–0xff	0x0	R/W	
HIF Address Register 1 (HIF_ADDR1)	**Host 0x9** (8 bits)	D7–0	**ADDR [15:8]**	S1C33L27 internal address A[15:8]	0x0–0xff	0x0	R/W	
HIF Address Register 2 (HIF_ADDR2)	**Host 0xa** (8 bits)	D7–0	**ADDR [23:16]**	S1C33L27 internal address A[23:16]	0x0–0xff	0x0	R/W	
HIF Address Register 3 (HIF_ADDR3)	**Host 0xb** (8 bits)	D7–0	**ADDR [31:24]**	S1C33L27 internal address A[31:24]	0x0–0xff	0x0	R/W	

In 16-bit interface mode (HIF_ADDR0–HIF_ADDR1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Address Register 0 (HIF_ADDR0)	**Host 0x8** (16 bits)	D15–0	**ADDR [15:0]**	Address A[15:0]	0x0–0xffff	0x0	R/W	
HIF Address Register 1 (HIF_ADDR1)	**Host 0x9** (16 bits)	D15–0	**ADDR [31:16]**	Address A[31:16]	0x0–0xffff	0x0	R/W	

ADDR[31:0]: S1C33L27 Internal Address Bits

Sets the S1C33L27 memory address accessed by the external host processor. (Default: 0x0)
Both the direct interface mode and the indirect interface mode use these registers to specify the address.

Notes: • When accessing the S1C33L27 in half-word size, set a half-word boundary address (A0 = 0). When accessing in word size, set a word boundary address (A[1:0] = 0b00).

• If an access attempt is made for an address of a memory not actually installed, an invalid write or invalid read will occur.

HIF Special Register Index Register (HIF_SP_IDX)

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Special Register Index Register (HIF_SP_IDX)	**Host 0xc** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4–0	**INDEX[4:0]**	Special register index	0x1–0x13	0x0	R/W	

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Special Register Index Register (HIF_SP_IDX)	**Host 0xc** (16 bits)	D15–5	**–**	reserved	–	–	–	0 when being read.
		D4–0	**INDEX[4:0]**	Special register index	0x1–0x13	0x0	R/W	

D[15:5] Reserved (16-bit interface)

D[7:5] Reserved (8-bit interface)

D[4:0] INDEX[4:0]: Special Register Index Bits

Specifies the special register accessed through the HIF_SP_DAT register (host address 0xd) using an index number (0x1, 0x8 to 0xf, 0x10 to 0x13). (Default: 0x0)
Special registers can be accessed by reading/writing from/to the HIF_SP_DAT register (host address 0xd) after writing the index to this register.
The special registers are shared registers that can also be accessed by the C33 PE Core. For more information on these registers, see Section 11.7.2, “Shared Registers.”

Table 11.7.1.9 List of Special Registers

Index	Special register	Function
0x1	HIF Extended Control (HIF_EXTCTRL1) Register 1	Set interrupt mode
0x8	HIF Semaphore (HIF_SMPH0) Register 0	Semaphore 0
0x9	HIF Semaphore (HIF_SMPH1) Register 1	Semaphore 1
0xa	HIF Semaphore (HIF_SMPH2) Register 2	Semaphore 2
0xb	HIF Semaphore (HIF_SMPH3) Register 3	Semaphore 3
0xc	HIF Semaphore (HIF_SMPH4) Register 4	Semaphore 4
0xd	HIF Semaphore (HIF_SMPH5) Register 5	Semaphore 5
0xe	HIF Semaphore (HIF_SMPH6) Register 6	Semaphore 6
0xf	HIF Semaphore (HIF_SMPH7) Register 7	Semaphore 7
0x10	HIF Scratchpad (HIF_SCRATCH0) Register 0	Scratchpad register 0
0x11	HIF Scratchpad (HIF_SCRATCH1) Register 1	Scratchpad register 1
0x12	HIF Scratchpad (HIF_SCRATCH2) Register 2	Scratchpad register 2
0x13	HIF Scratchpad (HIF_SCRATCH3) Register 3	Scratchpad register 3

HIF Special Register Data Register (HIF_SP_DAT)

In 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Special Register Data Register (HIF_SP_DAT)	**Host 0xd** (8 bits)	D7–0	**SP_DAT[7:0]**	Special register data	0x0–0xff	0x0	R/W	

In 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Special Register Data Register (HIF_SP_DAT)	**Host 0xd** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	**SP_DAT[7:0]**	Special register data	0x0–0xff	0x0	R/W	

D[15:8] Reserved (16-bit interface)

D[7:0] SP_DAT[7:0]: Special Register Data Bits

The external host processor reads/writes the special register specified with the HIF_SP_IDX register (host address 0xc) through this register. (Default: 0x0)

11.7.2 Shared Registers

Table 11.7.2.1 List of Shared Registers

Address	Register name		Function
Host 0xd[0x1] /0x300901	HIF_EXTCTRL1	HIF Extended Control Register 1	Set interrupt mode
Host 0xd[0x8] /0x300908	HIF_SMPH0	HIF Semaphore Register 0	Semaphore 0
Host 0xd[0x9] /0x300909	HIF_SMPH1	HIF Semaphore Register 1	Semaphore 1
Host 0xd[0xa] /0x30090a	HIF_SMPH2	HIF Semaphore Register 2	Semaphore 2
Host 0xd[0xb] /0x30090b	HIF_SMPH3	HIF Semaphore Register 3	Semaphore 3
Host 0xd[0xc] /0x30090c	HIF_SMPH4	HIF Semaphore Register 4	Semaphore 4
Host 0xd[0xd] /0x30090d	HIF_SMPH5	HIF Semaphore Register 5	Semaphore 5
Host 0xd[0xe] /0x30090e	HIF_SMPH6	HIF Semaphore Register 6	Semaphore 6
Host 0xd[0xf] /0x30090f	HIF_SMPH7	HIF Semaphore Register 7	Semaphore 7
Host 0xd[0x10] /0x300910	HIF_SCRATCH0	HIF Scratchpad Register 0	Scratchpad register 0
Host 0xd[0x11] /0x300911	HIF_SCRATCH1	HIF Scratchpad Register 1	Scratchpad register 1
Host 0xd[0x12] /0x300912	HIF_SCRATCH2	HIF Scratchpad Register 2	Scratchpad register 2

Address	Register name		Function
Host 0xd[0x13] /0x300913	HIF_SCRATCH3	HIF Scratchpad Register 3	Scratchpad register 3
Host 0xe /0x300918	HIF_FLAG0	HIF Flag Register 0	Slave flag
Host 0xf /0x300919	HIF_FLAG1	HIF Flag Register 1	Host flag

Each shared register that can be accessed by both the C33 PE Core and the external host processor is described below. The C33 PE Core and the external host processor access addresses differ. These addresses are described below.

Example: Host 0xe External host processor address
Host 0xd[0x1] Special register specified with the index 0x1 set at HIF_SP_IDX (host address 0xc) and accessed through the HIF_SP_DAT register (host address 0xd)
0x300901 S1C33L27 address

Notes: • When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

• Do not read or write the HIF registers on the S1C33L27 side when HIF_CLK is disabled by HIF_CKE/CMU_CLKCTL2 register is set to 0.

HIF Extended Control Register 1 (HIF_EXTCTRL1)

On the host side (8-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Extended Control Register 1 (HIF_EXTCTRL1)	**Host 0xd [0x1]** (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	**HINT_MD**	Host interrupt mode select	1 INT 0 NMI	0	R/W	
		D0	–	reserved	–	–	–	0 when being read.

On the host side (16-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Extended Control Register 1 (HIF_EXTCTRL1)	**Host 0xd [0x1]** (16 bits)	D15–2	–	reserved	–	–	–	0 when being read.
		D1	**HINT_MD**	Host interrupt mode select	1 INT 0 NMI	0	R/W	
		D0	–	reserved	–	–	–	0 when being read.

On the S1C33L27 side

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Extended Control Register 1 (HIF_EXTCTRL1)	**0x300901** (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	**HINT_MD**	Host interrupt mode select	1 INT 0 NMI	0	R/W	
		D0	–	reserved	–	–	–	0 when being read.

D[15:8] Reserved (External host processor, 16-bit interface mode)

D[7:2] Reserved

D1 HINT_MD: Host Interrupt Mode Select Bit

Selects the type of interrupt generated when the external host processor writes 1 to the HINT/HIF_FLAG1 register.
1 (R/W): Maskable interrupt
0 (R/W): NMI (default)

When HINT_MD is 0, setting HINT to 1 generates an NMI. When HINT_MD is set to 1, setting HINT to 1 issues an HIF interrupt request to the ITC.

D0 Reserved

HIF Semaphore Registers 0–7 (HIF_SMPH0–7)

On the host side (8-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Semaphore Register *x* (HIF_SMPH*x*)	**Host 0xd [0x8] | [0xf]** (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	**SMPH*x***	Semaphore *x*	1 Busy 0 Not busy	0	R/W	

On the host side (16-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Semaphore Register *x* (HIF_SMPH*x*)	**Host 0xd [0x8] | [0xf]** (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	**SMPH*x***	Semaphore *x*	1 Busy 0 Not busy	0	R/W	

On the S1C33L27 side

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Semaphore Register *x* (HIF_SMPH*x*)	**0x300908 | 0x30090f** (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	**SMPH*x***	Semaphore *x*	1 Busy 0 Not busy	0	R/W	

Note: The "*x*" in the register and bit names represents a semaphore number (0 to 7).

D[15:8] Reserved (External host processor, 16-bit interface mode)

D[7:1] Reserved

D0 SMPH*x*: Semaphore *x* Bit

Switches the semaphore.
1 (W): Resets the semaphore
0 (W): Ignored
1 (R): Resource in use
0 (R): Resource can be used (set to 1 after reading) (default)

The module (external host processor or S1C33L27 program) that will use the resource reads the corresponding SMPH*x*. If SMPH*x* = 0 (can be used), the resource is used and processing continues. SMPH*x* automatically changes to 1 (in use) when read. Accordingly, if SMPH*x* = 1, another module is using the resource. In this case, resource use is placed on hold until SMPH*x* reverts to 0. When a module has finished using a resource, 1 must be written to SMPH*x* to reset it to 0.

HIF Scratchpad Registers 0–3 (HIF_SCRATCH0–3)

On the host side (8-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Scratchpad Register *x* (HIF_SCRATCH*x*)	**Host 0xd [0x10] | [0x13]** (8 bits)	D7–0	**SCR*x*[7:0]**	Scratch data	0x0–0xff	0x0	R/W	

On the host side (16-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Scratchpad Register *x* (HIF_SCRATCH*x*)	**Host 0xd [0x10] | [0x13]** (16 bits)	D15-8	–	reserved	–	–	–	0 when being read.
		D7–0	**SCR*x*[7:0]**	Scratch data	0x0–0xff	0x0	R/W	

On the S1C33L27 side

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Scratchpad Register *x* (HIF_SCRATCH*x*)	**0x300910** | **0x300913** (8 bits)	D7–0	**SCR*x*[7:0]**	Scratch data	0x0–0xff	0x0	R/W	

Note: The "*x*" in the register and bit names represents the scratchpad number (0 to 3).

D[15:8] Reserved (External host processor, 16-bit interface mode)

D[7:0] SCR*x*[7:0]: Scratch Data Bits

This register is used as a general-purpose register. (Default: 0x0)
Any data can be stored to this register temporarily.
The data written to this register has no effect on HIF operations.

HIF Flag Register 0 (HIF_FLAG0)

On the host side (8-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Flag Register 0 (HIF_FLAG0)	**Host 0xe** (8 bits)	D7	**PROTECT**	HIF_FLAG write protect	1 Set/remove 0 Disable	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	**SFLAG3**	Slave flag 3	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D3	**SFLAG2**	Slave flag 2	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D2	**SFLAG1**	Slave flag 1	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D1	**SFLAG0**	Slave flag 0	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D0	**SINT**	Slave interrupt flag	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	

On the host side (16-bit interface mode)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Flag Register 0 (HIF_FLAG0)	**Host 0xe** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7	**PROTECT**	HIF_FLAG write protect	1 Set/remove 0 Disable	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	**SFLAG3**	Slave flag 3	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D3	**SFLAG2**	Slave flag 2	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D2	**SFLAG1**	Slave flag 1	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D1	**SFLAG0**	Slave flag 0	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D0	**SINT**	Slave interrupt flag	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	

On the S1C33L27 side

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Flag Register 0 (HIF_FLAG0)	**0x300918** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	**SFLAG3**	Slave flag 3	1 Set 0 Clear	0	R/W	
		D3	**SFLAG2**	Slave flag 2	1 Set 0 Clear	0	R/W	
		D2	**SFLAG1**	Slave flag 1	1 Set 0 Clear	0	R/W	
		D1	**SFLAG0**	Slave flag 0	1 Set 0 Clear	0	R/W	
		D0	**SINT**	Slave interrupt flag	1 Set 0 Clear	0	R/W	

D[15:8] Reserved (External host processor, 16-bit interface mode)

D7 PROTECT: HIF_FLAG Write Protect Bits (for the external host processor only)

Changes the write-protect setting of the HIF_FLAG0/1 registers. This register has a special configuration, and the values for the entire HIF_FLAG0 register must be written (writing only to the D7 bit will have no effect).

0x96 (W): Disable write protection
0xff (W): Write-protect the registers

Notes: • PROTECT cannot be set from the C33 PE Core.

• To reset the flags in the HIF_FLAG0 register, set PROTECT to 0 when writing to the HIF_FLAG0 register. The flags will not be reset if PROTECT is 1.

D[6:5] Reserved (External host processor)

D[7:5] Reserved (S1C33L27)

D[4:1] SFLAG[3:0]: Slave Flag Bits 3–0

These general-purpose flags can be set by the C33 PE Core.

	C33 PE Core	External host processor
1 (W):	Set flag (1)	Reset flag (0)
0 (W):	Reset flag (0)	Ignored
1 (R):	Flag = 1	Flag = 1
0 (R):	Flag = 0	Flag = 0

The C33 PE Core can set or reset the flag by writing 1 or 0.
The external host processor can read the flags and can reset by writing 1 (cannot set the flags).

D0 SINT: Slave Interrupt Flag Bit

This flag activates the #HIF_INT output and sends an interrupt request to an external host processor.

	C33 PE core	External host processor
1 (W):	Set flag (#HIF_INT → Low)	Reset flag (0)
0 (W):	Reset flag (#HIF_INT → High)	Ignored
1 (R):	Flag = 1 (#HIF_INT = Low)	Flag = 1
0 (R):	Flag = 0 (#HIF_INT = High)	Flag = 0

Interrupts by this flag are only enabled when INTSEL/HIF_CTRL register = 0. If INTSEL is 1, the interrupt request signal from the ITC to the C33 PE Core is sent to an external host processor.
This flag cannot be set from the external host processor.
When an interrupt has been received by the external host processor, this flag must be reset (make #HIF_INT inactive).

HIF Flag Register 1 (HIF_FLAG1)

On the host side (8-bit interface mode)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
HIF Flag Register 1 (HIF_FLAG1)	**Host 0xf** (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	**HFLAG2**	Host flag 2	1	Set	0	Clear	0	R/W	
		D2	**HFLAG1**	Host flag 1	1	Set	0	Clear	0	R/W	
		D1	**HFLAG0**	Host flag 0	1	Set	0	Clear	0	R/W	
		D0	**HINT**	Host interrupt flag	1	Set	0	Clear	0	R/W	

On the host side (16-bit interface mode)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
HIF Flag Register 1 (HIF_FLAG1)	**Host 0xf** (16 bits)	D15–4	–	reserved	–				–	–	0 when being read.
		D3	**HFLAG2**	Host flag 2	1	Set	0	Clear	0	R/W	
		D2	**HFLAG1**	Host flag 1	1	Set	0	Clear	0	R/W	
		D1	**HFLAG0**	Host flag 0	1	Set	0	Clear	0	R/W	
		D0	**HINT**	Host interrupt flag	1	Set	0	Clear	0	R/W	

On the S1C33L27 side

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
HIF Flag Register 1 (HIF_FLAG1)	**0x300919** (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	**HFLAG2**	Host flag 2	1	Set	0	Cleared	0	R	
					1	Reset	0	Ignored	–	W	
		D2	**HFLAG1**	Host flag 1	1	Set	0	Cleared	0	R	
					1	Reset	0	Ignored	–	W	
		D1	**HFLAG0**	Host flag 0	1	Set	0	Cleared	0	R	
					1	Reset	0	Ignored	–	W	
		D0	**HINT**	Host interrupt flag	1	Set	0	Cleared	0	R	
					1	Reset	0	Ignored	–	W	

Note: To prevent errant operations resulting from unnecessary overwrites, the default setting for the HIF_FLAG1 register is the write-protected state. To overwrite this register, the write protection must be cleared by writing 0x96 to the HIF_FLAG0 register (host address 0xe). (This can only be done from the external host processor.)

D[15:4] Reserved (External host processor, 16-bit interface mode)

D[7:4] Reserved

D[3:1] HFLAG[2:0]: Host Flag Bits 2–0

These general-purpose flags can be set by the external host processor.

	External host processor	C33 PE core
1 (W):	Set flag (1)	Reset flag (0)
0 (W):	Reset flag (0)	Ignored
1 (R):	Flag = 1	Flag = 1
0 (R):	Flag = 0	Flag = 0

The external host processor can set or reset the flag by writing 1 or 0.
The C33 PE Core can read the flags and can reset by writing 1 (cannot set the flags).

D0 HINT: Host Interrupt Flag Bit

The setting at the external host processor determines whether an NMI is generated or an interrupt request is issued to the ITC.

	External host processor	C33 PE core
1 (W):	Set flag (Generates an NMI/interrupt)	Reset flag (0)
0 (W):	Reset flag (Clears the NMI/interrupt)	Disabled
1 (R):	Flag = 1	Flag = 1
0 (R):	Flag = 0	Flag = 0

This flag cannot be set from the C33 PE core.
When an interrupt has been received by the C33 PE Core, this flag must be reset.
Whether to generate a maskable HIF interrupt or an NMI by the HINT operation is determined by the value set to HINT_MD/HIF_EXTCTRL1 register. When HINT_MD is set to 0 (default), an NMI is generated; when it is set to 1, an HIF interrupt request is issued to the ITC.

12 Cache Controller (CCU)

12.1 CCU Module Overview

In order to enable fast access to instructions and data, the S1C33L27 incorporates a cache controller (CCU) that runs by the 4-Way set associative method. Addresses 0x0 to 0x3ff (1K bytes) and 0x4c00 to 0x4fff (1K bytes) in A0RAM are used as cache memories for instructions and data, respectively, enabling fast access to external ROM/SRAM/SDRAM in the specified area (excluding access to the embedded memory and internal peripheral modules from data to be cached).
The cache can be locked before executing an interrupt handler routine of the specified interrupt level, this makes it possible to avoid refilling the cache by a lower priority interrupt when a routine that requires high-speed processing has been cached. Also, the cache is automatically locked in debugging mode, enabling debugging in a hardware break in the same timing as the normal operation.
The main features of the CCU are outlined below.

- Cache adopting the 4-Way set associative method with separate memories for instructions (1K bytes) and data (1K bytes).
- One area can be selected separately for each of the categories, instructions and data, (from Areas 14 to 22) as the area for caching.
- One-word write buffer is built in to support write through mode.
- Refill is performed using the LRU algorithm.
- A four-word burst reading function is provided to reduce waiting time for refill.
- A locking function works in debugging and interrupts (with specification of the interrupt level).
- An automatic flush function is provided for the instruction cache to work in response to software PC break in debugging.

Table 12.1.1 Cache Speed

Status	Number of cycles
Reading from the instruction or data cache (upon hitting)	2 cycles *1
Concurrent reading from the instruction and data caches (upon hitting)	
Reading from A0RAM	
Writing to the data cache (upon hitting and emptying the write buffer)	2 cycles *2
Writing to the data cache (upon mishitting and emptying the write buffer)	
Writing to A0RAM	1 cycle
Reading from the instruction or data cache (upon mishitting)	Depends on the access setting for the external memory
Writing to the data cache (when the write buffer is full)	

*1: As the instruction and data caches can be accessed at the same time, performance is maintained even when data are accessed while instructions are being fetched.
*2: Data are written to the data cache and the write buffer at the same time in the first cycle and written to an external memory in the next cycle.

Notes: • The CCU does not have a snooping function (for maintaining the data in the cache memory to match those in the external memory). The cache and the external memory are maintained in synch if reading/writing is only executed in the C33 PE Core. Use software to secure data integrity in cases where data are shared with the DMAC.

• When the CPU executes the halt or slp instruction, the clocks supplied to the C33 PE Core and cache both stop. To avoid unexpected bus operations, lock or disable the cache and make sure that the cache has been locked or disabled using the CCU_STAT register.

12.2 Cache Configuration

The CCU uses addresses 0x0 to 0x3ff (1K bytes) and 0x4c00 to 0x4fff (1K bytes) in A0RAM for the instruction cache and the data cache, respectively. The instruction cache and the data cache can be separately enabled in software. When using the CCU, take care not to allow data to be accidentally written to the cache from an application. The cache memory spaces only store data sections; the TAG sections are stored in the memory within the CCU module.

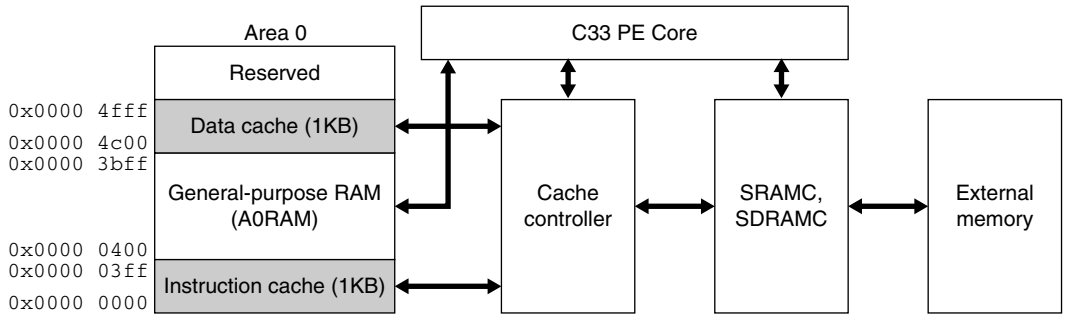


Figure 12.2.1 Cache Memory

The CCU adopts the 4-Way set associative method.
One frame is composed of cache data containing four lines (4 × 4 words), and one Way consists of four frames. Four frames located at the corresponding area in each Way are managed under one LRU entry.
Data to a cache from the external memory are loaded in units of a line (four words).
Figure 12.2.2 shows the cache configuration.

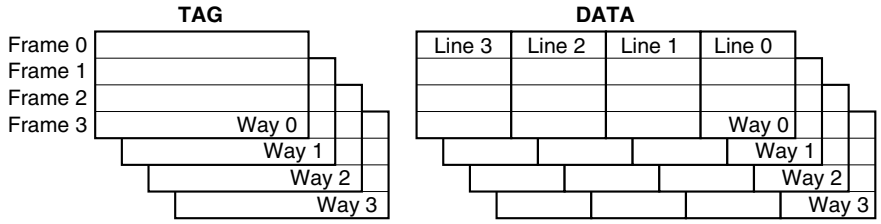


Figure 12.2.2 Cache Configuration

TAG and Data Sections

Each frame is divided into the TAG and Data sections as shown in Figure 12.2.3. The TAG section stores 18-bit addresses for comparison. The Data section consists of four words (16 bytes each) by four lines.

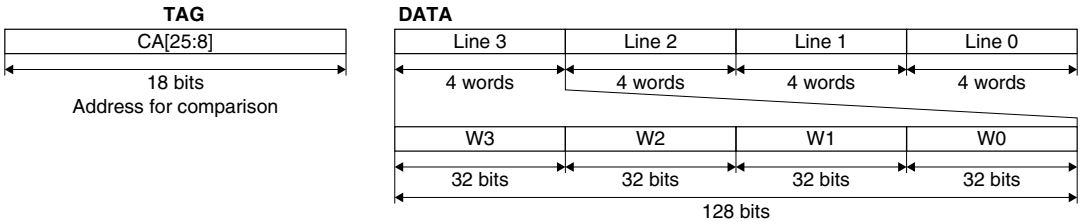


Figure 12.2.3 Frame Configuration

LRU Section

Configured with 4-Ways, the CCU has four frames of data assigned the same entry number. If nothing is hit, it is needed to select one of the four Ways to replace with, in which case, the LRU section stores the Way number.

12.3 Cache Settings and Operations

12.3.1 Cache Enable

At initial reset, the caching function is disabled.
Set up as follows to use caching.

When the instruction cache is used
Set the IC/CCU_CFG register to 1.

When the data cache is used
Set the DC/CCU_CFG register to 1.

Turning IC and DC back to 0 flushes and clears all data cached.

Note: Be sure to disable the instruction and data caches before executing the halt or slp instruction.

12.3.2 Selecting Area to Be Cached

The CCU caches access to one area (an external memory) out of Areas 14 to 22. Select a target area for caching in the instruction cache in ARIC[2:0]/CCU_AREA register and one for caching in the data cache in ARDC[2:0]/CCU_AREA register, respectively.

Table 12.3.2.1 Selecting Area to Be Cached

ARIC[2:0]/ARDC[2:0]	Areas to be cached
0x7	Area 22 (0x80000000 to 0xffffffff)
0x6	Area 21 (0x40000000 to 0x7fffffff)
0x5	Area 20 (0x20000000 to 0x3fffffff)
0x4	Area 19 (0x10000000 to 0x1fffffff)
0x3	Area 18 (0x0c000000 to 0x0fffffff)
0x2	Area 17 (0x08000000 to 0x0bffffff)
0x1	Areas 15 and 16 (0x04000000 to 0x07ffffff)
0x0	Area 14 (0x03000000 to 0x03ffffff)

(Default: 0x0)

The CCU only caches access to the range of 64MB starting at the top of the selected area. The areas after the leading 64MB space in Areas 19 to 22 are mirror areas.

12.3.3 Comparing Addresses and Cache Hit/Mishit

Among addresses output by the SRAMC, two bits at A[7:6] represent the entry number (frame offset). A[25:8] is deemed as a comparison address and compared with the address for comparison (CA[25:8]) stored in the TAG section containing four Ways under the entry selected in A[7:6].

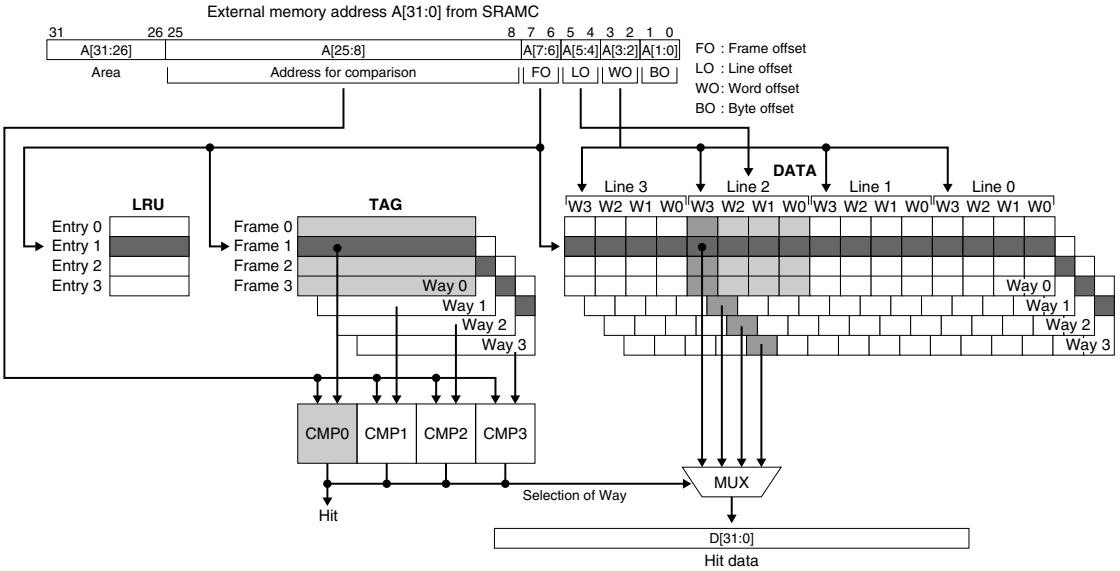


Figure 12.3.3.1 Cache-wise Operation

The following describes operation up to cache hit/mishit judgement.

1. Generates entry numbers (0–3) from address A[7:6] output by the SRAMC.
2. Reads information for four Ways from the TAG section in the selected entry. Reads word data for four Ways at the same time from the Data section that are indicated by the line offset (A[5:4]) and the word offset (A[3:2]).
3. Compares CA[25:8] in the TAG section in each Way against A[25:8].
4. It is judged as Hit if a matching Way is found in Step 3 and at the same time data in the relevant line are valid (cached from the external memory). At this stage, which Way is hit is determined.
   If no matching Way is found in Step 3, it is judged as mishit.

For example, if Way 0 is hit at A[7:6] = 0b01, A[5:4] = 0b10 and A[3:2] = 0b11, reading/writing is performed from/to Way 0 - Frame 1 - Line 2 - W3 in the cache memory.

12.3.4 Reading Operation

The following describes operations for cases where any data are hit or not hit in reading.

When any data is hit

The 32-bit data hit are transferred to the CPU and at the same time the LRU information of the relevant entry is updated.

When any data is mishit

No instructions or data in the cache are transferred to the CPU.

If no data is hit, the CCU updates (refills) the replace Way (the Way accessed earliest) obtained from the current LRU information. Refill operation takes place in units of one line (containing four words each). The CCU reads four words including the subject instruction or data from the external memory and writes the words to the relevant frame/line in the cache memory. At the same time, it transfers the subject instruction or data to the CPU and updates LRU information. The line cached is enabled in the CCU, after which instructions/data in the same line are read from the cache memory. Other lines within the same frame are disabled until it is refilled through access to the relevant addresses.

12.3.5 Writing Operation

Writing only supports the write through mode.
The write through mode enables the relevant external memory to be written at the same time when the cache memory is written in response to a request for writing to the data address in the cache. Thus, data matching is always ensured (if not changed by the DMAC). Also, in order to prevent the processing speed from dropping due to write through operation, the CCU incorporates a one-word write buffer. To enable the write buffer, set WBEN/CCU_CFG register to 1 (default). When WBEN is set to 0, the write buffer is disabled. In this case the CPU is placed into wait status until the data writing to the external memory has completed.

When any data is hit

A write cycle is issued to both the cache and the external memory. Additionally, LRU information is updated.

When any data is mishit

When any data is mishit, the cache memory is not written, and only the external memory is written. If this is the case, LRU information is not updated since nothing is written to the cache. No refilling is performed, either.

When the write buffer is enabled, writing data to the external memory is performed in two steps, first the data is written to the write buffer and then the external memory is updated. WBEMPTY/CCU_WB_STAT register and WEFINISH/CCU_WB_STAT register are provided to check the write buffer status and whether the buffered data has been actually written to the external memory or not.
WBEMPTY is set to 0 by writing data to the write buffer and is set to 1 when the buffered data is read out for writing to the external memory. At the same time the WBEMPTY is set to 1, WEFINISH goes 0 to indicate that the data is being written to the external memory. WEFINISH is set to 1 upon completion of writing to the external memory.
The write buffer improves writing speed as the CPU does not need to wait for completion of writing to the external memory. However, read the above flags to check if the data has been written to, especially when data is written to a low-speed external device.

12.3.6 Flush

Flushing refers to nullifying all data in the cache.
To flush the instruction cache, set IC/CCU_CFG register to 0 (to disable the instruction cache). To flush the data cache, set DC/CCU_CFG register to 0 (to disable the data cache).
Set these bits back to 1 to enable the caches again.
Note that the cache is flushed several cycles after writing 0 to IC or DC. Before resuming caching operation, check the status bit to ensure that flushing is completed. Check ICS/CCU_STAT register for the operating status of the instruction cache, and DCS/CCU_STAT register for that of the data cache. If these bits are 1, then the cache is in operation and the flushing is not completed. When the flushing has completed and cache stops the operation, then these bits turns to 0.

12.4 Cache Lock with Interrupt Level Specified

The cache memory size is not large, therefore, frequently generated refilling may cause system performance degradation depending on the program. To avoid this, the CCU has a feature to lock the cache against refilling with interrupt handler routines of the specified levels. LKPRI[7:0]/CCU_LK register is provided for this setting.

By setting an LKPRI[7:0] bit to 1, the priority level of interrupts to disabled refilling can be selected. Each LKPRI[7:0] bit corresponds to an interrupt level, for example, LKPRI0 corresponds to interrupt level 0 (IL[2:0] = 0) and LKPRI7 corresponds to interrupt level 7 (IL[2:0] = 7). If the interrupt level in IL[2:0] (set by the interrupt occurred) and an LKPRI[7:0] bit that has been set to 1 are matched, the cache will be locked after a lapse of several cycles. After this point, the CCU will not refill the cache until the IL[2:0] value is altered to the interrupt level of an LKPRI[7:0] bit that has been set to 0.

Example 1: When LKPRI1 is set to 1 and other bits are set to 0 (LKPRI[7:0] = 0x02, the current IL[2:0] value is 0)

The CCU can refill until a level 1 interrupt occurs. When IL[2:0] is set to 1 due to occurrence of a level 1 interrupt, the cache is locked. Therefore, the handler routine for the interrupt occurred is executed at the store location in the memory (when not hit to the cached data). Executing the reti instruction at the end of the interrupt handler routine reset IL[2:0] to 0 and releases the cache lock status.

If a higher priority interrupt occurs while an interrupt handler routine is being executed, the cache lock is released as IL[2:0] is altered. In this case, the cache will be locked again after the handler routine for the higher priority interrupt ends by the reti instruction as IL[2:0] returns to level 1.

Example 2: When all the LKPRI[7:0] bits to 1 (0xff)

The cache is locked immediately after this setting, then no refilling will occur. When no cached data is hit, the program is executed at the store location in the memory. If LKPRI[7:0] is altered and the bit corresponding to the interrupt level set in IL[2:0] is reset to 0, the cache lock is released.

Read ICLKS/CCU_STAT register and DCLKS/CCU_STAT register to check whether the instruction and data caches are locked or not. If the status bit is 1, the cache is locked. If the bit is 0, the cache is not locked.
To release the lock status, set all the LKPRI[7:0] bits to 0.

12.5 Caching Operation during Debugging

In debugging mode, the automatic lock function works on both the instruction cache and the data cache.
To execute the program in debugging mode with the same timings and performance as normal operating mode, set SBRK/CCU_CFG register to 0 and use only the hardware PC break function for suspending program execution.
Setting SBRK to 1 (default) allows use of both hardware PC break and software PC break. However, execution timings and performance will not be the same as those in normal operating mode.

Note: When SBRK is set to 0, a software PC break point cannot be set in the a target area for caching.

12.6 Cache Data Integrity

The CCU does not support a snooping function (for maintaining the data in the cache memory to match those in the external memory). The cache and the external memory are maintained in synch if reading/writing is only executed in the C33 PE Core. When data are transferred to the area subject to caching via DMAC or when data are written to the program area subject to caching by the CPU, flush the cache or otherwise ensure data integrity using software.

12.7 Control Register Details

Table 12.7.1 List of CCU Registers

Address	Register name		Function
0x302300	CCU_CFG	Cache Configuration Register	Enable instruction and data caches
0x302304	CCU_AREA	Cacheable Area Select Register	Select cacheable areas
0x302308	CCU_LK	Cache Lock Register	Configure cache lock function
0x30230c	CCU_STAT	Cache Status Register	Indicate cache statuses
0x302318	CCU_WB_STAT	Cache Write Buffer Status Register	Indicate write buffer status

The following describes each CCU register.

Note: When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

Cache Configuration Register (CCU_CFG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Cache Configuration Register (CCU_CFG)	**0x302300** (32 bits)	D31–9	–	reserved	–	–	–	0 when being read.
		D8	**WBEN**	Write buffer enable	1 Enable 0 Disable	1	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2	**SBRK**	Software break enable	1 Enable 0 Disable	1	R/W	
		D1	**IC**	Instruction cache enable	1 Enable 0 Disable	0	R/W	
		D0	**DC**	Data cache enable	1 Enable 0 Disable	0	R/W	

D[31:9] Reserved

D8 WBEN: Write Buffer Enable Bit

Enables the write buffer.
1 (R/W): Enabled (default)
0 (R/W): Disabled

When WBEN is set to 1 (default), the write buffer is enabled. When the write buffer is enabled, writing data to the external memory is performed in two steps, first the data is written to the write buffer and then the external memory is updated.
When WBEN is set to 0, the write buffer is disabled. In this case the CPU is placed into wait status until the data writing to the external memory has completed.

D[7:3] Reserved

D2 SBRK: Software Break Enable Bit

Enables the software PC break function during debugging.
1 (R/W): Enabled (default)
0 (R/W): Disabled

Note: When SBRK is set to 0, a software PC break point cannot be set in the a target area for caching.

D1 IC: Instruction Cache Enable Bit

Enables the instruction cache.
1 (R/W): Enabled
0 (R/W): Disable (default)

By setting IC to 1, addresses 0x0 to 0x3ff (1KB) in A0RAM are set for use as the instruction cache, after which the cache is used for fetching instructions from the specified area.
Setting IC to 0 flushes the instruction cache and clears all cache data. Note that the cache is flushed several cycles after writing 0 to IC. Before resuming caching operation, check ICS/CCU_STAT register to ensure that flushing is completed.

D0 DC: Data Cache Enable Bit

Enables the data cache.
1 (R/W): Enabled
0 (R/W): Disable (default)

By setting DC to 1, addresses 0x4c00 to 0x4fff (1KB) in A0RAM are set for use as the data cache, after which the cache is used for reading data from the specified area. Data is written in the write through mode.
Setting DC to 0 flushes the data cache and clears all cache data. Note that the cache is flushed several cycles after writing 0 to DC. Before resuming caching operation, check DCS/CCU_STAT register to ensure that flushing is completed.

Notes: • Be sure to disable the instruction and data caches before executing the halt or slp instruction.

• General-purpose RAM areas are allocated to the instruction and data caches. Be sure to avoid accessing these areas from the application when the cache is enabled.

Cacheable Area Select Register (CCU_AREA)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Cacheable Area Select Register (CCU_AREA)	**0x302304** (32 bits)	D31–7	–	reserved	–		–	–	0 when being read.
		D6–4	**ARIC[2:0]**	Instruction cache area select	ARIC[2:0]	Area	0x0	R/W	
					0x7	Area 22			
					0x6	Area 21			
					0x5	Area 20			
					0x4	Area 19			
					0x3	Area 18			
					0x2	Area 17			
					0x1	Areas 15 & 16			
					0x0	Area 14			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	**ARDC[2:0]**	Data cache area select	ARDC[2:0]	Area	0x0	R/W	
					0x7	Area 22			
					0x6	Area 21			
					0x5	Area 20			
					0x4	Area 19			
					0x3	Area 18			
					0x2	Area 17			
					0x1	Areas 15 & 16			
					0x0	Area 14			

D[31:7] Reserved

D[6:4] ARIC[2:0]: Instruction Cache Area Select Bits

Selects the area to read the instruction from via the instruction cache. (See Table 12.7.2)

D3 Reserved

D[2:0] ARDC[2:0]: Data Cache Area Select Bits

Selects the area to read the data from via the data cache.

Table 12.7.2 Selecting Area to Be Cached

ARIC[2:0]/ARDC[2:0]	Areas to be cached
0x7	Area 22 (0x80000000 to 0xffffffff)
0x6	Area 21 (0x40000000 to 0x7fffffff)
0x5	Area 20 (0x20000000 to 0x3fffffff)
0x4	Area 19 (0x10000000 to 0x1fffffff)
0x3	Area 18 (0x0c000000 to 0x0fffffff)
0x2	Area 17 (0x08000000 to 0x0bffffff)
0x1	Areas 15 and 16 (0x04000000 to 0x07ffffff)
0x0	Area 14 (0x03000000 to 0x03ffffff)

(Default: 0x0)

The CCU only caches access to the range of 64MB starting at the top of the selected area. The areas after the leading 16MB space in Areas 19 to 22 are mirror areas.

Cache Lock Register (CCU_LK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Cache Lock Register (CCU_LK)	0x302308 (32 bits)	D31–8	–	reserved	–	–	–	0 when being read.
		D7	LKPRI7	Interrupt level 7 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D6	LKPRI6	Interrupt level 6 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D5	LKPRI5	Interrupt level 5 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D4	LKPRI4	Interrupt level 4 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D3	LKPRI3	Interrupt level 3 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D2	LKPRI2	Interrupt level 2 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D1	LKPRI1	Interrupt level 1 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D0	LKPRI0	Interrupt level 0 cache-lock enable	1 Lock 0 Unlock	0	R/W	

D[31:8] Reserved

D[7:0] LKPRI[7:0]: Interrupt Level [7:0] Cache-Lock Enable Bits

Selects the interrupt levels to lock the cache (the interrupt levels of the interrupt handler routines to disable refilling).

1 (R/W): Lock the cache when IL[2:0] = specified interrupt level

0 (R/W): Release the cache lock when IL[2:0] = specified interrupt level (default)

By setting an LKPRI[7:0] bit to 1, the priority level of interrupts to disabled refilling can be selected. Each LKPRI[7:0] bit corresponds to an interrupt level, for example, LKPRI0 corresponds to interrupt level 0 (IL[2:0] = 0) and LKPRI7 corresponds to interrupt level 7 (IL[2:0] = 7). If the interrupt level in IL[2:0] (set by the interrupt occurred) and an LKPRI[7:0] bit that has been set to 1 are matched, the cache will be locked after a lapse of several cycles. After this point, the CCU will not refill the cache until the IL[2:0] value is altered to the interrupt level of an LKPRI[7:0] bit that has been set to 0.

Cache Status Register (CCU_STAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Cache Status Register (CCU_STAT)	0x30230c (32 bits)	D31–4	–	reserved	–	–	–	0 when being read.
		D3	ICLKS	Instruction cache lock status	1 Locked 0 Not locked	X	R	
		D2	DCLKS	Data cache lock status	1 Locked 0 Not locked	0	R	
		D1	ICS	Instruction cache operating status	1 Active 0 Inactive	0	R	
		D0	DCS	Data cache operating status	1 Active 0 Inactive	0	R	

D[31:4] Reserved

D3 ICLKS: Instruction Cache Lock Status Bit

Indicates the lock status of the instruction cache. (Default: undefined)

1 (R): Locked

0 (R): Not locked

Read ICLKS to check whether the instruction cache is locked or not when the cache lock function is enabled.

D2 DCLKS: Data Cache Lock Status Bit

Indicates the lock status of the data cache.

1 (R): Locked

0 (R): Not locked (default)

Read DCLKS to check whether the data cache is locked or not when the cache lock function is enabled.

D1 ICS: Instruction Cache Operating Status Bit

Indicates the operating status of the instruction cache.

1 (R): Active

0 (R): Inactive (default)

Setting IC/CCU_CFG register to 1 activates the instruction cache and sets ICS to 1. Setting IC to 0 flushes the instruction cache and disables caching. Note that the instruction cache is flushed several cycles after writing 0 to IC. When flushing finishes, ICS is reset to 0. To resume caching after flushing, check that flushing has completed by reading ICS.

D0 DCS: Data Cache Operating Status Bit

Indicates the operating status of the data cache.

1 (R): Active

0 (R): Inactive (default)

Setting DC/CCU_CFG register to 1 activates the data cache and sets DCS to 1. Setting DC to 0 flushes the data cache and disables caching. Note that the data cache is flushed several cycles after writing 0 to DC. When flushing finishes, DCS is reset to 0. To resume caching after flushing, check that flushing has completed by reading DCS.

Cache Write Buffer Status Register (CCU_WB_STAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Cache Write Buffer Status Register (CCU_WB_STAT)	0x302318 (32 bits)	D31–10	–	reserved	–	–	–	X when being read.
		D9	**WEFINISH**	Write-finish flag	1 Finished 0 Writing	1	R	
		D8	**WBEMPTY**	Write buffer empty flag	1 Empty 0 Full	1	R	
		D7–0	–	reserved	–	–	–	X when being read.

D[31:10] Reserved

D9 WEFINISH: Write-Finish Flag Bit

Indicates whether the data writing from the write buffer to the external memory has finished or not.

1 (R): Finished (default)

0 (R): During writing

When the write buffer is enabled, writing data to the external memory is performed in two steps, first the data is written to the write buffer and then the external memory is updated. WEFINISH is set to 0 when data is loaded to the write buffer and is set to 1 upon completion of writing to the external memory.

D8 WBEMPTY: Write Buffer Empty Flag Bit

Indicates the write buffer status.

1 (R): Empty (default)

0 (R): Full

WBEMPTY is set to 0 by writing data to the write buffer and is set to 1 when the buffered data is read out for writing to the external memory. The write buffer improves writing speed as the CPU does not need to wait for completion of writing to the external memory. However, read the above flags to check if the data has been written to, especially when data is written to a low-speed external device.

D[7:0] Reserved

13 Interrupt Controller (ITC)

13.1 ITC Module Overview

The interrupt controller (ITC) honors interrupt requests from the peripheral modules and outputs the interrupt request, interrupt level and vector number signals to the C33 PE Core according to the priority and interrupt levels. The features of the ITC module are listed below.

- Supports the following maskable interrupt systems:
  - Port input interrupt (two systems)
  - Key input interrupt (two systems)
  - DMAC interrupt (four systems)
  - 16-bit audio PWM timer (T16P) interrupt (two systems)
  - 16-bit PWM timer (T16A6) interrupt (four systems)
  - Fine mode 8-bit timer (T8F) interrupt (two systems)
  - A/D converter (ADC10) interrupt (one system)
  - RTC interrupt (one system)
  - USI interrupt (three systems)
  - USIL interrupt (one system)
  - UART interrupt (one system)
  - LCDC interrupt (one system)
  - SD/MMC interface (SD_MMC) interrupt (one system)
  - Remote controller (REMC) interrupt (one system)
  - I^2S interrupt (two systems)
  - HIF interrupt (one system)
  - USB interrupt (one system)
- Supports seven interrupt levels (1 to 7) to prioritize the interrupt sources.

The ITC enables the interrupt level (priority) for determining the processing sequence when multiple interrupts occur simultaneously to be set for each interrupt system separately.
Each interrupt system includes one or more interrupt causes. Settings to enable or disable interrupts for different causes are performed by the respective peripheral module registers.
For specific information on interrupt causes and their control, refer to the descriptions of the peripheral module.
Figure 13.1.1 shows the structure of the interrupt system.

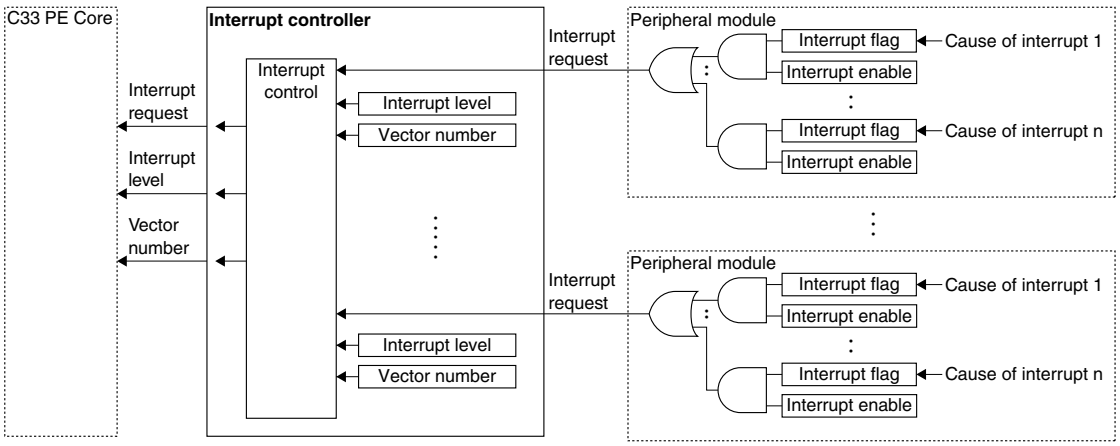


Figure 13.1.1 Interrupt System

13.2 Vector Table

The vector table contains the vectors to the interrupt handler routines (handler routine start address) that will be read by the C33 PE Core to execute the handler when an interrupt occurs.
Table 13.2.1 shows the vector table of the S1C33L27.

Table 13.2.1 Vector Table

Vector No.	Vector address	Interrupt name	Cause of interrupt	Priority
0 (0x0)	TTBR + 0x0	Reset	• Low input to the #RESET pin • Watchdog timer overflow *2	1
1 (0x1)	TTBR + 0x4	Reserved	–	–
2 (0x2)	TTBR + 0x8	ext exception	ext instruction (illegal use)	4
3 (0x3)	TTBR + 0xc	Undefined instruction exception	Undefined instruction	3
4 (0x4)	TTBR + 0x10	Reserved	–	–
5 (0x5)	TTBR + 0x14			
6 (0x6)	TTBR + 0x18	Address misaligned exception	Memory access instruction	2
–	(0x60000)	Debugging exception	brk instruction, etc.	5
7 (0x7)	TTBR + 0x1c	NMI	• Low input to the #NMI pin • Watchdog timer overflow *2 • Host processor software trigger *3	6
8 (0x8)	TTBR + 0x20	Reserved	–	–
9 (0x9)	TTBR + 0x24			
10 (0xa)	TTBR + 0x28			
11 (0xb)	TTBR + 0x2c			
12 (0xc)	TTBR + 0x30	Software exception 0	int instruction	High *1 ↑
13 (0xd)	TTBR + 0x34	Software exception 1	int instruction	
14 (0xe)	TTBR + 0x38	Software exception 2	int instruction	
15 (0xf)	TTBR + 0x3c	Software exception 3	int instruction	
16 (0x10)	TTBR + 0x40	Port input interrupt 0	FPT0–3 input (rising/falling edge or high/low level)	
17 (0x11)	TTBR + 0x44	Port input interrupt 1	FPT4–7 input (rising/falling edge or high/low level)	
18 (0x12)	TTBR + 0x48	Key input interrupt 0	FPK0 input (rising/falling edge or high/low level)	
19 (0x13)	TTBR + 0x4c	Key input interrupt 1	FPK1 input (rising/falling edge or high/low level)	
20 (0x14)	TTBR + 0x50	DMAC Ch.0/2 interrupt	End of DMA transfer	
21 (0x15)	TTBR + 0x54	DMAC Ch.1/3 interrupt	End of DMA transfer	
22 (0x16)	TTBR + 0x58	DMAC Ch.4/6 interrupt	End of DMA transfer	
23 (0x17)	TTBR + 0x5c	DMAC Ch.5/7 interrupt	End of DMA transfer	
24 (0x18)	TTBR + 0x60	16-bit audio PWM timer (T16P) Ch.0 interrupt	• Compare A/B • Buffer empty	
25 (0x19)	TTBR + 0x64	16-bit audio PWM timer (T16P) Ch.1 interrupt	• Compare A/B • Buffer empty	
26 (0x1a)	TTBR + 0x68	16-bit PWM timer (T16A6) Ch.0 interrupt	• Compare A/B • Capture A/B • Capture A/B overwrite	
27 (0x1b)	TTBR + 0x6c	16-bit PWM timer (T16A6) Ch.1 interrupt	• Compare A/B • Capture A/B • Capture A/B overwrite	
28 (0x1c)	TTBR + 0x70	16-bit PWM timer (T16A6) Ch.2 interrupt	• Compare A/B • Capture A/B • Capture A/B overwrite	
29 (0x1d)	TTBR + 0x74	16-bit PWM timer (T16A6) Ch.3 interrupt	• Compare A/B • Capture A/B • Capture A/B overwrite	
30 (0x1e)	TTBR + 0x78	Fine mode 8-bit timer (T8F) Ch.0/2/4 interrupt	Timer underflow	
31 (0x1f)	TTBR + 0x7c	Fine mode 8-bit timer (T8F) Ch.1/3/5 interrupt	Timer underflow	
32 (0x20)	TTBR + 0x80	A/D converter (ADC10) interrupt	• Conversion completion • Conversion result overwrite	
33 (0x21)	TTBR + 0x84	RTC interrupt	1/512 second, 1/256 second, 1/128 second, 1/64 second, 1 second, 1 minute, or 1 hour cycles	
34 (0x22)	TTBR + 0x88	USI Ch.0 interrupt	• Transmit buffer empty • Receive buffer full • Receive error	
		UART interrupt	• Transmit buffer empty • Receive buffer full • Receive error	
35 (0x23)	TTBR + 0x8c	USI Ch.1 interrupt	• Transmit buffer empty • Receive buffer full • Receive error	

Vector No.	Vector address	Interrupt name	Cause of interrupt	Priority
36 (0x24)	TTBR + 0x90	USI Ch.2 interrupt	• Transmit buffer empty • Receive buffer full • Receive error	
37 (0x25)	TTBR + 0x94	USIL interrupt	• Transmit buffer empty • Receive buffer full • Receive error	
38 (0x26)	TTBR + 0x98	LCDC interrupt	Beginning of a frame	
39 (0x27)	TTBR + 0x9c	SD/MMC interface (SD_MMC) interrupt	• Command without response • Command with response • FIFO half-empty • FIFO half-full • Response timeout error • Response CRC error • Data timeout error • Data CRC error	
40 (0x28)	TTBR + 0xa0	Remote controller (REMC) interrupt	• Data length counter underflow • Input rising edge detected • Input falling edge detected	
41 (0x29)	TTBR + 0xa4	I^2S Ch.0 (output) interrupt	• I^2S Ch.0 FIFO whole empty • I^2S Ch.0 FIFO half-empty • I^2S Ch.0 FIFO one empty	
42 (0x2a)	TTBR + 0xa8	I^2S Ch.1 (input) interrupt	• I^2S Ch.1 FIFO whole full • I^2S Ch.1 FIFO one data	
43 (0x2b)	TTBR + 0xac	Host interface (HIF) interrupt	Host processor software trigger *3	↓
44 (0x2c)	TTBR + 0xb0	USB interrupt	USB operation, bus and FIFO statuses	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.
*3 Either maskable interrupt or NMI can be selected as the HIF interrupt with software.

Vector numbers 16 to 44 are assigned to the maskable interrupts supported by the S1C33L27.

Interrupts that share an interrupt vector

Interrupt vector number 34 is shared with two different interrupt modules, USI Ch.0 and UART.

The interrupt signals from the two modules are input to the ITC through an OR gate.
The USI Ch.0 and UART interrupts can be used simultaneously. The interrupt vector 34 handler routine should check which interrupt has occurred by reading the interrupt flags in the USI Ch.0 and UART modules.

Vector table base address

The S1C33L27 allows the base (starting) address of the vector table to be set using the TTBR register. "TTBR" indicated in Table 13.2.1 means the value that is set to this register. Set the TTBR register in the initial routine executed after booting. Bits 9 to 0 of the TTBR register are fixed at 0. Therefore, the vector table starting address always begins with a 1 KB boundary address.

13.3 Control of Maskable Interrupts

13.3.1 Interrupt Control Bits in Peripheral Modules

The peripheral module that generates an interrupt includes an interrupt enable bit and an interrupt flag for each interrupt cause. The interrupt flag is set to 1 when the cause of interrupt occurs. By setting the interrupt enable bit to 1 (interrupt enabled), the flag state will be sent to the ITC as an interrupt request signal, generating an interrupt request to the C33 PE Core.
The corresponding interrupt enable bits should be set to 0 for those causes for which interrupts are not desired. In this case, although the interrupt flag is set to 1 if the interrupt cause occurs, the interrupt request signal sent to the ITC will not be asserted.
The interrupt flag set to 1 must be reset in the interrupt handler routine after the interrupt has occurred. The ITC will generate the same interrupt again once the interrupt handler routine has been ended by the reti instruction with the interrupt flag still set to 1, since it detects interrupt requests using the signal level.
For specific information on causes of interrupts, interrupt flags, and interrupt enable bits, refer to the respective peripheral module descriptions.

13.3.2 ITC Interrupt Request Processing

On receiving an interrupt signal from a peripheral module, the ITC sends the interrupt request, interrupt level, and vector number signals to the C33 PE Core.
Vector numbers are determined by the ITC internal hardware for each interrupt cause, as shown in Table 13.2.1.
The interrupt level is a value used by the C33 PE Core to compare with the IL bits (PSR). This interrupt level is used in the C33 PE Core to disable subsequently occurring interrupts with the same or lower level. (See Section 13.3.3.)
The default ITC settings are level 0 for all maskable interrupts. Interrupt requests are not accepted by the C33 PE Core if the level is 0.
The ITC includes control bits (INT_LV[2:0]/ITC_*xxx*_LV register) for selecting the interrupt level, and the level can be set to between 1 (low) and 7 (high) interrupt levels for each interrupt vector.
If interrupt requests are input to the ITC simultaneously from two or more peripheral modules, the ITC outputs the interrupt request with the highest priority to the C33 PE Core in accordance with the following conditions.

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

The other interrupts occurring at the same time are held until all interrupts with higher priority levels have been accepted by the C33 PE Core.
If an interrupt cause with higher priority occurs while the ITC is outputting an interrupt request signal to the C33 PE Core (before being accepted by the C33 PE Core), the ITC alters the vector number and interrupt level signals to the setting information on the more recent interrupt. The previously occurring interrupt is held. The held interrupt is canceled and no interrupt is generated if the interrupt flag in the peripheral module is reset with software.

Table 13.3.2.1 Interrupt Level Registers

Hardware interrupt	Interrupt level register	
Port input 0 interrupt (FPT0–3 interrupt)	ITC_PINT0_LV	(0x300210)
Port input 1 interrupt (FPT4–7 interrupt)	ITC_PINT1_LV	(0x300211)
Key input 0 interrupt (FPK0 interrupt)	ITC_KINT0_LV	(0x300212)
Key input 1 interrupt (FPK1 interrupt)	ITC_KINT1_LV	(0x300213)
DMAC Ch.0/2 interrupt	ITC_DMA02_LV	(0x300214)
DMAC Ch.1/3 interrupt	ITC_DMA13_LV	(0x300215)
DMAC Ch.4/6 interrupt	ITC_DMA46_LV	(0x300216)
DMAC Ch.5/7 interrupt	ITC_DMA57_LV	(0x300217)
16-bit audio PWM timer (T16P) Ch.0 interrupt	ITC_T16P0_LV	(0x300218)
16-bit audio PWM timer (T16P) Ch.1 interrupt	ITC_T16P1_LV	(0x300219)
16-bit PWM timer (T16A6) Ch.0 interrupt	ITC_T16A0_LV	(0x30021a)
16-bit PWM timer (T16A6) Ch.1 interrupt	ITC_T16A1_LV	(0x30021b)
16-bit PWM timer (T16A6) Ch.2 interrupt	ITC_T16A2_LV	(0x30021c)
16-bit PWM timer (T16A6) Ch.3 interrupt	ITC_T16A3_LV	(0x30021d)
Fine mode 8-bit timer (T8F) Ch.0/2/4 interrupt	ITC_T8F024_LV	(0x30021e)
Fine mode 8-bit timer (T8F) Ch.1/3/5 interrupt	ITC_T8F135_LV	(0x30021f)
A/D converter (ADC10) interrupt	ITC_ADC10_LV	(0x300220)
RTC interrupt	ITC_RTC_LV	(0x300221)
USI Ch.0/UART interrupt	ITC_USI0_LV	(0x300222)
USI Ch.1 interrupt	ITC_USI1_LV	(0x300223)
USI Ch.2 interrupt	ITC_USI2_LV	(0x300224)
USIL interrupt	ITC_USIL_LV	(0x300225)
LCDC interrupt	ITC_LCDC_LV	(0x300226)
SD_MMC interrupt	ITC_SDMMC_LV	(0x300227)
Remote controller (REMC) interrupt	ITC_REMC_LV	(0x300228)
I^2S Ch.0 interrupt	ITC_I2SOUT_LV	(0x300229)
I^2S Ch.1 interrupt	ITC_I2SIN_LV	(0x30022a)
Host interface (HIF) interrupt	ITC_HIF_LV	(0x30022b)
USB interrupt	ITC_USB_LV	(0x30022c)

13.3.3 Interrupt Processing by the C33 PE Core

A Maskable interrupt to the C33 PE Core occurs when all of the following conditions are met:

- The interrupt is enabled by the interrupt control bit inside the peripheral module.
- The IE (Interrupt Enable) bit of the PSR (Processor Status Register) in the C33 PE Core has been set to 1.
- The cause of interrupt that has occurred has a higher interrupt level than the value set in the IL field of the PSR.
- No other cause of interrupt having higher priority, such as NMI, has occurred.

If an interrupt cause that has been enabled in the peripheral module occurs, the corresponding interrupt flag is set to 1, and this state is maintained until it is reset by the program. This means that the interrupt cause is not cleared even if the conditions listed above are not met when the interrupt cause occurs. An interrupt occurs if the above conditions are met.
If multiple maskable interrupt causes occur simultaneously, the interrupt cause with the highest interrupt level and lowest vector number becomes the subject of the interrupt request to the C33 PE Core. Interrupts with lower levels are held until the above conditions are subsequently met.

The C33 PE Core samples interrupt requests for each cycle. On accepting an interrupt request, the C33 PE Core switches to interrupt processing immediately after execution of the current instruction has been completed.
Interrupt processing involves the following steps:
(1) The PSR and current program counter (PC) values are saved to the stack.
(2) The PSR IE bit is reset to 0 (disabling subsequent maskable interrupts).
(3) The PSR IL bits are set to the received interrupt level. (The NMI does not affect the IL bits.)
(4) The vector for the interrupt occurred is loaded to the PC to execute the interrupt handler routine.

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

13.4 NMI

In the S1C33L27, a low level input to the #NMI pin, the watchdog timer, or the external host processor connected to HIF can generate a non-maskable interrupt (NMI). The vector number for NMI is 7, with the vector address set to the vector table's starting address + 28 bytes.
This interrupt takes precedence over other interrupts and is unconditionally accepted by the C33 PE Core.
For detailed information on generating NMI by the watchdog timer, see the “Watchdog Timer (WDT)” chapter.
For detailed information on generating NMI by the external host processor, see the “Host interface (HIF)” chapter.

13.5 Software Exception

A software exception can be generated by use of “int imm2” instruction of the C33 PE Core. A software exception number (0 to 3) is specified with imm2 of the operand.

13.6 HALT and SLEEP Mode Cancellation

HALT mode is cleared by the following signals, which start the CPU.
- Interrupt request signal sent to the CPU from the ITC
- NMI signal output by the watchdog timer/host interface or input to the #NMI pin
- Debug interrupt signal
- Reset signal output by the watchdog timer or input to the #RESET pin

SLEEP mode is cleared by the following signals, which start the CPU.
- Port/key input interrupt or RTC interrupt request signal sent from the GPIO or RTC
- NMI signal input to the #NMI pin
- Debug interrupt signal
- Reset signal input to the #RESET pin

Notes: • If the CPU is able to receive interrupts when HALT or SLEEP mode has been cleared by an interrupt request for the CPU from the ITC, processing branches to the interrupt handler routine immediately after cancellation. In all other cases, the program is executed following the halt or slp instruction.

• HALT or SLEEP mode clearing due to interrupt requests cannot be masked (prohibited) using ITC interrupt level settings. When using a cause of interrupt to clear HALT or SLEEP mode, the interrupt enable bit corresponding to the cause of interrupt must be set to 1 (interrupt enabled).

For more information, see the "Power Saving" section in Appendix.

13.7 Control Register Details

Table 13.7.1 List of ITC Registers

Address	Register name		Function
0x300210	ITC_PINT0_LV	Port Input 0 Interrupt Level Register	Set port input 0 interrupt level
0x300211	ITC_PINT1_LV	Port Input 1 Interrupt Level Register	Set port input 1 interrupt level
0x300212	ITC_KINT0_LV	Key Input 0 Interrupt Level Register	Set key input 0 interrupt level
0x300213	ITC_KINT1_LV	Key Input 1 Interrupt Level Register	Set key input 1 interrupt level
0x300214	ITC_DMA02_LV	DMAC Ch.0/2 Interrupt Level Register	Set DMAC Ch.0 and 2 interrupt levels
0x300215	ITC_DMA13_LV	DMAC Ch.1/3 Interrupt Level Register	Set DMAC Ch.1 and 3 interrupt levels
0x300216	ITC_DMA46_LV	DMAC Ch.4/6 Interrupt Level Register	Set DMAC Ch.4 and 6 interrupt levels
0x300217	ITC_DMA57_LV	DMAC Ch.5/7 Interrupt Level Register	Set DMAC Ch.5 and 7 interrupt levels
0x300218	ITC_T16P0_LV	T16P Ch.0 Interrupt Level Register	Set T16P Ch.0 interrupt level
0x300219	ITC_T16P1_LV	T16P Ch.1 Interrupt Level Register	Set T16P Ch.1 interrupt level
0x30021a	ITC_T16A0_LV	T16A6 Ch.0 Interrupt Level Register	Set T16A6 Ch.0 interrupt level
0x30021b	ITC_T16A1_LV	T16A6 Ch.1 Interrupt Level Register	Set T16A6 Ch.1 interrupt level
0x30021c	ITC_T16A2_LV	T16A6 Ch.2 Interrupt Level Register	Set T16A6 Ch.2 interrupt level
0x30021d	ITC_T16A3_LV	T16A6 Ch.3 Interrupt Level Register	Set T16A6 Ch.3 interrupt level
0x30021e	ITC_T8F024_LV	T8F Ch.0/2/4 Interrupt Level Register	Set T8F Ch.0, 2, and 4 interrupt levels
0x30021f	ITC_T8F135_LV	T8F Ch.1/3/5 Interrupt Level Register	Set T8F Ch.1, 3, and 5 interrupt levels
0x300220	ITC_ADC10_LV	ADC10 Interrupt Level Register	Set ADC10 interrupt level
0x300221	ITC_RTC_LV	RTC Interrupt Level Register	Set RTC interrupt level
0x300222	ITC_USI0_LV	USI Ch.0/UART Interrupt Level Register	Set USI Ch.0 and UART interrupt levels
0x300223	ITC_USI1_LV	USI Ch.1 Interrupt Level Register	Set USI Ch.1 interrupt level
0x300224	ITC_USI2_LV	USI Ch.2 Interrupt Level Register	Set USI Ch.2 interrupt level
0x300225	ITC_USIL_LV	USIL Interrupt Level Register	Set USIL interrupt level
0x300226	ITC_LCDC_LV	LCDC Interrupt Level Register	Set LCDC interrupt level
0x300227	ITC_SDMMC_LV	SD_MMC Interrupt Level Register	Set SD_MMC interrupt level
0x300228	ITC_REMC_LV	REMC Interrupt Level Register	Set REMC interrupt level
0x300229	ITC_I2SOUT_LV	I^2S Output Interrupt Level Register	Set I^2S output channel interrupt level
0x30022a	ITC_I2SIN_LV	I^2S Input Interrupt Level Register	Set I^2S input channel interrupt level
0x30022b	ITC_HIF_LV	HIF Interrupt Level Register	Set HIF interrupt level
0x30022c	ITC_USB_LV	USB Interrupt Level Register	Set USB interrupt level

The ITC registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

Interrupt Level Registers (ITC_*xxx*_LV)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Interrupt Level Register (ITC_*xxx*_LV)	**0x300210 | 0x30022c** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	Interrupt level	1 to 7	0x0	R/W	

D[7:3] Reserved

D[2:0] INT_LV[2:0]: Interrupt Level Bits

Sets the interrupt level (1 to 7). (Default: 0x0)

The C33 PE Core does not accept interrupts with a level set lower than the PSR IL value.

The ITC uses the interrupt level when multiple interrupt requests occur simultaneously.

If multiple interrupt requests enabled by the interrupt enable bit occur simultaneously, the ITC sends the interrupt request with the highest level set by the ITC_*xxx*_LV registers (0x300210 to 0x30022c) to the C33 PE Core.
If multiple interrupt requests with the same interrupt level occur simultaneously, the interrupt with the lowest vector number is processed first. The other interrupts are held until all interrupts of higher priority have been accepted by the C33 PE Core.
If an interrupt requests of higher priority occurs while the ITC outputs an interrupt request signal to the C33 PE Core (before acceptance by the C33 PE Core), the ITC alters the vector number and interrupt level signals to the setting details of the most recent interrupt. The immediately preceding interrupt is held.

Note: Make sure that the PSR IE bit is set to 0 before setting the interrupt level registers (ITC_*xxx*_LV).

14 DMA Controller (DMAC)

14.1 DMAC Module Overview

The S1C33L27 incorporates a DMA controller (DMAC) capable of controlling eight table DMA channels.
The table DMA transfers data according to the control information programmed in the RAM.
The following shows the features of the DMAC.

- Number of channels: Maximum eight channels
- Control information: Programmable in the DSTRAM, IVRAM (Area 3) or an external RAM (16-byte control information + 16-byte backup data per channel)
- Dual-address transfer: DSTRAM, IVRAM (Area 3), an external memory or an internal peripheral module area (from address 0x300000) can be specified as the transfer source and destination.
  1. Data transfer within DSTRAM or IVRAM
  2. Data transfer between DSTRAM or IVRAM and an external memory
  3. Data transfer between DSTRAM or IVRAM and an internal peripheral module area
  4. Data transfer between an external memory and an external memory
  5. Data transfer between an external memory and an internal peripheral module area
  6. Data transfer within internal peripheral module area

  * Note that Area 0 (including IVRAM relocated to Area 0) cannot be specified as the transfer source and destination.
- Transfer data size: 8 bits, 16 bits or 32 bits
- Transfer mode:
  1. Single transfer (one unit of data is transferred by one trigger)
  2. Successive transfer (specified number of data are transferred by one trigger, with 12-bit transfer counter)
- Transfer address control: The source and/or destination addresses can be fixed or incremented in units of the transfer data size upon completion of transfer.
- Trigger:
  1. Software trigger via register control
  2. Hardware trigger by interrupt source modules (USI, USIL, I2S, T16P, T16A6, ADC10, GPIO, USB, SD_MMC)
- Pointer transfer: Transfer data can be specified using the specified source as a pointer.
  Transfer data = *(Pointer base address + *(Source address))
- Interrupt: End-of-transfer interrupt
- Others:
  - Auto-reload function for the identical DMA transfers without resetting
  - A DMA pause (temporary standby) function of low-priority channel DMA by high-priority trigger

14.2 DMAC Operating Clock

The DMAC operates with PCLK supplied from the CMU. A CMU register is used for clock supply control (default: enabled). For more information on the clock control, see the “Clock Management Unit (CMU)” chapter. In SLEEP mode, the CMU stops supplying PCLK to the DMAC.

14.3 Programming Control Information

The DMAC operates according to the transfer conditions specified with control information. The control information must be programmed in DSTRAM, IVRAM (Area 3) or an external RAM. The control information size is 4 words per channel. When using the auto-reload function, each channel needs additional 4 words for storing reload data (control information resetting data). The auto-reload function resets control information, which is updated during data transfer, with the reload data after a DMA transfer has finished. Therefore, a consecutive 256-byte space is required for the control table to use eight DMA channels.

The following explains how to set the base address of the control table and the contents of control information.

14.3.1 Setting the Base Address

The RAM area beginning with the specified base address is allocated to the control table. The base address is the start address of the control information for Ch.0 and can be specified using TBL_BASE[31:0]/DMAC_TBL_BASE register. TBL_BASE[9:0] is fixed at 0 regardless of the contents written, therefore the base address is always set to 1,024-byte boundary address. The initial value of TBL_BASE[31:0] is 0x84000 (DSTRAM).

Base + 0xf0	Ch.7 auto-reload data area
Base + 0xe0	Ch.7 control table
Base + 0xd0	Ch.6 auto-reload data area
Base + 0xc0	Ch.6 control table
Base + 0xb0	Ch.5 auto-reload data area
Base + 0xa0	Ch.5 control table
Base + 0x90	Ch.4 auto-reload data area
Base + 0x80	Ch.4 control table
Base + 0x70	Ch.3 auto-reload data area
Base + 0x60	Ch.3 control table
Base + 0x50	Ch.2 auto-reload data area
Base + 0x40	Ch.2 control table
Base + 0x30	Ch.1 auto-reload data area
Base + 0x20	Ch.1 control table
Base + 0x10	Ch.0 auto-reload data area
Base	Ch.0 control table

Figure 14.3.1.1 Control Table Map

Note: The control table must be placed on DSTRAM, IVRAM (Area 3) or an external RAM. A0RAM/IRAM (Area 0) and BBRAM cannot be used to store control information.

14.3.2 Control Information

The address to store control information is determined by the base address and a channel number.

Start address of channel = base address + (channel number × 32 [bytes])

Note: The control information must be written only when the channel to be configured does not start a DMA transfer. If a DMA transfer starts when the control information is being written, proper transfer cannot be performed. Reading the control information can always be performed.

The contents of control information in each channel are shown in the table below.

Table 14.3.2.1 Control Information

Word	Address	Bit	Name	Function	Setting	
1st word (32 bits)	Ch.0: Base + 0x0	D31–24	–	reserved	–	
	Ch.1: Base + 0x20 Ch.2: Base + 0x40 Ch.3: Base + 0x60 Ch.4: Base + 0x80 Ch.5: Base + 0xa0 Ch.6: Base + 0xc0	D23–12	**TC[11:0]**	Transfer counter	TC[11:0]	Count
					0xfff	4,095
					:	:
					0x1	1
					0x0	4,096
	Ch.7: Base + 0xe0	D11	**ST**	Source type	1 Pointer	0 Data

Word	Address	Bit	Name	Function	Setting
1st word (32 bits)	Ch.0: Base + 0x0 Ch.1: Base + 0x20 Ch.2: Base + 0x40 Ch.3: Base + 0x60 Ch.4: Base + 0x80 Ch.5: Base + 0xa0 Ch.6: Base + 0xc0 Ch.7: Base + 0xe0	D10–8	**UNIT[2:0]**	Transfer data unit	UNIT[2:0] → Data unit 0x7–0x3 → reserved 0x2 → 32 bits 0x1 → 16 bits 0x0 → 8 bits
		D7–6	**SRINC[1:0]**	Source address control	SRINC[1:0] → Address 0x3–0x2 → reserved 0x1 → Increment 0x0 → Fixed
		D5–4	**DSINC[1:0]**	Destination address control	DSINC[1:0] → Address 0x3–0x2 → reserved 0x1 → Increment 0x0 → Fixed
		D3	**CHEN**	Channel enable	1 Enable / 0 Disable
		D2	**TM**	Transfer mode	1 Successive / 0 Single
		D1	**RELOAD**	Auto-reload enable	1 Enable / 0 Disable
		D0	**PTW**	Pointer bit width	1 8 bits / 0 16 bits
2nd word (32 bits)	Ch.0: Base + 0x4 Ch.1: Base + 0x24 Ch.2: Base + 0x44 Ch.3: Base + 0x64 Ch.4: Base + 0x84 Ch.5: Base + 0xa4 Ch.6: Base + 0xc4 Ch.7: Base + 0xe4	D31–0	**SRADR[31:0]**	Source address/source data pointer	0x0 to 0xffffffff
3rd word (32 bits)	Ch.0: Base + 0x8 Ch.1: Base + 0x28 Ch.2: Base + 0x48 Ch.3: Base + 0x68 Ch.4: Base + 0x88 Ch.5: Base + 0xa8 Ch.6: Base + 0xc8 Ch.7: Base + 0xe8	D31–0	**DSADR[31:0]**	Destination address	0x0 to 0xffffffff
4th word (32 bits)	Ch.0: Base + 0xc Ch.1: Base + 0x2c Ch.2: Base + 0x4c Ch.3: Base + 0x6c Ch.4: Base + 0x8c Ch.5: Base + 0xac Ch.6: Base + 0xcc Ch.7: Base + 0xec	D31–16	**PTBASE [31:16]**	Pointer base address (high-order 16 bits)	0x0 to 0xffff (PTBASE[31:0] = 0x0 to 0xffff0000)
		D15–0	**PTBASE [15:0]**	Fix at 0 (Pointer base address low-order 16 bits)	0x0
5th word (32 bits)	Ch.0: Base + 0x10 Ch.1: Base + 0x30 Ch.2: Base + 0x50 Ch.3: Base + 0x70 Ch.4: Base + 0x90 Ch.5: Base + 0xb0 Ch.6: Base + 0xd0 Ch.7: Base + 0xf0	D31–0	**RELOAD0 [31:0]**	Reload data 0	(Same contents as 1st word)
6th word (32 bits)	Ch.0: Base + 0x14 Ch.1: Base + 0x34 Ch.2: Base + 0x54 Ch.3: Base + 0x74 Ch.4: Base + 0x94 Ch.5: Base + 0xb4 Ch.6: Base + 0xd4 Ch.7: Base + 0xf4	D31–0	**RELOAD1 [31:0]**	Reload data 1	(Same contents as 2nd word)
7th word (32 bits)	Ch.0: Base + 0x18 Ch.1: Base + 0x38 Ch.2: Base + 0x58 Ch.3: Base + 0x78 Ch.4: Base + 0x98 Ch.5: Base + 0xb8 Ch.6: Base + 0xd8 Ch.7: Base + 0xf8	D31–0	**RELOAD2 [31:0]**	Reload data 2	(Same contents as 3rd word)
8th word (32 bits)	Ch.0: Base + 0x1c Ch.1: Base + 0x3c Ch.2: Base + 0x5c Ch.3: Base + 0x7c Ch.4: Base + 0x9c Ch.5: Base + 0xbc Ch.6: Base + 0xdc Ch.7: Base + 0xfc	D31–0	**RELOAD3 [31:0]**	Reload data 3	(Same contents as 4th word)

TC[11:0]: Transfer counter (D[23:12]/1st word)

Set the number of times for unit data transfers to be executed. Writing 0x1 to 0xfff sets the transfer count to 1 to 4,095 and writing 0x0 sets it to 4,096. After a transfer of the data unit specified in UNIT[2:0] is completed, the transfer counter is decremented.

ST: Source type (D11/1st word)

Selects whether the memory contents on the specified source address are used as data or pointers.

ST = 0: Data

The DMAC transfers the data stored in the source address to the destination address.

ST = 1: Pointer

The DMAC uses the specified source address as a pointer and determines the address in which transfer data is stored as follows:
Transfer data = *(Base address + *(Source address))

Example: When base address (PTBASE[31:16]) = 0x85 (i.e., address 0x850000), source address (SRADR[31:0]) = 0xfc000, and the contents of address 0xfc000 = 0x2
The data stored in address 0x850002 is transferred.

UNIT[2:0]: Transfer data unit (D[10:8]/1st word)

Sets the data size for the transfer unit.

Table 14.3.2.2 Transfer Data Unit

UNIT[2:0]	Transfer data unit
0x7–0x3	Reserved
0x2	32 bits
0x1	16 bits
0x0	8 bits

SRINC[1:0]: Source address control (D[7:6]/1st word)

Sets the control method for the source address after a unit data transfer.

Table 14.3.2.3 Source Address Control

SRINC[1:0]	Source address control
0x3–0x2	Reserved
0x1	Increment
0x0	Fixed

SRINC[1:0] = 0x0: Address fixed

The source address is not changed by a data transfer performed. Even when transferring multiple data, the transfer data is always read from the same address.

SRINC[1:0] = 0x1: Address increment

After a transfer of a data unit specified with UNIT[2:0] is completed, the source address is incremented for the transferred data unit. The address that has been incremented during transfer does not return to the initial value.

DSINC[1:0]: Destination address control (D[5:4]/1st word)

Sets the control method for the destination address after a unit data transfer.

Table 14.3.2.4 Destination Address Control

DSINC[1:0]	Destination address control
0x3–0x2	Reserved
0x1	Increment
0x0	Fixed

DSINC[1:0] = 0x0: Address fixed

The destination address is not changed by a data transfer performed. Even when transferring multiple data, the transfer data is always written to the same address.

DSINC[1:0] = 0x1: Address increment

After a transfer of a data unit specified with UNIT[2:0] is completed, the destination address is incremented for the transferred data unit. The address that has been incremented during transfer does not return to the initial value.

CHEN: Channel enable (D3/1st word)

Enables or disables DMA transfers in each channel.

CHEN = 0: Transfer disabled

DMA transfers in the channel with this bit set to 0 are disabled.

CHEN = 1: Transfer enabled

DMA transfers in the channel are enabled. When triggered in this status, the DMAC starts DMA transfers in that channel (unless the transfer pausing function is operating, triggered by a high-priority channel).
CHEN is cleared upon completion of the current transfer. When the RELOAD bit (D1/1st word) is set to 0, this disables subsequent DMA transfers. When RELOAD is 1, CHEN will be replaced with RELOAD03 (D3/reload data 0) by auto-reloading. Therefore, if RELOAD03 is set to 1, CHEN will be set to 1 again enabling the DMAC to accept the subsequent DMA trigger without setting via software.

TM: Transfer mode (D2/1st word)

Sets the transfer mode (single transfer mode, successive transfer mode).

TM = 0: Single transfer mode

In this mode, a transfer operation invoked by one trigger is completed after transferring one data unit of the size set in UNIT[2:0]. If a data transfer needs to be performed a number of times as set by the transfer counter, an equal number of triggers are required.

TM = 1: Successive transfer mode

In this mode, one trigger performs data transfer a number of times as set by the transfer counter. The transfer counter is decremented each time a unit data is transferred, and successive transfers end when the counter reaches 0.

RELOAD: Auto-reload enable (D1/1st word)

Enables or disables the auto-reload function. The auto-reload function resets the initial value of control information set in the auto-reload area (configured the same as the four words in the control table) in the control table as soon as the transfer counter reaches 0. With this function, you can execute transfers with the new conditions without resetting the initial value in the DMAC interrupt handler routine.

RELOAD = 0: Auto-reload disabled

Setting this bit to 0 disables the auto-reload function. Furthermore, the CHEN bit is set to 0 when the transfer counter reaches 0, disabling subsequent DMA transfers. The control table retains the contents when the transfer counter reaches 0.

RELOAD = 1: Auto-reload enabled

Setting this bit to 1 enables the auto-reload function. When the transfer counter reaches 0, the control information stored in the auto-reload area is reset on the control table. The next trigger executes DMA transfer with the new conditions set in the reload data.

Note: Control information (initial value) for the auto-reload area should be prepared in the application program.

PTW: Pointer bit width (D0/1st word)

Sets the pointer size when pointer is selected for the source type (ST).

- **PTW = 0: 16 bits**

  DMAC performs a 16-bit read from the specified source address to obtain the pointer.

- **PTW = 1: 8 bits**

  DMAC performs an 8-bit read from the specified source address to obtain the pointer.

SRADR[31:0]: Source address (D[31:0]/2nd word)

Set the start address of the transfer source (or the pointer to the transfer source). This setting is updated according to the setting of SRINC[1:0].

DSADR[31:0]: Destination address (D[31:0]/3rd word)

Set the start address of the transfer destination. This setting is updated according to the setting of DSINC[1:0].

PTBASE[31:16]: Pointer base address (D[31:16]/4th word)

Set the pointer base address (see ST) when pointer is selected for the source type. The low-order 16 bits of the base address are fixed at 0x0 (D[15:0] is ignored).

14.3.3 Auto-Reload Data

As shown in Figure 14.3.1.1, a RAM area is allocated to an auto-reloading data area for each channel along with the control table. When the auto-reload function is enabled by setting RELOAD (D1/1st word) to 1, the contents of the auto-reloading data area will be reset on the control table when the transfer counter reaches 0, enabling transfers with the new conditions to be executed without setting of the conditions in the DMAC interrupt handler routine. The four words (32 bytes) in the auto-reloading area are handled as exactly the same bit configuration as the four words on the control table. The auto-reloading area can be used as a control information buffer. Before using a DMAC channel with the auto-reload function, write the first transfer conditions with RELOAD set to 1 to the control table and the second transfer conditions to the auto-reloading area. The control information written to the auto-reloading area is loaded to the control table upon completion of the first data transfer and it will control the second data transfer. If the auto-reload function is not used, the control table must be reset to the subsequent transfer conditions in the DMAC interrupt handler routine.
The address of the auto-reload data area can be calculated from the equation below.

Start address of auto-reloading data area = base address + (channel number × 32) + 16

14.4 DMAC Invocation

The triggers by which DMA is invoked have the following two causes:
1. Software trigger via register control
2. Hardware trigger due to a cause of interrupt in internal peripheral modules

Enabling DMAC

Each DMAC channel enters ready-to-operate status by setting DMAON*x*/DMAC_CH_EN register to 1. When DMAON*x* is 0 (default), the DMA channel does not accept triggers even if the control information enables transfers.

Enabling DMA transfers

Writing 1 to the CHEN bit (D3/1st word) on the control table enables DMA transfers in that channel, making it ready to accept triggers.

DMAC invocation by a software trigger

Any DMAC channel can be invoked via software. In order to invoke DMA transfer using Ch.*x*, write 1 to TRG*x*/DMAC_TRG_FLG register.
TRG*x* retains 1 until the DMA request is accepted and then it is reset to 0 by the hardware. TRG*x* is also set to 1 by a hardware trigger.

DMAC invocation by a cause of interrupt in internal peripheral modules

To respective channels of the DMAC, hardware trigger sources (causes of interrupt in peripheral modules) shown in Table 14.4.1 are assigned, which can be selected with the TRG_SEL*x*[1:0]/DMAC_TRG_SEL register.

Table 14.4.1 DMAC Trigger Source

Channel	Control bits	Setting	Trigger source	Channel priority
Ch.7	TRG_SEL7[1:0]	0x3	A/D converter (ADC10) conversion completion	Low
		0x2	USI Ch.2 transmit buffer empty	↑
		0x1	USIL transmit buffer empty	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.6	TRG_SEL6[1:0]	0x3	USB interrupt	
		0x2	USI Ch.2 receive buffer full	
		0x1	USIL receive buffer full	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.5	TRG_SEL5[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture A *	
		0x2	SD/MMC interface (SD_MMC) FIFO half-empty	
		0x1	USI Ch.1 transmit buffer empty	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.4	TRG_SEL4[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture B *	
		0x2	SD/MMC interface (SD_MMC) FIFO half-full	
		0x1	USI Ch.1 receive buffer full	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.3	TRG_SEL3[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture A *	
		0x2	I^2S input (R)	
		0x1	USI Ch.0 transmit buffer empty	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.2	TRG_SEL2[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture B *	
		0x2	I^2S input (L)	
		0x1	USI Ch.0 receive buffer full	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.1	TRG_SEL1[1:0]	0x3	Port input interrupt	
		0x2	16-bit audio PWM timer (T16P) Ch.1 buffer empty	
		0x1	I^2S output (R)	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.0	TRG_SEL0[1:0]	0x3	A/D converter (ADC10) conversion completion	
		0x2	16-bit audio PWM timer (T16P) Ch.0 buffer empty	
		0x1	I^2S output (L)	↓
		0x0	Hardware trigger disabled (software trigger only)	High

* Set the T16A6 channel for invoking the DMAC using DMASEL[1:0]/T16A_CTL*x* register. (Default: 0x0)

At initial reset, TRG_SEL*x*[1:0] in all channels are set to 0x0 (hardware trigger disabled). Note that software triggers are enabled regardless of the trigger source selected.

These trigger sources (causes of interrupt) are used in common for interrupt requests and DMAC invocation requests. When interrupts due to the cause used for a trigger is enabled and the interrupt level is set to 1 or more, an interrupt is also generated simultaneously with the trigger for the DMAC. When an interrupt vector and handler routine are located in A0RAM, interrupt handling can be executed even during a DMA transfer. An instruction for accessing the transfer source/destination is not executed until the DMA transfer is completed. When only invoking the DMAC and not using an interrupt, set the interrupt enable bit to 0 (interrupt disabled).

DMA request generated during a DMA transfer

A low-priority DMA request generated during a DMA transfer is not accepted until the transfer currently being executed is completed (until the unit data transfer is completed in the single transfer mode or until the transfer counter reaches 0 in the successive transfer mode).

A DMA request for another high-priority channel that is generated during successive transfers in a channel is accepted after the transfer of the current data unit is completed. The current DMA transfer is suspended at that point, and is resumed after that high-priority DMA transfer generated later is completed.

DMA request when the channel is disabled to transfer

Triggers are disabled for a channel with the CHEN bit (D3/1st word) set to 0 (DMA transfer disabled).

TRG*x* for the channel will not be set.

14.5 Operation of DMAC

The DMAC has two transfer modes (single and successive transfer modes), in each of which data transfer operates differently. The following describes the operation in each transfer mode.

14.5.1 Single Transfer Mode

The channels for which TM (D2/1st word) in control information is set to 0 operate in single transfer mode. In this mode, a transfer operation invoked by one trigger is completed after transferring one data unit of the size set in UNIT[2:0]. If data transfer needs to be performed a number of times as set by the transfer counter, an equal number of triggers are required. The operation in the single transfer mode is shown by the flow chart in Figure 14.5.1.1.

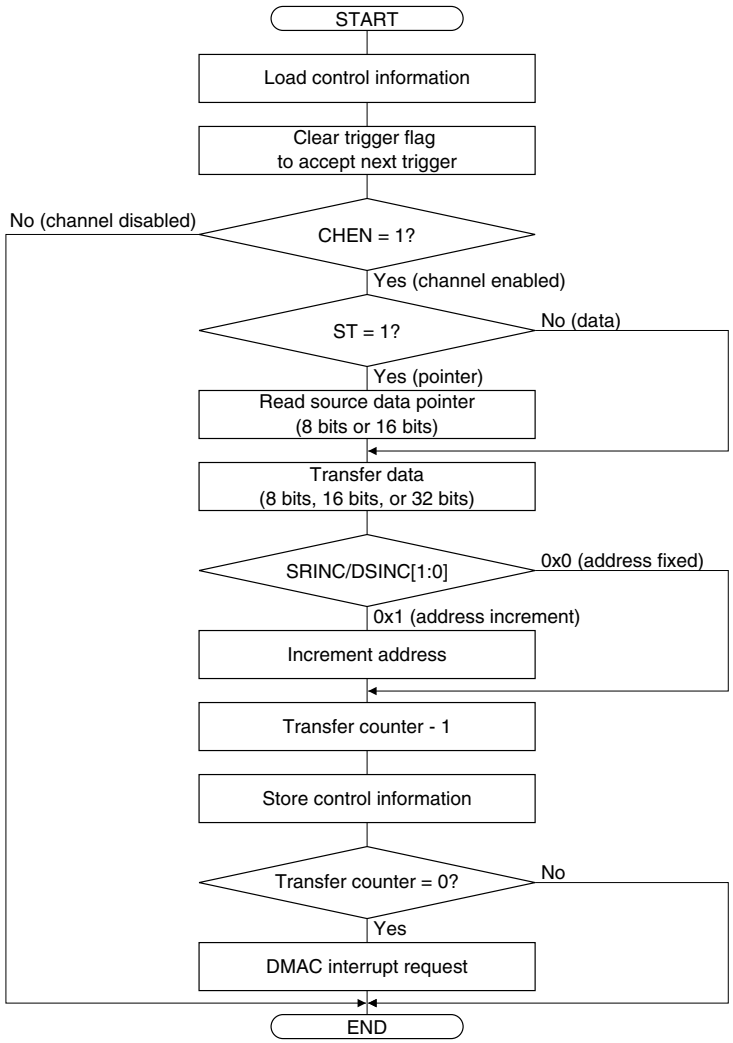


Figure 14.5.1.1 Operation Flow in Single Transfer Mode

(1) When the DMAC accepts a trigger, it loads the control information of the channel into the DMAC module.

(2) To allow the next trigger, the DMAC clears the trigger flag (TRG*x*/DMAC_TRG_FLG register).

(3) The DMAC checks to see if CHEN is set to 1 (DMA transfer enabled). It abort data transfer if CHEN is set to 0.

(4) If the source type specified in the control information is pointer (ST = 1), the DMAC read the contents of the specified source address to determine the pointer to the source data.

(5) The DMAC reads the specified data unit from the source address into a buffer and then write it to the destination address.
The transfer status flag (RUN*x*/DMAC_RUN_STAT register) is set and retains 1 while data is being transferred.

(6) According to the control information, the DMAC increments the source and/or destination addresses. The addresses are not changed if "address fixed" is specified. Also the transfer counter is decremented.

(7) The DMAC writes the modified control information back to the control table.

(8) The DMAC checks the transfer counter. If the value of the counter is not 0, the process is terminated here. Step (9) is not executed. Step (9) is executed if the transfer counter has reached 0.

(9) The DMAC sets the end-of-transfer flag (ENDF*x*/DMAC_END_FLG register) and clears the transfer status flag (RUN*x*). If DMAIE*x*/DMAC_IE register is set to 1 (end-of-transfer interrupt enabled), the DMAC outputs an interrupt request to the ITC.

This completes the single transfer process.

14.5.2 Successive Transfer Mode

The channels for which TM (D2/1st word) in control information is set to 1 operate in the successive transfer mode. In this mode, a data transfer is performed by one trigger a number of times as set by the transfer counter.
The operation in the successive transfer mode is shown by the flow chart in Figure 14.5.2.1.

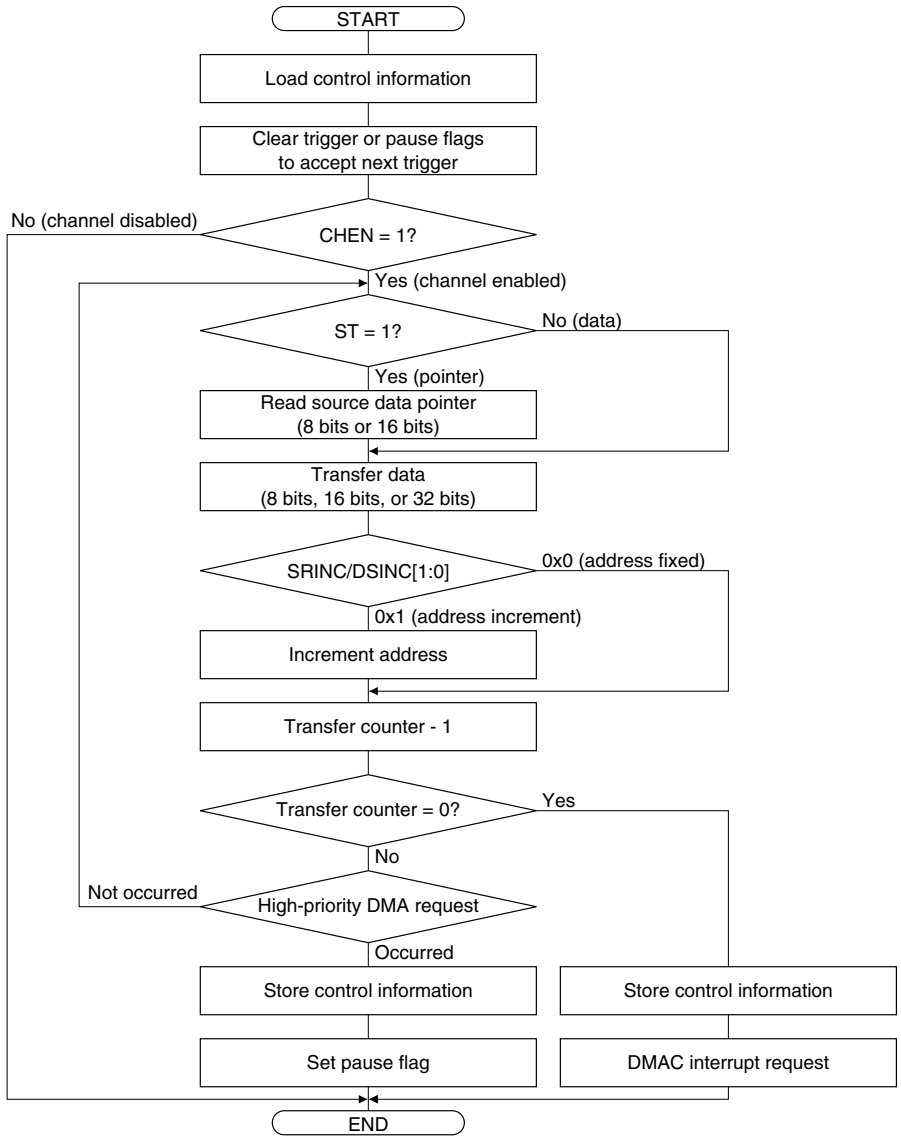


Figure 14.5.2.1 Operation Flow in Successive Transfer Mode

(1) When the DMAC accepts a trigger (or pause status is released), it loads the control information of the channel into the DMAC module.

(2) To allow the next trigger, the DMAC clears the trigger flag (TRG*x*/DMAC_TRG_FLG register) or the pause flag (PAUSE*x*/DMAC_PAUSE_STAT register) according to the cause of the current DMA transfer.

(3) The DMAC checks to see if CHEN is set to 1 (DMA transfer enabled). It abort data transfer if CHEN is set to 0.

(4) If the source type specified in the control information is pointer (ST = 1), the DMAC read the contents of the specified source address to determine the pointer to the source data.

(5) The DMAC reads the specified data unit from the source address into a buffer and then write it to the destination address.
The transfer status flag (RUN*x*/DMAC_RUN_STAT register) is set and retains 1 while data is being transferred.

(6) According to the control information, the DMAC increments the source and/or destination addresses. The addresses are not changed if "address fixed" is specified. Also the transfer counter is decremented.

(7) The DMAC checks the transfer counter. It goes to Step (10) if the transfer counter has reaches 0.

(8) The DMAC checks to see if any DMA request has been generated from other high-priority channels. If a high-priority trigger flag is set, the DMAC sets the pause flag (PAUSE*x*) of the channel currently performing a transfer and suspends the transfer. The suspended DMA transfer will resume after other high-priority DMA transfers are completed.

(9) If no DMA request has issued from high-priority channels, the DMAC returns to Step (4) to transfer the next data unit.

(10) The DMAC sets the end-of-transfer flag (ENDF*x*/DMAC_END_FLG register) and clears the transfer status flag (RUN*x*). Also it writes the modified control information back to the control table. If DMAIE*x*/DMAC_IE register is set to 1 (end-of-transfer interrupt enabled), the DMAC outputs an interrupt request to the ITC.

This completes the successive transfer process.

Suspending successive transfers due to other high-priority DMA request

Successive transfers can be temporarily suspended due to occurrence of a high-priority DMA request.
When a high-priority DMA request is generated, the channel performing a transfer saves control information required for resuming transfers (such as the current transfer count and the transfer source and destination addresses) as soon as the current data unit transfer is completed and then suspends transfers. At the same time, the pause flag (PAUSE*x*/DMAC_PAUSE_STAT register) in the suspended channel is set.
After that, the higher-priority DMA transfer is executed. After the transfer is completed, the suspended DMA transfer is resumed by PAUSE*x* that has been set. PAUSE*x* is cleared when the DMA transfer is resumed.

Notes: • Single transfers cannot be suspended.

• The software triggered high-priority DMA request can not suspend any low-priority DMA transfer being performed.

14.6 DMAC Interrupt

The DMAC module includes a function for generating interrupts upon completion of a data transfer.
For the interrupts generated from trigger sources, see the descriptions of the peripheral modules.

End-of-transfer interrupt

This cause of interrupt occurs when a transfer in a channel has completed (when the transfer counter has reaches 0) and sets ENDF*x*/DMAC_END_FLG corresponding to the channel to 1.
To use this interrupt, set DMAIE*x*/DMAC_IE register to 1. When DMAIE*x* is set to 0 (default), interrupt requests for this interrupt cause are not sent to the interrupt controller (ITC).
If ENDF*x* is set to 1 while DMAIE*x* is set to 1 (interrupt enabled), the DMAC module outputs an interrupt request to the ITC. An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied.
To check the channel that has completed a data transfer, read ENDF*x* in the interrupt handler routine.
For more information on interrupt processing, see the "Interrupt Controller (ITC)" chapter.

Notes: • To prevent interrupt recurrences, the DMAC module interrupt flag ENDF*x* must be reset in the interrupt handler routine after a DMAC interrupt has occurred.

• To prevent unwanted interrupts, ENDF*x* should be reset before enabling DMAC interrupts with DMAIE*x*. ENDF*x* can be reset to 0 by writing 1.

14.7 Control Register Details

Table 14.7.1 List of DMAC Registers

Address	Register name		Function
0x302100	DMAC_CH_EN	DMAC Channel Enable Register	Enable DMAC channels
0x302104	DMAC_TBL_BASE	DMAC Control Table Base Address Register	Set control table base address
0x302108	DMAC_IE	DMAC Interrupt Enable Register	Enable/disable DMAC interrupts
0x30210c	DMAC_TRG_SEL	DMAC Trigger Select Register	Select trigger sources
0x302110	DMAC_TRG_FLG	DMAC Trigger Flag Register	Control software trigger and indicate trigger status
0x302114	DMAC_END_FLG	DMAC End-of-Transfer Flag Register	Indicate DMA completed channels
0x302118	DMAC_RUN_STAT	DMAC Running Status Register	Indicates running channel
0x30211c	DMAC_PAUSE_STAT	DMAC Pause Status Register	Indicate DMA suspended channels

The DMAC module registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

DMAC Channel Enable Register (DMAC_CH_EN)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
DMAC Channel Enable Register (DMAC_CH_EN)	**0x302100** (32 bits)	D31–8	**–**	reserved	–				–	–	0 when being read.
		D7	**DMAON7**	DMAC Ch.7 enable	1	Enable	0	Disable	0	R/W	
		D6	**DMAON6**	DMAC Ch.6 enable	1	Enable	0	Disable	0	R/W	
		D5	**DMAON5**	DMAC Ch.5 enable	1	Enable	0	Disable	0	R/W	
		D4	**DMAON4**	DMAC Ch.4 enable	1	Enable	0	Disable	0	R/W	
		D3	**DMAON3**	DMAC Ch.3 enable	1	Enable	0	Disable	0	R/W	
		D2	**DMAON2**	DMAC Ch.2 enable	1	Enable	0	Disable	0	R/W	
		D1	**DMAON1**	DMAC Ch.1 enable	1	Enable	0	Disable	0	R/W	
		D0	**DMAON0**	DMAC Ch.0 enable	1	Enable	0	Disable	0	R/W	

D[31:8] Reserved

D[7:0] DMAON*x*: DMAC Ch.*x* Enable Bit

Enables DMAC Ch.*x* to accept DMA triggers.
1 (R/W): Enabled
0 (R/W): Disabled/Forced termination (default)

To perform DMA transfer using DMAC Ch.*x*, write 1 to DMAON*x*. When DMAON*x* is 0, DMAC Ch.*x* does not accept triggers and data transfer cannot be started.

DMAC Control Table Base Address Register (DMAC_TBL_BASE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
DMAC Control Table Base Address Register (DMAC_TBL_BASE)	**0x302104** (32 bits)	D31–10	**TBL_BASE [31:10]**	DMAC control table base address	0x0 to 0xfffffc00 (1,024-byte boundary address within a RAM)	0x84000	R/W	
		D9–0	**TBL_BASE [9:0]**	Fixed at 0x0 (Cannot be altered.)			R	

D[31:0] TBL_BASE[31:0]: DMAC Control Table Base Address Bits

Sets a base address for the control table for writing control information and auto reload information.
The size of control information is 4 words (16 bytes) per channel. The area for auto-reloading also requires 4 words (16 bytes) per channel. Therefore, a consecutive 256-byte space is needed for the control table in order to support eight channels.
The control table is secured in the RAM with the base address specified in these registers assumed to be the start address of the control information for Ch.0.
Since TBL_BASE[9:0] of this register is fixed at 0 regardless of the contents written, it is always set to 1,024-byte boundary address. The initial value of the register is 0x84000.

Base + 0xf0	Ch.7 auto-reload data area
Base + 0xe0	Ch.7 control table
Base + 0xd0	Ch.6 auto-reload data area
Base + 0xc0	Ch.6 control table
Base + 0xb0	Ch.5 auto-reload data area
Base + 0xa0	Ch.5 control table
Base + 0x90	Ch.4 auto-reload data area
Base + 0x80	Ch.4 control table
Base + 0x70	Ch.3 auto-reload data area
Base + 0x60	Ch.3 control table
Base + 0x50	Ch.2 auto-reload data area
Base + 0x40	Ch.2 control table
Base + 0x30	Ch.1 auto-reload data area
Base + 0x20	Ch.1 control table
Base + 0x10	Ch.0 auto-reload data area
Base	Ch.0 control table

Figure 14.7.1 Control Table Map

Note: The control table must be placed on DSTRAM, IVRAM (Area 3) or an external RAM. A0RAM/IRAM (Area 0) and BBRAM cannot be used to store control information.

DMAC Interrupt Enable Register (DMAC_IE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
DMAC Interrupt Enable Register (DMAC_IE)	**0x302108** (32 bits)	D31–8	**–**	reserved	–	–	–	0 when being read.
		D7	**DMAIE7**	DMAC Ch.7 interrupt enable	1 Enable 0 Disable	0	R/W	
		D6	**DMAIE6**	DMAC Ch.6 interrupt enable	1 Enable 0 Disable	0	R/W	
		D5	**DMAIE5**	DMAC Ch.5 interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**DMAIE4**	DMAC Ch.4 interrupt enable	1 Enable 0 Disable	0	R/W	
		D3	**DMAIE3**	DMAC Ch.3 interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**DMAIE2**	DMAC Ch.2 interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**DMAIE1**	DMAC Ch.1 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**DMAIE0**	DMAC Ch.0 interrupt enable	1 Enable 0 Disable	0	R/W	

D[31:8] Reserved

D[7:0] DMAIE*x*: DMAC Ch.*x* Interrupt Enable Bit

Enables or disables DMAC Ch.*x* interrupts.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Setting DMAIE*x* to 1 enables the output of DMAC Ch.*x* interrupt requests to the ITC. Interrupts from Ch.*x* will not be generated if DMAIE*x* is set to 0.

DMAC Trigger Select Register (DMAC_TRG_SEL)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
DMAC Trigger Select Register (DMAC_TRG_SEL)	0x30210c (32 bits)	D31–16	–	reserved	–		–	–	0 when being read.
		D15–14	TRG_SEL7 [1:0]	Ch.7 trigger select	TRG_SEL7[1:0]	Trigger source	0x0	R/W	
					0x3	ADC complete			
					0x2	USI Ch.2 Tx			
					0x1	USIL Tx			
					0x0	No hard trigger			
		D13–12	TRG_SEL6 [1:0]	Ch.6 trigger select	TRG_SEL6[1:0]	Trigger source	0x0	R/W	
					0x3	USB			
					0x2	USI Ch.2 Rx			
					0x1	USIL Rx			
					0x0	No hard trigger			
		D11–10	TRG_SEL5 [1:0]	Ch.5 trigger select	TRG_SEL5[1:0]	Trigger source	0x0	R/W	
					0x3	T16A6 A			
					0x2	SD_MMC Tx			
					0x1	USI Ch.1 Tx			
					0x0	No hard trigger			
		D9–8	TRG_SEL4 [1:0]	Ch.4 trigger select	TRG_SEL4[1:0]	Trigger source	0x0	R/W	
					0x3	T16A6 B			
					0x2	SD_MMC Rx			
					0x1	USI Ch.1 Rx			
					0x0	No hard trigger			
		D7–6	TRG_SEL3 [1:0]	Ch.3 trigger select	TRG_SEL3[1:0]	Trigger source	0x0	R/W	
					0x3	T16A6 A			
					0x2	I^2S input R			
					0x1	USI Ch.0 Tx			
					0x0	No hard trigger			
		D5–4	TRG_SEL2 [1:0]	Ch.2 trigger select	TRG_SEL2[1:0]	Trigger source	0x0	R/W	
					0x3	T16A6 B			
					0x2	I^2S input L			
					0x1	USI Ch.0 Rx			
					0x0	No hard trigger			
		D3–2	TRG_SEL1 [1:0]	Ch.1 trigger select	TRG_SEL1[1:0]	Trigger source	0x0	R/W	
					0x3	Port			
					0x2	T16P Ch.1			
					0x1	I^2S output R			
					0x0	No hard trigger			
		D1–0	TRG_SEL0 [1:0]	Ch.0 trigger select	TRG_SEL0[1:0]	Trigger source	0x0	R/W	
					0x3	ADC complete			
					0x2	T16P Ch.0			
					0x1	I^2S output L			
					0x0	No hard trigger			

D[31:16] Reserved

D[15:0] TRG_SEL*x*[1:0]: Ch.*x* Trigger Select Bits

Selects a trigger source for each DMAC channel.

Table 14.7.2 DMAC Trigger Source

Channel	Control bits	Setting	Trigger source	Channel priority
Ch.7	TRG_SEL7[1:0]	0x3	A/D converter (ADC10) conversion completion	Low ↑
		0x2	USI Ch.2 transmit buffer empty	
		0x1	USIL transmit buffer empty	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.6	TRG_SEL6[1:0]	0x3	USB interrupt	
		0x2	USI Ch.2 receive buffer full	
		0x1	USIL receive buffer full	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.5	TRG_SEL5[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture A *	
		0x2	SD/MMC interface (SD_MMC) FIFO half-empty	
		0x1	USI Ch.1 transmit buffer empty	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.4	TRG_SEL4[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture B *	
		0x2	SD/MMC interface (SD_MMC) FIFO half-full	
		0x1	USI Ch.1 receive buffer full	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.3	TRG_SEL3[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture A *	
		0x2	I^2S input (R)	
		0x1	USI Ch.0 transmit buffer empty	
		0x0	Hardware trigger disabled (software trigger only)	

Channel	Control bits	Setting	Trigger source	Channel priority
Ch.2	TRG_SEL2[1:0]	0x3	16-bit PWM timer (T16A6) Ch.*x* compare/capture B *	
		0x2	I^2S input (L)	
		0x1	USI Ch.0 receive buffer full	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.1	TRG_SEL1[1:0]	0x3	Port input interrupt	
		0x2	16-bit audio PWM timer (T16P) Ch.1 buffer empty	
		0x1	I^2S output (R)	
		0x0	Hardware trigger disabled (software trigger only)	
Ch.0	TRG_SEL0[1:0]	0x3	A/D converter (ADC10) conversion completion	
		0x2	16-bit audio PWM timer (T16P) Ch.0 buffer empty	
		0x1	I^2S output (L)	↓
		0x0	Hardware trigger disabled (software trigger only)	High

* Set the T16A6 channel for invoking the DMAC using DMASEL[1:0]/T16A_CTL*x* register. (Default: 0x0)

At initial reset, TRG_SEL*x*[1:0] in all channels are set to 0x0 (hardware trigger disabled). Note that software triggers are enabled regardless of the trigger source selected.

DMAC Trigger Flag Register (DMAC_TRG_FLG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
DMAC Trigger Flag Register (DMAC_TRG_FLG)	**0x302110** (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**TRG7**	Ch.7 software trigger/trigger status	1	(W) Soft trigger	0	(W) Ignored	0	R/W	
		D6	**TRG6**	Ch.6 software trigger/trigger status					0	R/W	
		D5	**TRG5**	Ch.5 software trigger/trigger status					0	R/W	
		D4	**TRG4**	Ch.4 software trigger/trigger status					0	R/W	
		D3	**TRG3**	Ch.3 software trigger/trigger status		(R) Triggered		(R) Not triggered	0	R/W	
		D2	**TRG2**	Ch.2 software trigger/trigger status					0	R/W	
		D1	**TRG1**	Ch.1 software trigger/trigger status					0	R/W	
		D0	**TRG0**	Ch.0 software trigger/trigger status					0	R/W	

D[31:8] Reserved

D[7:0] TRG*x*: Ch.*x* Software Trigger/Trigger Status Bit

Invokes a DMA of the specified channel by software trigger. Also indicates trigger status in respective channels, including hardware trigger.

1 (W): Software trigger
0 (W): Ignored
1 (R): Triggered
0 (R): Not triggered (default)

To use software trigger to start a Ch.*x* DMA transfer, write 1 to TRG*x*. In the case of a hardware trigger, the DMA transfer starts after TRG*x* is set to 1.

Among DMAC channels, Ch.0 is assigned the highest priority, which goes down in the ascending order of channels numbers. Therefore, when there are multiple settings of TRG*x*, channels with lower channel numbers are processed before the higher-number channels. Lower-priority channels are kept pending until all DMA transfers in higher-priority channels are completed, and TRG*x* also retains 1. The above applies to cases where another trigger is generated during a DMA transfer. That is, regardless of the order of trigger generation, a DMA request from the highest-priority channel is accepted as soon as the current DMA transfer is completed or suspended.

After the DMAC accepts a trigger, the DMA transfer of the channel starts. At the same time TRG*x* is cleared, allowing the channel to be re-triggered.

Note that acceptance of the trigger does not start a DMA transfer if CHEN (D3/1st word) in control information is set to 0.

If DMAON*x*/DMAC_CH_EN register is set to 0 (forced termination), TRG*x* that has been set is cleared and the pending DMA request is canceled.

DMAC End-of-Transfer Flag Register (DMAC_END_FLG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
DMAC End-of-Transfer Flag Register (DMAC_END_FLG)	0x302114 (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**ENDF7**	Ch.7 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	Reset by writing 1.
		D6	**ENDF6**	Ch.6 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D5	**ENDF5**	Ch.5 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D4	**ENDF4**	Ch.4 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D3	**ENDF3**	Ch.3 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D2	**ENDF2**	Ch.2 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D1	**ENDF1**	Ch.1 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D0	**ENDF0**	Ch.0 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	

D[31:8] Reserved

D[7:0] ENDF*x*: Ch.*x* End-of-Transfer Flag Bit

Indicates the channel that has finished transfers.

1 (R): Finished
0 (R): Not finished (default)
1 (W): Flag is reset
0 (W): Ignored

If the transfer counter in DMA transfer reaches 0, the DMAC sets ENDF*x* indicating that transfers are finished. At the same time, an interrupt request is output to the ITC if DMAIE*x*/DMAC_IE is set to 1 (interrupt enabled).

Read this register in the DMAC interrupt handler routine and check which channel has finished transfers. Also, in preparation for next interrupts, write 1 to ENDF*x* for resetting it.

In a channel with DMAIE*x* is set to 0 (interrupt disabled), an interrupt is not generated even if ENDF*x* is set.

DMAC Running Status Register (DMAC_RUN_STAT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
DMAC Running Status Register (DMAC_RUN_STAT)	0x302118 (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**RUN7**	Ch.7 running status	1	Running	0	Idle/paused	0	R	
		D6	**RUN6**	Ch.6 running status	1	Running	0	Idle/paused	0	R	
		D5	**RUN5**	Ch.5 running status	1	Running	0	Idle/paused	0	R	
		D4	**RUN4**	Ch.4 running status	1	Running	0	Idle/paused	0	R	
		D3	**RUN3**	Ch.3 running status	1	Running	0	Idle/paused	0	R	
		D2	**RUN2**	Ch.2 running status	1	Running	0	Idle/paused	0	R	
		D1	**RUN1**	Ch.1 running status	1	Running	0	Idle/paused	0	R	
		D0	**RUN0**	Ch.0 running status	1	Running	0	Idle/paused	0	R	

D[31:8] Reserved

D[7:0] RUN*x*: Ch.*x* Running Status Bit

Indicates whether the channel is performing a DMA transfer or not.

1 (R): Performing a DMA transfer
0 (R): Idle/paused (default)

RUN*x* is set to 1 when DMAC Ch.*x* starts a DMA transfer and reset to 0 upon completion of the transfer operation. Also this bit reverts to 0 when the transfer is suspended due to a high-priority DMA request.

When modifying control information after a data transfer or forced termination, check this bit to ensure that the transfer operation is actually completed.

DMAC Pause Status Register (DMAC_PAUSE_STAT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
DMAC Pause Status Register (DMAC_ PAUSE_STAT)	0x30211c (32 bits)	D31–8	–	reserved	–	–	–	0 when being read.
		D7	**PAUSE7**	Ch.7 pause status	1 Paused 0 Not paused	0	R	
		D6	**PAUSE6**	Ch.6 pause status	1 Paused 0 Not paused	0	R	
		D5	**PAUSE5**	Ch.5 pause status	1 Paused 0 Not paused	0	R	
		D4	**PAUSE4**	Ch.4 pause status	1 Paused 0 Not paused	0	R	
		D3	**PAUSE3**	Ch.3 pause status	1 Paused 0 Not paused	0	R	
		D2	**PAUSE2**	Ch.2 pause status	1 Paused 0 Not paused	0	R	
		D1	**PAUSE1**	Ch.1 pause status	1 Paused 0 Not paused	0	R	
		D0	**PAUSE0**	Ch.0 pause status	1 Paused 0 Not paused	0	R	

D[31:8] Reserved

D[7:0] PAUSE*x*: Ch.*x* Paused Status Bit

Indicates whether the successive transfer operation is suspended due to a high-priority DMA transfer or not.

1 (R): Suspended

0 (R): Status other than suspension (default)

When a DMA request is generated that has higher priority than that of the channel in operation, the channel performing a transfer saves control information required for resuming transfers (such as the current transfer count and the transfer source and destination addresses) as soon as the current data transfer is completed and then suspends transfers. In this case, PAUSE*x* is also set to 1, indicating that the channel has suspended a transfer. After that, the high-priority DMA transfer is executed. After the transfer is completed, suspended DMA transfers are resumed. At this time, the DMAC checks PAUSE*x* and TRG*x*/DMAC_TRG_FLG register, and processes the channels with their bits set, starting with one with the highest-priority (with the channel with the lowest number).

When the DMAC resumes DMA transfers that have been suspended, PAUSE*x* is cleared.

15 Fine Mode 8-bit Programmable Timers (T8F)

15.1 T8F Module Overview

The S1C33L27 incorporates a six-channel fine mode 8-bit programmable timer module (T8F).
The features of T8F are listed below.

- 8-bit presettable down counter with an 8-bit reload data register for setting the preset value
- The count clock is selectable from 15 clocks output from the prescaler.
- Generates the USI, USIL, and UART operating clocks (transfer clock source), and an A/D trigger signal from the counter underflow signals.
- Generates an underflow interrupt signal to the interrupt controller (ITC).
- Any desired time intervals and serial transfer rates can be programmed by selecting an appropriate count clock and preset value.
- Fine mode is provided to minimize transfer rate errors.

Figure 15.1.1 shows the T8F configuration.

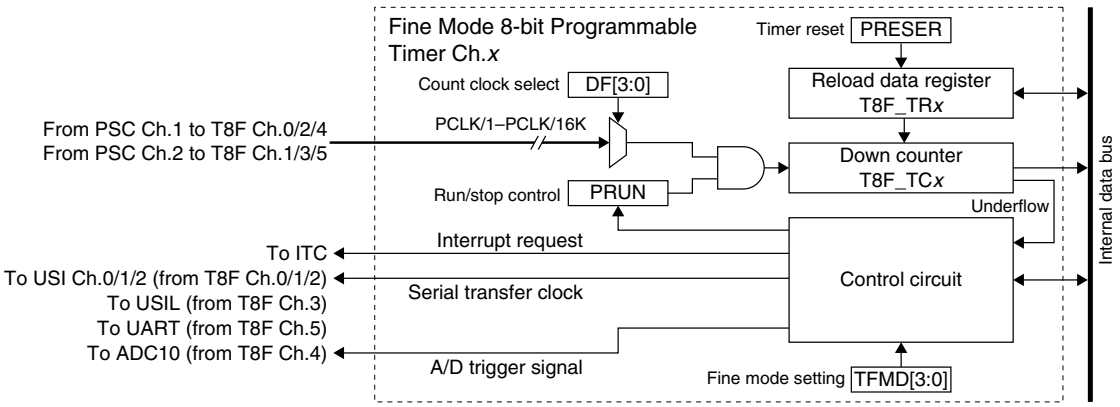


Figure 15.1.1 T8F Configuration (one channel)

T8F consists of an 8-bit presettable down counter and an 8-bit reload data register holding 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 signals are used to generate interrupts, serial interface (USI, USIL, and UART) clocks, and an A/D trigger signal. The underflow cycle can be programmed by selecting the prescaler clock and reload data, enabling the application program to obtain time intervals and serial transfer speeds as required. Fine mode provides a function that minimizes transfer rate errors.

Note: Six channels of T8F module have the same functions except for the control register addresses. The description in this chapter applies to all channels. The '*x*' in the register name refers to the channel number (0 to 5).
Example: T8F_CTL*x* register
Ch.0: T8F_CTL0 register
Ch.1: T8F_CTL1 register
Ch.2: T8F_CTL2 register
:
Ch.5: T8F_CTL5 register

15.2 Count Clock

The count clock is selected by DF[3:0]/T8F_CLK*x* register from the 15 types generated by the prescaler dividing the PCLK clock into 1/1 to 1/16K.

* T8F Ch.0, 2, and 4 use the PSC Ch.1 output clocks. T8F Ch.1, 3, and 5 use the PSC Ch.2 output clocks.

Table 15.2.1 Count Clock (PCLK Division Ratio) Selection

DF[3:0]	Division ratio	DF[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Notes: • The prescaler must run before T8F can operate.
• Make sure the counter is halted before setting the count clock.

For detailed information on the prescaler control, see the "Prescaler (PSC)" chapter.

15.3 Count Mode

T8F features two count modes: repeat mode and one-shot mode. These modes are selected using TRMD/T8F_CTL*x* register.

Repeat mode (TRMD = 0, default)

Setting TRMD to 0 sets T8F to repeat mode.
In this mode, once the count starts, the timer continues running until stopped by the application program. When the counter underflows, the timer presets the reload data register value into the counter and continues the count. Thus, the timer periodically outputs an underflow pulse. T8F should be set to this mode to generate periodic interrupts or A/D triggers at desired intervals or to generate a serial transfer clock.

One-shot mode (TRMD = 1)

Setting TRMD to 1 sets T8F to one-shot mode.
In this mode, the timer stops automatically as soon as the counter underflows. This means only one interrupt can be generated after the timer starts. Note that the timer presets the reload data register value to the counter, then stops after an underflow has occurred. T8F should be set to this mode to set a specific wait time.

Note: Make sure the counter is halted before setting the count mode.

15.4 Reload Data Register and Underflow Cycle

The reload data register T8F_TR*x* is used to set the initial value for the down counter.
The initial counter value set in the reload data register is preset to the down counter if T8F is reset or the counter underflows. If T8F is started after resetting, the timer counts down from the reload value (initial value). This means that the reload value and the input clock frequency determine the time elapsed from the point at which the timer starts until the underflow occurs (or between underflows). The time determined is used to obtain the specified wait time, the intervals between periodic interrupts or A/D triggers, and the programmable serial interface transfer clock.

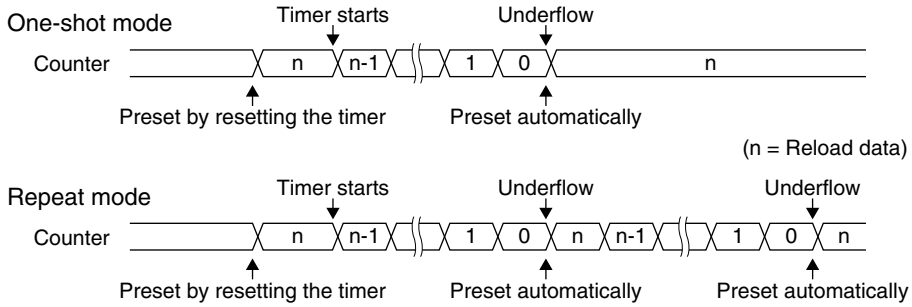


Figure 15.4.1 Preset Timing

The underflow cycle can be calculated as follows:

$$\text{Underflow interval} = \frac{TR + 1}{clk\_in}\ [s] \qquad \text{Underflow cycle} = \frac{clk\_in}{TR + 1}\ [Hz]$$

clk_in: Count clock (prescaler output clock) frequency [Hz]
TR: Reload data (0–255)

15.5 Timer Reset

T8F is reset by writing 1 to PRESER/T8F_CTL*x* register. The reload data is preset and the counter is initialized.

15.6 RUN/STOP Control

Make the following settings before starting T8F.
(1) Select the count clock (prescaler output clock). See Section 15.2.
(2) Set the count mode (one-shot or repeat). See Section 15.3.
(3) Calculate the initial counter value and set it to the reload data register. See Section 15.4.
(4) Reset the timer to preset the counter to the initial value. See Section 15.5.
(5) When using timer interrupts, set the interrupt level and enable interrupts for the relevant timer channel. See Section 15.9.

To start T8F, write 1 to PRUN/T8F_CTL*x* register.
The timer starts counting down from the initial value or from the current counter value if no initial value was preset. When the counter underflows, the timer outputs an underflow pulse and presets the counter to the initial value. An interrupt request is sent simultaneously to the interrupt controller (ITC).
If one-shot mode is set, the timer stops counting.
If repeat mode is set, the timer continues counting from the reloaded initial value.

Write 0 to PRUN to stop T8F via the application program. The counter stops counting and retains the current counter value until either the timer is reset or restarted. To restart the count from the initial value, the timer should be reset before writing 1 to PRUN.
When the timer is reset during running, the timer loads the reload register value to the counter and continues counting.

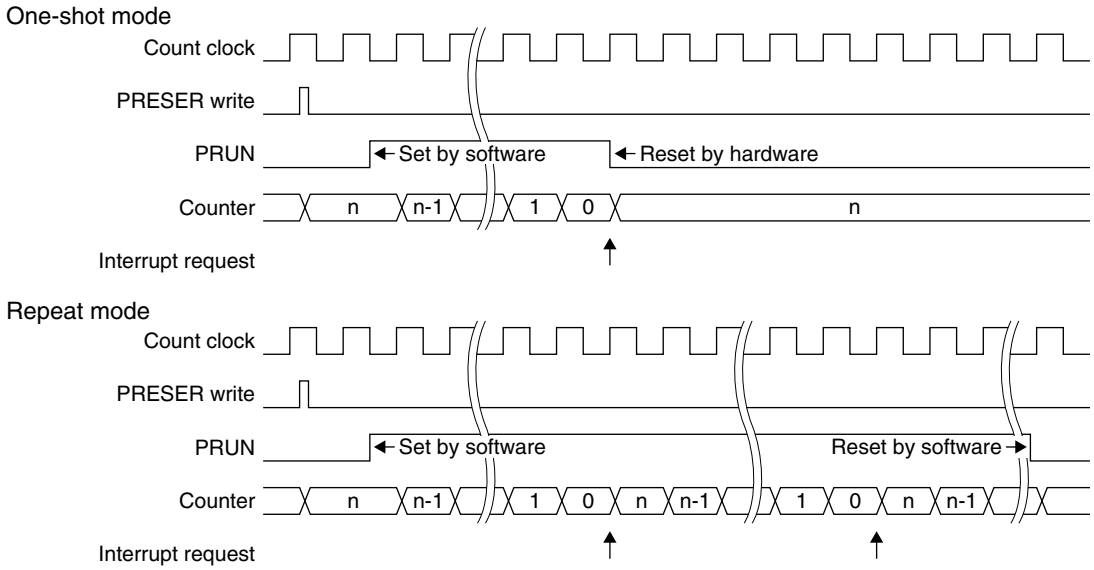


Figure 15.6.1 Count Operation

15.7 T8F Output Signals

T8F outputs underflow pulses when the counter underflows.
These pulses are used for timer interrupt requests.
These pulses are also used to generate the serial transfer clock for the internal serial interface or the A/D trigger signal.
The clock generated is sent to the internal peripheral module, as shown below.
T8F Ch.0 output clock → USI Ch.0
T8F Ch.1 output clock → USI Ch.1
T8F Ch.2 output clock → USI Ch.2
T8F Ch.3 output clock → USIL
T8F Ch.4 output clock → A/D converter
T8F Ch.5 output clock → UART

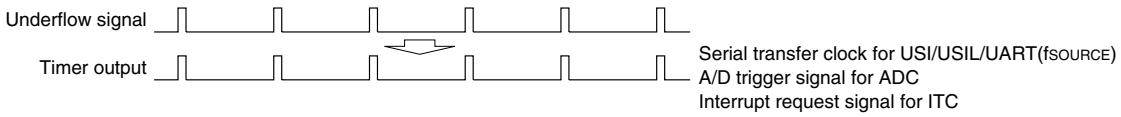


Figure 15.7.1 Timer Output Clock

15.8 Fine Mode

Fine mode provides a function that minimizes transfer rate errors.
T8F can output a programmable clock signal for use as the USI serial transfer clock. The timer output clock can be set to the required frequency by selecting the appropriate prescaler output clock and reload data. Note that errors may occur, depending on the transfer rate. Fine mode extends the output clock cycle by delaying the underflow pulse from the counter. This delay can be specified with the TFMD[3:0]/T8F_CTL*x* register.
TFMD[3:0] specifies the delay pattern to be inserted into a 16 underflow period. Inserting one delay extends the output clock cycle by one count clock cycle. This setting delays the interrupt timing in the same way.

Table 15.8.1 Delay Patterns Specified by TFMD[3:0]

TFMD[3:0]	Underflow number															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0x0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
0x1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	D
0x2	–	–	–	–	–	–	–	D	–	–	–	–	–	–	–	D
0x3	–	–	–	–	–	–	–	D	–	–	–	D	–	–	–	D
0x4	–	–	–	D	–	–	–	D	–	–	–	D	–	–	–	D
0x5	–	–	–	D	–	–	–	D	–	–	–	D	–	D	–	D
0x6	–	–	–	D	–	D	–	D	–	–	–	D	–	D	–	D
0x7	–	–	–	D	–	D	–	D	–	D	–	D	–	D	–	D
0x8	–	D	–	D	–	D	–	D	–	D	–	D	–	D	–	D
0x9	–	D	–	D	–	D	–	D	–	D	–	D	–	D	D	D
0xa	–	D	–	D	–	D	D	D	–	D	–	D	–	D	D	D
0xb	–	D	–	D	–	D	D	D	–	D	D	D	–	D	D	D
0xc	–	D	D	D	–	D	D	D	–	D	D	D	–	D	D	D
0xd	–	D	D	D	–	D	D	D	–	D	D	D	D	D	D	D
0xe	–	D	D	D	D	D	D	D	–	D	D	D	D	D	D	D
0xf	–	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

D: Indicates the insertion of a delay cycle.

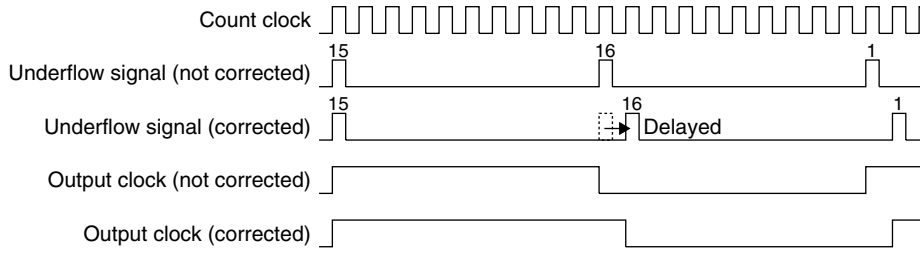


Figure 15.8.1 Delay Cycle Insertion in Fine Mode

At initial reset, TFMD[3:0] is set to 0x0, preventing insertion of delay cycles.

15.9 T8F Interrupts

T8F outputs an interrupt request to the interrupt controller (ITC) when the counter underflows.

Timer underflow interrupt

When the counter underflows, the interrupt flag T8FIF/T8F_INT*x* register, which is provided for each channel in the T8F module, is set to 1. At the same time, an interrupt request is sent to the ITC if T8FIE/T8F_INT*x* register has been set to 1 (interrupt enabled). An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied.

If T8FIE is set to 0 (interrupt disabled, default), no interrupt request will be sent to the ITC.

For specific information on interrupt processing, see the "Interrupt Controller (ITC)" chapter.

Notes: • The T8F module interrupt flag T8FIF must be reset in the interrupt handler routine after a T8F interrupt has occurred to prevent recurring interrupts.

• Reset T8FIF before enabling T8F interrupts with T8FIE to prevent occurrence of unwanted interrupt. T8FIF is reset by writing 1.

15.10 Control Register Details

Table 15.10.1 List of T8F Registers

Address	Register name		Function
0x301100	T8F_CLK0	T8F Ch.0 Input Clock Select Register	Select prescaler output clock
0x301102	T8F_TR0	T8F Ch.0 Reload Data Register	Set reload data
0x301104	T8F_TC0	T8F Ch.0 Counter Data Register	Counter data
0x301106	T8F_CTL0	T8F Ch.0 Control Register	Set timer mode and start/stop timer
0x301108	T8F_INT0	T8F Ch.0 Interrupt Control Register	Control interrupt
0x301110	T8F_CLK1	T8F Ch.1 Input Clock Select Register	Select prescaler output clock
0x301112	T8F_TR1	T8F Ch.1 Reload Data Register	Set reload data
0x301114	T8F_TC1	T8F Ch.1 Counter Data Register	Counter data
0x301116	T8F_CTL1	T8F Ch.1 Control Register	Set timer mode and start/stop timer
0x301118	T8F_INT1	T8F Ch.1 Interrupt Control Register	Control interrupt
0x301120	T8F_CLK2	T8F Ch.2 Input Clock Select Register	Select prescaler output clock
0x301122	T8F_TR2	T8F Ch.2 Reload Data Register	Set reload data
0x301124	T8F_TC2	T8F Ch.2 Counter Data Register	Counter data
0x301126	T8F_CTL2	T8F Ch.2 Control Register	Set timer mode and start/stop timer
0x301128	T8F_INT2	T8F Ch.2 Interrupt Control Register	Control interrupt
0x301130	T8F_CLK3	T8F Ch.3 Input Clock Select Register	Select prescaler output clock
0x301132	T8F_TR3	T8F Ch.3 Reload Data Register	Set reload data
0x301134	T8F_TC3	T8F Ch.3 Counter Data Register	Counter data
0x301136	T8F_CTL3	T8F Ch.3 Control Register	Set timer mode and start/stop timer
0x301138	T8F_INT3	T8F Ch.3 Interrupt Control Register	Control interrupt
0x301140	T8F_CLK4	T8F Ch.4 Input Clock Select Register	Select prescaler output clock
0x301142	T8F_TR4	T8F Ch.4 Reload Data Register	Set reload data
0x301144	T8F_TC4	T8F Ch.4 Counter Data Register	Counter data
0x301146	T8F_CTL4	T8F Ch.4 Control Register	Set timer mode and start/stop timer
0x301148	T8F_INT4	T8F Ch.4 Interrupt Control Register	Control interrupt
0x301150	T8F_CLK5	T8F Ch.5 Input Clock Select Register	Select prescaler output clock
0x301152	T8F_TR5	T8F Ch.5 Reload Data Register	Set reload data
0x301154	T8F_TC5	T8F Ch.5 Counter Data Register	Counter data
0x301156	T8F_CTL5	T8F Ch.5 Control Register	Set timer mode and start/stop timer
0x301158	T8F_INT5	T8F Ch.5 Interrupt Control Register	Control interrupt

The T8F registers are described in detail below.

Note: When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

T8F Ch.*x* Input Clock Select Registers (T8F_CLK*x*)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
T8F Ch.*x* Input Clock Select Register (T8F_CLK*x*)	**0x301100** **0x301110** **0x301120** **0x301130** **0x301140** **0x301150** (16 bits)	D15–4	**–**	reserved	–		–	–	0 when being read.
		D3–0	**DF[3:0]**	T8F clock division ratio select (Prescaler output clock)	DF[3:0]	Division ratio	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
					0xf	reserved			
					0xe	1/16384			
					0xd	1/8192			
					0xc	1/4096			
					0xb	1/2048			
					0xa	1/1024			
					0x9	1/512			
					0x8	1/256			
					0x7	1/128			
					0x6	1/64			
					0x5	1/32			
					0x4	1/16			
					0x3	1/8			
					0x2	1/4			
					0x1	1/2			
					0x0	1/1			

D[15:4] Reserved

D[3:0] DF[3:0]: T8F Input Clock Select Bits

Selects the T8F count clock from the 15 different prescaler output clocks.

Table 15.10.2 Count Clock (PCLK Division Ratio) Selection

DF[3:0]	Division ratio	DF[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Note: Make sure the counter is halted before setting the count clock.

T8F Ch.*x* Reload Data Registers (T8F_TR*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.*x*	**0x301102**	D15-8	**–**	reserved	–	–	–	0 when being read.
Reload Data	**0x301112**	D7–0	**TR[7:0]**	T8F reload data	0x0 to 0xff	0x0	R/W	
Register	**0x301122**			TR7 = MSB				
(T8F_TR*x*)	**0x301132**			TR0 = LSB				
	0x301142							
	0x301152							
	(16 bits)							

D[15:8] Reserved

D[7:0] TR[7:0]: T8F Reload Data Bits

Sets the counter initial value. (Default: 0x0)

The reload data set in this register is preset to the counter when the timer is reset or the counter underflows. If the timer is started after resetting, it counts down from the reload value (initial value). This means that the reload value and the input clock frequency determine the time elapsed from the point at which the timer starts until the underflow occurs (or between underflows). The time determined is used to obtain the desired wait time, the intervals between periodic interrupts or A/D triggers, and the programmable serial interface transfer clock.

T8F Ch.*x* Counter Data Registers (T8F_TC*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.*x*	**0x301104**	D15-8	**–**	reserved	–	–	–	0 when being read.
Counter Data	**0x301114**	D7–0	**TC[7:0]**	T8F counter data	0x0 to 0xff	0xff	R	
Register	**0x301124**			TC7 = MSB				
(T8F_TC*x*)	**0x301134**			TC0 = LSB				
	0x301144							
	0x301154							
	(16 bits)							

D[15:8] Reserved

D[7:0] TC[7:0]: T8F Counter Data Bits

The counter data can be read out. (Default: 0xff)

This register is read-only and cannot be written to.

T8F Ch.*x* Control Registers (T8F_CTL*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.*x*	**0x301106**	D15–12	**–**	reserved	–	–	–	0 when being read.
Control Register	**0x301116**	D11–8	**TFMD[3:0]**	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
(T8F_CTL*x*)	**0x301126**							
	0x301136							
	0x301146							
	0x301156	D7–5	**–**	reserved	–	–	–	0 when being read.
	(16 bits)	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[15:12] Reserved

D[11:8] TFMD[3:0]: Fine Mode Setup Bits

Corrects the transfer rate error. (Default: 0x0)

TFMD[3:0] specifies the delay pattern to be inserted into a 16 underflow period. Inserting one delay extends the output clock cycle by one count clock cycle. This setting delays the interrupt timing in the same way.

Table 15.10.3 Delay Patterns Specified by TFMD[3:0]

TFMD[3:0]	Underflow number															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0x0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
0x1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	D
0x2	–	–	–	–	–	–	–	D	–	–	–	–	–	–	–	D
0x3	–	–	–	–	–	–	–	D	–	–	–	D	–	–	–	D
0x4	–	–	–	D	–	–	–	D	–	–	–	D	–	–	–	D
0x5	–	–	–	D	–	–	–	D	–	–	–	D	–	D	–	D
0x6	–	–	–	D	–	D	–	D	–	–	–	D	–	D	–	D
0x7	–	–	–	D	–	D	–	D	–	D	–	D	–	D	–	D
0x8	–	D	–	D	–	D	–	D	–	D	–	D	–	D	–	D
0x9	–	D	–	D	–	D	–	D	–	D	–	D	–	D	D	D
0xa	–	D	–	D	–	D	D	D	–	D	–	D	–	D	D	D
0xb	–	D	–	D	–	D	D	D	–	D	D	D	–	D	D	D
0xc	–	D	D	D	–	D	D	D	–	D	D	D	–	D	D	D
0xd	–	D	D	D	–	D	D	D	–	D	D	D	D	D	D	D
0xe	–	D	D	D	D	D	D	D	–	D	D	D	D	D	D	D
0xf	–	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

D: Indicates the insertion of a delay cycle.

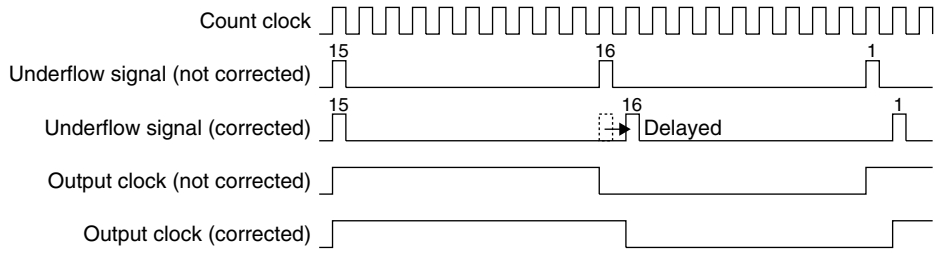


Figure 15.10.1 Delay Cycle Insertion in Fine Mode

D[7:5] Reserved

D4 TRMD: Count Mode Select Bit

Selects the T8F count mode.

1 (R/W): One-shot mode

0 (R/W): Repeat mode (default)

Setting TRMD to 0 sets T8F to repeat mode. In this mode, once the count starts, the timer continues to run until stopped by the application program. When the counter underflows, the timer presets the counter to the reload data register value and continues the count. Thus, the timer periodically outputs an underflow pulse. Set T8F to this mode to generate periodic interrupts or A/D triggers at desired intervals or to generate a serial transfer clock.

Setting TRMD to 1 sets T8F to one-shot mode. In this mode, the timer stops automatically as soon as the counter underflows. This means only one interrupt can be generated after the timer starts. Note that the timer presets the counter to the reload data register value, then stops when an underflow occurs. Set T8F to this mode to set a specific wait time.

Note: Make sure the counter is halted before setting the count mode.

D[3:2] Reserved

D1 PRESER: Timer Reset Bit

Resets the timer.

1 (W): Reset

0 (W): Ignored

0 (R): Always 0 when read (default)

Writing 1 to this bit presets the counter to the reload data value.

D0 PRUN: Timer Run/Stop Control Bit

Controls the timer RUN/STOP.

1 (R/W): Run

0 (R/W): Stop (default)

The timer starts counting when PRUN is written as 1 and stops when written as 0. When the timer is stopped, the counter data is retained until reset or until the next RUN state.

T8F Ch.*x* Interrupt Control Registers (T8F_INT*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.*x* Interrupt Control Register (T8F_INT*x*)	0x301108 0x301118 0x301128 0x301138 0x301148 0x301158 (16 bits)	D15–9	–	reserved	–	–	–	0 when being read.
		D8	T8FIE	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	–	reserved	–	–	–	0 when being read.
		D0	T8FIF	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

D[15:9] Reserved

D8 T8FIE: T8F Interrupt Enable Bit

Enables or disables interrupts caused by counter underflows for each channel.

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

Setting T8FIE to 1 enables T8F interrupt requests to the ITC; setting to 0 disables interrupts.

D[7:1] Reserved

D0 T8FIF: T8F Interrupt Flag Bit

Indicates whether the cause of counter underflow interrupt has occurred or not.

1 (R): Cause of interrupt has occurred

0 (R): No cause of interrupt has occurred (default)

1 (W): Flag is reset

0 (W): Ignored

T8FIF is the T8F interrupt flag that is set to 1 when the counter underflows.

T8FIF is reset by writing 1.

16 16-bit PWM Timer (T16A6)

16.1 T16A6 Module Overview

The S1C33L27 includes a 16-bit PWM timer (T16A6) module with four timer channels.
The features of T16A6 are listed below.

- 16-bit up counter with a comparator and capture unit
- The count clock is selectable from 15 clocks output from the prescaler.
- Supports event counter function using an external clock.
- Includes a comparator that compares the counter value with two specified comparison values to generate interrupts and various output waveform including a PWM waveform.
- Includes a capture unit that captures counter values using two external trigger signals and generates interrupts.

Figure 16.1.1 shows the T16A6 configuration.

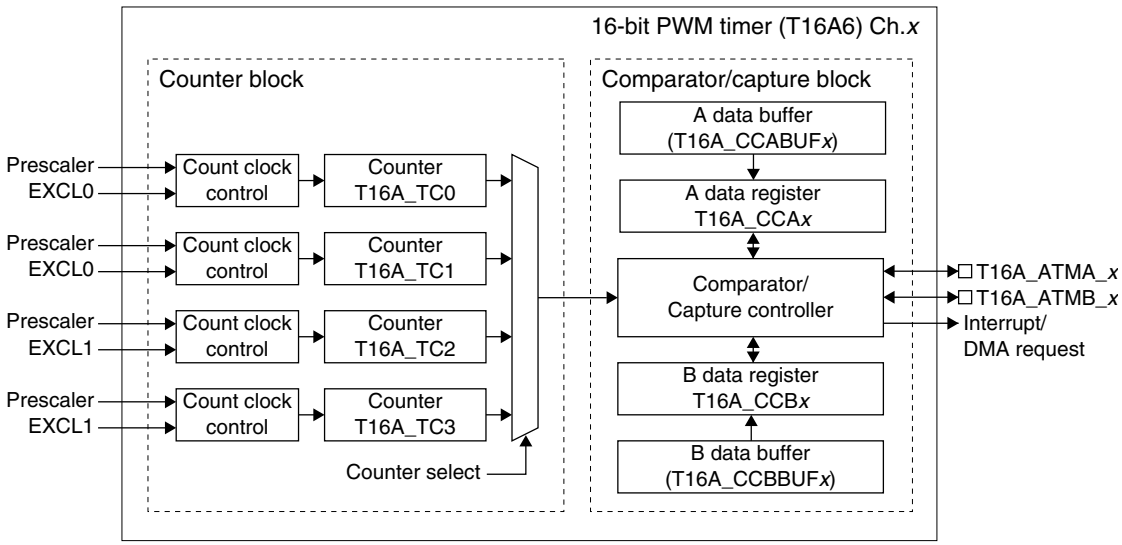


Figure 16.1.1 T16A6 Configuration (one channel)

The T16A6 module consists of a counter block and a comparator/capture block.

Counter block

The counter block includes a 16-bit up-counter that operates with a prescaler output clock, or the external count clock input from outside the IC. The 16-bit PWM timer (T16A6) allows software to run and stop the counter, and to reset the counter value (cleared to 0) as well as selection of the count clock. The counter can also be reset by the compare B signal output from the comparator/capture block.

Comparator/capture block

The comparator/capture block includes two systems (units A and B) of comparators that compare between the counter value and the specified comparison value and capture circuits that capture the counter value by an external trigger signal. Note, however, that the comparator and capture functions cannot be used at the same time in each system. One of the two functions must be selected by the software switch.

When using the comparator function, set the value(s) to be compared with the counter value to the compare A and/or compare B registers. When the counter reaches the value set in the compare A or compare B register, the comparator asserts the compare A or compare B signal. These signals can generate interrupts. Also the signals control the cycle time and duty ratio of the timer output signal allowing the timer to output a PWM or other waveform. In addition to these functions, the compare B signal is used to reset the counter.

Comparison data can be read or written directly from/to the compare A and compare B registers. The compare buffers are separately provided to load data to the compare A and compare B registers automatically by the compare B signal. Software can select which of the compare register and buffer the comparison values are written to. Note that enabling the compare buffers disables software to write directly to the compare A and compare B registers.

When the capture function is enabled, the compare A and compare B registers are used as the capture A and capture B registers, respectively. The capture A and capture B circuits can input a trigger signal individually, and the counter value is loaded to the respective capture register at the selected edge of the trigger signal.
The capturing operation can generate an interrupt, this make it possible to read the captured data in the interrupt handler routine. Also an overwrite interrupt can be generated for the error handling when the counter value is captured before reading the previous captured data.

Note: All channels of the T16A6 module has the same functions except for the control register addresses. The description in this section applies to all channels of the T16A6 module unless otherwise specified. Letter '*x*' in the register name refers to a channel number (0 to 3).
Example: T16A_CTL*x* register
Ch.0: T16A_CTL0 register
Ch.1: T16A_CTL1 register
Ch.2: T16A_CTL2 register
Ch.3: T16A_CTL3 register

16.2 T16A6 Input/Output Pins

Table 16.2.1 lists the input/output pins for the T16A6 module.

Table 16.2.1 List of T16A6 Pins

Pin name	I/O	Qty	Function
EXCL0 (Ch.0/1) EXCL1 (Ch.2/3)	I	2	T16A6 external clock input pin Inputs an external clock for the event counter function. The EXCL0 pin can also be used as the T16P Ch.0 external clock input pin. The EXCL1 pin can also be used as the T16P Ch.1 external clock input pin.
T16A_ATMA_0 (Ch.0) T16A_ATMA_1 (Ch.1) T16A_ATMA_2 (Ch.2) T16A_ATMA_3 (Ch.3)	I/O	4	T16A6 system A input/output pin Outputs timer generating signal in comparator mode. Inputs a counter-capture trigger signal in capture mode.
T16A_ATMB_0 (Ch.0) T16A_ATMB_1 (Ch.1) T16A_ATMB_2 (Ch.2) T16A_ATMB_3 (Ch.3)	I/O	4	T16A6 system B input/output pin Outputs timer generating signal in comparator mode. Inputs a counter-capture trigger signal in capture mode.

The T16A6 input/output pins (EXCL0/1, T16A_ATMA_*x*, T16A_ATMB_*x*) are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as T16A6 input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

16.3 Count Clock

The count clock is selected by CLKS[3:0]/T16A_CTL*x* register from the 15 types generated by the prescaler (PSC0) dividing the TCLK clock (SYSCLK, OSC3, or OSC1 *) into 1/1 to 1/16K and an external clock.

* T16A6 uses TCLK output from the CMU as the count clock source. TCLK can be selected from SYSCLK, OSC3, and OSC1 using a CMU register. For more information on the clock source selection, see the "Clock Management Unit (CMU)" chapter.

Table 16.3.1 Count Clock (TCLK Division Ratio) Selection

CLKS[3:0]	Division ratio	CLKS[3:0]	Division ratio
0xf	External clock	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Notes: • Make sure the counter is halted before setting the count clock.

• When using an external clock, the external clock cycle must be at least two CPU operating clock cycles.

For controlling the prescaler (PSC0), refer to the "Prescaler (PSC)" chapter.

16.4 T16A6 Operating Modes

T16A6 provides some operating modes to support various usages. This section describes the functions of each operating mode and how to enter the mode.

16.4.1 Comparator Mode and Capture Mode

The T16A_CCA*x* and T16A_CCB*x* registers that are embedded in the comparator/capture block can be set to comparator mode or capture mode, individually. The T16A_CCA*x* register mode is selected using CCAMD/T16A_CCCTL*x* register and the T16A_CCB*x* register mode is selected using CCBMD/T16A_CCCTL*x* register.

Comparator mode (CCAMD/CCBMD = 0, default)

The comparator mode compares the counter value and the comparison value set by software. It generates an interrupt and toggles the timer output signal level when the values are matched. The T16A_CCA*x* and T16A_CCB*x* registers function as the compare A and compare B registers that are used for loading comparison values in this mode.

The counter channel (Ch.0 to Ch.3) to be used can be selected using T16SEL[1:0]/T16A_CTL*x* register. This selection enables the two or more channels output compare A and compare B signals in sync with one 16-bit counter.

When the counter reaches the value set in the compare A register during counting, the comparator asserts the compare A signal. At the same time the compare A interrupt flag is set and an interrupt signal is output to the ITC if the interrupt has been enabled.

When the counter reaches the value set in the compare B register, the comparator asserts the compare B signal. At the same time the compare B interrupt flag is set and an interrupt signal is output to the ITC if the interrupt is enabled. Furthermore, the counter is reset to 0.

Note: The compare A data register (T16A_CCA*x*) and compare B data register (T16A_CCB*x*) must be set to four or more. If a value less than four is set, no cause of interrupt will occur.

The compare A and compare B signals are also used to generate a timer output waveform. See Section 16.6, "Timer Output Control," for more information.

To generate PWM waveform, the T16A_CCA*x* and T16A_CCB*x* registers must be both placed into comparator mode.

Compare buffers

Comparison data can be read or written directly from/to the compare registers. Comparison data for system A or B can also be written to the compare buffer so that it will be loaded to the compare A or compare B register by the compare B signal. The CBUFEN/T16A_CTL*x* register is used to select whether comparison data is written to the compare register or buffer.

Setting CBUFEN to 0 (default) selects the compare registers. Setting it to 1 selects the compare buffers. Although the T16A_CCA*x* and T16A_CCB*x* registers are used to write compare data even if CBUFEN = 1, compare buffers will be accessed. Note that compare data is always read from the compare registers regardless of whether the compare buffers are enabled or not.

Note: After an initial reset, the compare A and compare B registers are initialized to 0x0, so the compare A and compare B interrupts will occur and the corresponding output signals will be changed immediately after starting T16A6. In order to avoid this, disable the compare buffers and updated the compare A and B registers via software before running T16A6.

Capture mode (CCAMD/CCBMD = 1)

The capture mode captures the counter value when an external event such as a key entry occurs (at the specified edge of the external input signal). In this mode, the T16A_CCA*x* and/or T16A_CCB*x* registers function as the capture A and/or capture B registers for loading the captured data. To input a counter capture trigger signal, the capture A circuit uses the T16A_ATMA_*x* pin and the capture B circuit uses the T16A_ATMB_*x* pin. The T16A_ATMA_*x* and T16A_ATMB_*x* pins are shared with the timer outputs. They are configured for input when the system A or B is set to capture mode.

The counter channel (Ch.0 to Ch.3) to be used can be selected using T16SEL[1:0]/T16A_CTL*x* register. This selection enables the two or more channels capture values of the same 16-bit counter.

The trigger edge of the input signal can be selected using the CAPATRG[1:0]/T16A_CCCTL*x* register for capture A and CAPBTRG[1:0]/T16A_CCCTL*x* register for capture B.

Table 16.4.1.1 Capture Trigger Edge Selection

CAPATRG[1:0]/ CAPBTRG[1:0]	Trigger edge
0x3	Falling edge and rising edge
0x2	Falling edge
0x1	Rising edge
0x0	Not triggered

(Default: 0x0)

When a specified trigger edge is input during counting, the current counter value is loaded to the capture register. At the same time the capture A or capture B interrupt flag is set and an interrupt signal is output to the ITC if the interrupt has been enabled. This interrupt can be used to read the captured data from the T16A_CCA*x* or T16A_CCB*x* register. For example, external event cycles and pulse widths can be measured from the difference between two captured counter values read.

If the captured data is overwritten by the next trigger when the capture A or capture B interrupt flag has already been set, the overwrite interrupt flag will be set. This interrupt can be used to execute an overwrite error handling. To avoid occurrence of unnecessary overwrite interrupt, the capture A or capture B interrupt flag must be reset after the captured data has been read from the T16A_CCA*x* or T16A_CCB*x* register.

Notes:
- The correct captured data may not be obtained if the captured data is read at the same time the next value is being captured. Read the capture register twice to check if the read data is correct as necessary.
- To capture counter data properly, both the High and Low period of the T16A_ATMA_*x*/T16A_ATMB_*x* trigger signal must be longer than three count clock cycles.

The setting of CAPATRG[1:0] or CAPBTRG[1:0] is ineffective in comparator mode. No counter capturing operation will be performed, as the T16A_ATMA_*x*/T16A_ATMB_*x* pin is configured for output.

The capture mode cannot generate/output the timer signal, as no compare signal is generated.

16.4.2 Repeat Mode and One-Shot Mode

Each counter features two count modes: repeat mode and one-shot mode. The count mode is selected using TMMD /T16A_CTL*x* register.

Repeat mode (TMMD = 0, default)

Setting TMMD to 0 sets the counter to repeat mode.

In this mode, once the count starts, the counter continues running until stopped by the application program. If the counter is reset to 0 or returns to 0 due to a counter overflow, the counter continues the count. The counter should be set to this mode to generate periodic interrupts at desired intervals or to generate a timer output waveform.

One-shot mode (TMMD = 1)

Setting TMMD to 1 sets the counter to one-shot mode.
In this mode, the counter stops automatically as soon as the counter is reset or it overflows. The counter should be set to this mode to set a specific wait time or for pulse width measurement.

16.5 Counter Control

16.5.1 Counter Reset

The counter can be reset to 0 by writing 1 to PRESET/T16A_CTL*x* register.

Normally, the counter should be reset by writing 1 to this bit before starting the count.
The counter is reset by the hardware if the counter reaches the compare B register value after the count starts.

Notes: • The counter is reset in sync with the counter clock. It may take long time depending on the count clock selected. To make sure that the reset operation has finished, check if PRESET is set to 0 or BUSY/T16A_CTL*x* register is set to 0 (idle) by reading these bits.

• Resetting by PRESET does not stop the counter. If the counter is running at reset, it will continue counting from counter value 0.

16.5.2 Counter RUN/STOP Control

Make the following settings before starting the count operation.
(1) Switch the input/output pin functions to be used for T16A6. Refer to the "I/O Port (GPIO)" chapter.
(2) Select operating modes. See Section 16.4.
(3) Select the clock source. See Section 16.3.
(4) Configure the timer outputs. See Section 16.6.
(5) If using interrupts, set the interrupt level and enable the T16A6 interrupts. See Section 16.7.
(6) Reset the counter to 0. See Section 16.5.1.
(7) Set comparison data (in comparator mode). See Section 16.4.1.

The T16A6 module provides PRUN/T16A_CTL*x* register to control the counter operation.
The counter starts counting after 1 is written to PRUN (see Notes below). Writing 0 to PRUN disables clock input and stops the count.
This control does not affect the counter data. The counter data is retained even when the count is halted, allowing resumption of the count from that data.
If PRUN and PRESET are written as 1 simultaneously, the counter starts counting after reset.

Notes: • Always make sure that BUSY/T16A_CTL*x* register is set to 0 (idle) before writing to the T16A_CTL*x* register. The T16A_CTL*x* register bits can only be altered after BUSY is set to 0.

• Setting PRUN to 1 may not start the counter immediately as the counter starts counting in sync with the count clock. The counter operation should be checked by reading PRUN (check if it is set to 1) or BUSY (check if it is set to 0).

16.5.3 Reading Counter Values

The counter value can be read from T16ATC[15:0]/T16A_TC*x* register even if the counter is running. However, the counter value should be read at once using a 16-bit transfer instruction. If data is read twice using an 8-bit transfer instruction, the correct value may not be obtained due to occurrence of count up between readings.

Note: The counter value must be read from the T16A_TC*x* register of the channel selected using T16SEL[1:0]/T16A_CTL*x* register.

16.5.4 Timing Charts

Comparator mode

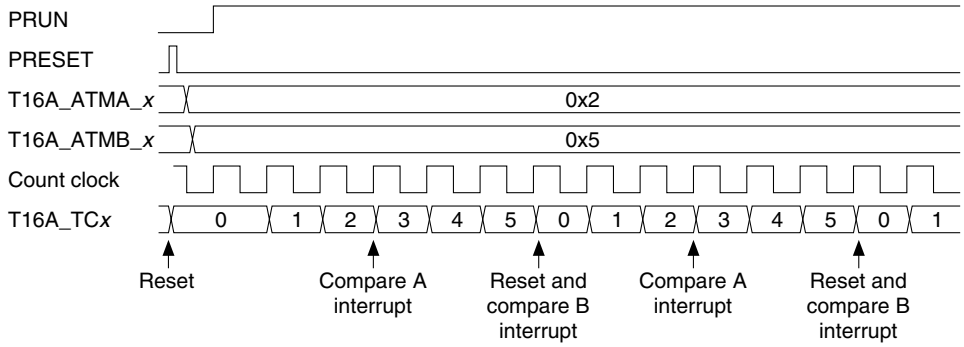


Figure 16.5.4.1 Operation Timing in Comparator Mode

Capture mode

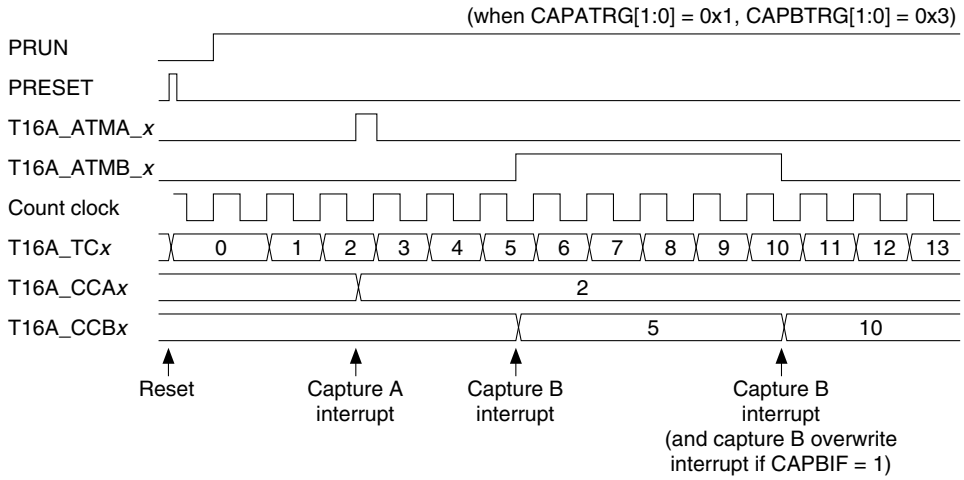


Figure 16.5.4.2 Operation Timing in Capture Mode

16.6 Timer Output Control

T16A6 in comparator mode can generate two TOUT signals using the compare A and compare B signals and can output them to external devices. Figure 16.6.1 shows the TOUT output circuit.

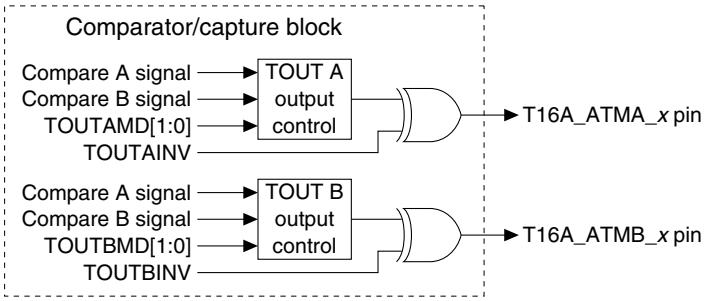


Figure 16.6.1 TOUT Output Circuit

T16A6 includes two TOUT output circuits and their signal generation and output can be controlled individually. Although the output circuit and register names use letters 'A' and 'B' to distinguish two systems, it does not mean that they correspond to compare A and B signals.

Note: The compare A and compare B signals can be generated from the counter value of another channel by setting T16SEL[1:0]/T16A_CTL*x* register.

TOUT output pins

The TOUT A signal is output from the T16A_ATMA_*x* pin and TOUT B signal is output from the T16A_ATMB_*x* pin. The T16A_ATMA_*x* and T16A_ATMB_*x* pins are shared with the capture trigger inputs. They are configured for output when the system A or B is set to comparator mode.

TOUT generation mode

TOUTAMD[1:0]/T16A_CCCTL*x* register (for system A) or TOUTBMD[1:0]/T16A_CCCTL*x* register (for system B) is used to set how the TOUT signal is changed by the compare A and compare B signals.

Table 16.6.1 TOUT Generation Mode

TOUTAMD[1:0]/ TOUTBMD[1:0]	When compare A occurs	When compare B occurs
0x3	No change	Toggle
0x2	Toggle	No change
0x1	Rise	Fall
0x0	Disable output	

(Default: 0x0)

TOUTAMD[1:0] and TOUTBMD[1:0] are also used to turn the TOUT output on and off.

TOUT signal polarity selection

By default, an active high output signal is generated. This logic can be inverted using TOUTAINV/T16A_CCCTL*x* register (for system A) or TOUTBINV/T16A_CCCTL*x* register (for system B). Writing 1 to TOUTAINV/TOUTBINV causes the timer to generate an active low TOUT signal.

Resetting the counter sets the TOUT signal to the inactive level.

Figure 16.6.2 shows the TOUT output waveform.

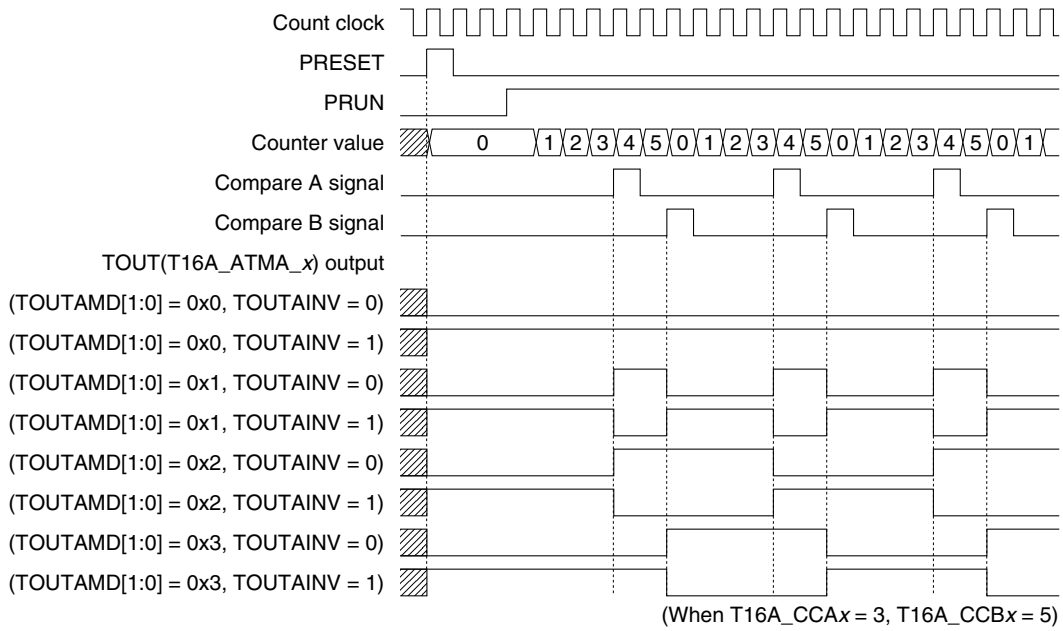


Figure 16.6.2 TOUT Output Waveform

16.7 T16A6 Interrupts and DMA

This section describes the T16A6 interrupts and invoking DMA.
For more information on interrupt processing and DMA transfer, see the "Interrupt Controller (ITC)" chapter and the "DMA Controller (DMAC)" chapter, respectively.

16.7.1 Interrupts

Each T16A6 channel can generate the following six kinds of interrupts:
- Compare A interrupt (in comparator mode)
- Compare B interrupt (in comparator mode)
- Capture A interrupt (in capture mode)
- Capture B interrupt (in capture mode)
- Capture A overwrite interrupt (in capture mode)
- Capture B overwrite interrupt (in capture mode)

A T16A6 channel outputs a single interrupt signal shared by the above interrupt causes to the interrupt controller (ITC). Read the interrupt flags in the T16A6 module to identify the interrupt cause that has been occurred.

Interrupts in comparator mode

Compare A interrupt

This interrupt request is generated when the counter matches the compare A register value during counting in comparator mode. It sets the interrupt flag CAIF/T16A_IFLG*x* register in the T16A6 module to 1.
To use this interrupt, set CAIE/T16A_IEN*x* register to 1. If CAIE is set to 0 (default), interrupt requests for this cause is not sent to the ITC.

Compare B interrupt

This interrupt request is generated when the counter matches the compare B register value during counting in comparator mode. It sets the interrupt flag CBIF/T16A_IFLG*x* register in the T16A6 module to 1.
To use this interrupt, set CBIE/T16A_IEN*x* register to 1. If CBIE is set to 0 (default), interrupt requests for this cause is not sent to the ITC.

Note: The compare A data register (T16A_CCA*x*) and compare B data register (T16A_CCB*x*) must be set to four or more. If a value less than four is set, no cause of interrupt will occur.

Interrupts in capture mode

Capture A interrupt

This interrupt request is generated when the counter value is captured in the capture A register by an external trigger during counting in capture mode. It sets the interrupt flag CAPAIF/T16A_IFLG*x* register in the T16A6 module to 1.
To use this interrupt, set CAPAIE/T16A_IEN*x* register to 1. If CAPAIE is set to 0 (default), interrupt requests for this cause is not sent to the ITC.

Capture B interrupt

This interrupt request is generated when the counter value is captured in the capture B register by an external trigger during counting in capture mode. It sets the interrupt flag CAPBIF/T16A_IFLG*x* register in the T16A6 module to 1.
To use this interrupt, set CAPBIE/T16A_IEN*x* register to 1. If CAPBIE is set to 0 (default), interrupt requests for this cause is not sent to the ITC.

Capture A overwrite interrupt

This interrupt request is generated if the capture A register is overwritten by a new external trigger when the capture A interrupt flag CAPAIF has been set (a counter value has already been loaded to the capture A register). It sets the interrupt flag CAPAOWIF/T16A_IFLG*x* register in the T16A6 module to 1.

To use this interrupt, set CAPAOWIE/T16A_IEN*x* register to 1. If CAPAOWIE is set to 0 (default), interrupt requests for this cause is not sent to the ITC.
CAPAOWIF will be set if the capture A register is overwritten when CAPAIF has been set regardless of whether the capture A register has been read or not. Therefore, be sure to reset CAPAIF immediately after the capture A register is read.

Capture B overwrite interrupt

This interrupt request is generated if the capture B register is overwritten by a new external trigger when the capture B interrupt flag CAPBIF has been set (a counter value has already been loaded to the capture B register). It sets the interrupt flag CAPBOWIF/T16A_IFLG*x* register in the T16A6 module to 1.
To use this interrupt, set CAPBOWIE/T16A_IEN*x* register to 1. If CAPBOWIE is set to 0 (default), interrupt requests for this cause is not sent to the ITC.
CAPBOWIF will be set if the capture B register is overwritten when CAPBIF has been set regardless of whether the capture B register has been read or not. Therefore, be sure to reset CAPBIF immediately after the capture B register is read.

If the interrupt flag is set to 1 when the interrupt has been enabled, the T16A6 module outputs an interrupt request to the ITC. An interrupt is generated if the ITC and S1C33 PE Core interrupt conditions are satisfied.
For more information on interrupt control registers and the operation when an interrupt occurs, see the "Interrupt Controller (ITC)" chapter.

Notes: • Reset the interrupt flag before enabling interrupts with the interrupt enable bit to prevent occurrence of unwanted interrupt. The interrupt flag is reset by writing 1.

• After an interrupt occurs, the interrupt flag in the T16A6 module must be reset in the interrupt handler routine.

16.7.2 DMA Transfer

The causes of compare A/capture A and compare B/capture B interrupts can invoke a DMA. This allows continuous data transfer via the DMAC between memory and the compare/capture data register. The interrupt signal is output to both the ITC and DMAC. Therefore, DMA transfer can be performed without generating a T16A6 interrupt.
The DMAC provides two channels (Ch.2 and 3 or Ch.4 and 5) for T16A6. The T16A6 channel used to invoke DMAC Ch.2 and 3 can be selected using DMASEL[1:0]/T16_CTL0 register. The T16A6 channel used to invoke DMAC Ch.4 and 5 can be selected using DMASEL[1:0]/T16_CTL1 register.

Table 16.7.2.1 DMAC Ch.2/3 Invoking Channel Selection

DMASEL[1:0]/T16_CTL0 register	T16A6 channel that invokes DMAC Ch.2/3
0x3	T16A6 Ch.3
0x2	T16A6 Ch.2
0x1	T16A6 Ch.1
0x0	T16A6 Ch.0

(Default: 0x0)

Table 16.7.2.2 DMAC Ch.4/5 Invoking Channel Selection

DMASEL[1:0]/T16_CTL1 register	T16A6 channel that invokes DMAC Ch.4/5
0x3	T16A6 Ch.3
0x2	T16A6 Ch.2
0x1	T16A6 Ch.1
0x0	T16A6 Ch.0

(Default: 0x0)

For more information on DMA transfer, see the "DMA Controller (DMAC)" chapter.

16.8 Control Register Details

Table 16.8.1 List of 16-bit PWM Timer (T16A6) Register

Address	Register name		Function
0x301160	T16A_CTL0	T16A6 Ch.0 Counter Control Register	Control counter
0x301162	T16A_TC0	T16A6 Ch.0 Counter Data Register	Counter data
0x301164	T16A_CCCTL0	T16A6 Ch.0 Comparator/Capture Control Register	Control comparator/capture block and TOUT
0x301166	T16A_CCA0	T16A6 Ch.0 Comparator/Capture A Data Register	Compare A/capture A data
0x301168	T16A_CCB0	T16A6 Ch.0 Comparator/Capture B Data Register	Compare B/capture B data
0x30116a	T16A_IEN0	T16A6 Ch.0 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
0x30116c	T16A_IFLG0	T16A6 Ch.0 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
0x301170	T16A_CTL1	T16A6 Ch.1 Counter Control Register	Control counter
0x301172	T16A_TC1	T16A6 Ch.1 Counter Data Register	Counter data
0x301174	T16A_CCCTL1	T16A6 Ch.1 Comparator/Capture Control Register	Control comparator/capture block and TOUT
0x301176	T16A_CCA1	T16A6 Ch.1 Comparator/Capture A Data Register	Compare A/capture A data
0x301178	T16A_CCB1	T16A6 Ch.1 Comparator/Capture B Data Register	Compare B/capture B data
0x30117a	T16A_IEN1	T16A6 Ch.1 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
0x30117c	T16A_IFLG1	T16A6 Ch.1 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
0x301180	T16A_CTL2	T16A6 Ch.2 Counter Control Register	Control counter
0x301182	T16A_TC2	T16A6 Ch.2 Counter Data Register	Counter data
0x301184	T16A_CCCTL2	T16A6 Ch.2 Comparator/Capture Control Register	Control comparator/capture block and TOUT
0x301186	T16A_CCA2	T16A6 Ch.2 Comparator/Capture A Data Register	Compare A/capture A data
0x301188	T16A_CCB2	T16A6 Ch.2 Comparator/Capture B Data Register	Compare B/capture B data
0x30118a	T16A_IEN2	T16A6 Ch.2 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
0x30118c	T16A_IFLG2	T16A6 Ch.2 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
0x301190	T16A_CTL3	T16A6 Ch.3 Counter Control Register	Control counter
0x301192	T16A_TC3	T16A6 Ch.3 Counter Data Register	Counter data
0x301194	T16A_CCCTL3	T16A6 Ch.3 Comparator/Capture Control Register	Control comparator/capture block and TOUT
0x301196	T16A_CCA3	T16A6 Ch.3 Comparator/Capture A Data Register	Compare A/capture A data
0x301198	T16A_CCB3	T16A6 Ch.3 Comparator/Capture B Data Register	Compare B/capture B data
0x30119a	T16A_IEN3	T16A6 Ch.3 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
0x30119c	T16A_IFLG3	T16A6 Ch.3 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status

The T16A6 registers are described in detail below.

Note: When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

T16A6 Ch.*x* Counter Control Registers (T16A_CTL*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.0 Counter Control Register (T16A_CTL0)	**0x301160** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13–12	**DMASEL [1:0]**	DMAC Ch.2/3 invoking channel select	DMASEL[1:0] / T16A6 channel: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x0	R/W	
		D11–8	**CLKS[3:0]**	Counter clock (division ratio) select	CLKS[3:0] / Division ratio: 0xf External clock; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	**BUSY**	Register writing status	1 Busy; 0 Idle	0	R	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**T16SEL [1:0]**	Counter select	T16SEL[1:0] / Counter channel: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x0	R/W	
		D3	**CBUFEN**	Compare buffer enable	1 Enable; 0 Disable	0	R/W	
		D2	**TMMD**	Count mode select	1 One-shot; 0 Repeat	0	R/W	
		D1	**PRESET**	Counter reset	1 Reset; 0 Ignored	0	W	0 when being read.
		D0	**PRUN**	Counter run/stop control	1 Run; 0 Stop	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.1 Counter Control Register (T16A_CTL1)	**0x301170** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13–12	**DMASEL [1:0]**	DMAC Ch.4/5 invoking channel select	DMASEL[1:0]: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x0	R/W	
		D11–8	**CLKS[3:0]**	Counter clock (division ratio) select	CLKS[3:0]: 0xf External clock; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	**BUSY**	Register writing status	1 Busy; 0 Idle	0	R	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**T16SEL [1:0]**	Counter select	T16SEL[1:0]: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x1	R/W	
		D3	**CBUFEN**	Compare buffer enable	1 Enable; 0 Disable	0	R/W	
		D2	**TMMD**	Count mode select	1 One-shot; 0 Repeat	0	R/W	
		D1	**PRESET**	Counter reset	1 Reset; 0 Ignored	0	W	0 when being read.
		D0	**PRUN**	Counter run/stop control	1 Run; 0 Stop	0	R/W	
T16A6 Ch.2 Counter Control Register (T16A_CTL2)	**0x301180** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**CLKS[3:0]**	Counter clock (division ratio) select	CLKS[3:0]: 0xf External clock; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	**BUSY**	Register writing status	1 Busy; 0 Idle	0	R	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**T16SEL [1:0]**	Counter select	T16SEL[1:0]: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x2	R/W	
		D3	**CBUFEN**	Compare buffer enable	1 Enable; 0 Disable	0	R/W	
		D2	**TMMD**	Count mode select	1 One-shot; 0 Repeat	0	R/W	
		D1	**PRESET**	Counter reset	1 Reset; 0 Ignored	0	W	0 when being read.
		D0	**PRUN**	Counter run/stop control	1 Run; 0 Stop	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.3 Counter Control Register (T16A_CTL3)	0x301190 (16 bits)	D15–12	–	reserved	–	–	–	0 when being read.
		D11–8	CLKS[3:0]	Counter clock (division ratio) select	CLKS[3:0] / Division ratio: 0xf External clock; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	BUSY	Register writing status	1 Busy / 0 Idle	0	R	
		D6	–	reserved	–	–	–	0 when being read.
		D5–4	T16SEL [1:0]	Counter select	T16SEL[1:0] / Counter channel: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x3	R/W	
		D3	CBUFEN	Compare buffer enable	1 Enable / 0 Disable	0	R/W	
		D2	TMMD	Count mode select	1 One-shot / 0 Repeat	0	R/W	
		D1	PRESET	Counter reset	1 Reset / 0 Ignored	0	W	0 when being read.
		D0	PRUN	Counter run/stop control	1 Run / 0 Stop	0	R/W	

D[15:14] Reserved

D[13:12] DMASEL[1:0]: DMAC Ch.2/3 Invoking Channel Select Bits (T16A_CTL0)
DMASEL[1:0]: DMAC Ch.4/5 Invoking Channel Select Bits (T16A_CTL1)

Selects the T16A6 channel to be used for DMA transfer (DMAC Ch.2/3 or DMAC Ch.4/5) when a cause of compare A/B or capture A/B interrupt occurs.

Table 16.8.2 DMAC Ch.2/3 Invoking Channel Selection

DMASEL[1:0]/T16_CTL0 register	T16A6 channel that invokes DMAC Ch.2/3
0x3	T16A6 Ch.3
0x2	T16A6 Ch.2
0x1	T16A6 Ch.1
0x0	T16A6 Ch.0

(Default: 0x0)

Table 16.8.3 DMAC Ch.4/5 Invoking Channel Selection

DMASEL[1:0]/T16_CTL1 register	T16A6 channel that invokes DMAC Ch.4/5
0x3	T16A6 Ch.3
0x2	T16A6 Ch.2
0x1	T16A6 Ch.1
0x0	T16A6 Ch.0

(Default: 0x0)

D[13:12] Reserved (T16A_CTL2, T16A_CTL3)

D[11:8] CLKS[3:0]: Counter Clock Select Bits

Selects the Ch.*x* counter clock from the 15 different prescaler (PSC0) output clocks (TCLK (SYSCLK/OSC3/OSC1) division ratio) and an external clock (EXCL*x* input clock).

Table 16.8.4 Counter Clock (TCLK Division Ratio) Selection

CLKS[3:0]	Division ratio	CLKS[3:0]	Division ratio
0xf	External clock	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Note: • Make sure the counter is halted before setting the count clock.

• When using an external clock, the external clock cycle must be at least two CPU operating clock cycles.

D7 BUSY: Register Writing Status Bit

Indicates the T16A6 register writing status.

1 (R): Busy

0 (R): Idle (default)

BUSY goes 1 when data is written to the T16A_CTL*x*, T16A_CCA*x*, or T16A_CCB*x* register and it reverts to 0 upon completion of the writing operation.

Note: Make sure that BUSY is set to 0 before writing to the T16A6 registers.

D6 Reserved

D[5:4] T16SEL[1:0]: Counter Select Bits

Selects the counter channel.

Table 16.8.5 Counter Channel Selection

T16SEL[1:0]	Counter channel
0x3	Ch.3
0x2	Ch.2
0x1	Ch.1
0x0	Ch.0

(Default: 0x0/T16A6 Ch.0, 0x1/T16A6 Ch.1, 0x2/T16A6 Ch.2, 0x3/T16A6 Ch.3)

A timer channel (comparator/capture block) allows use of the counter in another channel. This enables the comparator/capture blocks to compare and capture values of the same counter.
The counter value must be read from the T16A_TC*x* register of the channel selected using T16SEL[1:0].

D3 CBUFEN: Compare Buffer Enable Bit

Enables or disables writing to the compare buffer.

1 (R/W): Enabled

0 (R/W): Disabled (default)

When CBUFEN is set to 1, compare data is written via the compare data buffer. The buffer contents are loaded into the compare A and compare B registers by the compare B signal.
When CBUFEN is set to 0, compare data is written directly to the compare A and compare B registers.
Compare data is always read from compare A and compare B registers.

D2 TMMD: Count Mode Select Bit

Selects the count mode.

1 (R/W): One-shot mode

0 (R/W): Repeat mode (default)

Setting TMMD to 0 sets the counter to repeat mode. In this mode, once the count starts, the counter continues counting until stopped by the application program.
Setting TMMD to 1 sets the counter to one-shot mode. In this mode, the counter stops counting automatically as soon as the counter is reset by the compare B signal as well as stopped via software.

D1 PRESET: Counter Reset Bit

Resets the counter.

1 (W): Reset
0 (W): Ignored
0 (R): Normally 0 when read out (default)

Writing 1 to this bit resets the counter to 0.

D0 PRUN: Counter Run/Stop Control Bit

Starts/stops the count.

1 (W): Run
0 (W): Stop
1 (R): Counting
0 (R): Stopped (default)

The counter starts counting when PRUN is written as 1 and stops when written as 0. The counter data is retained even if the counter is stopped.

T16A6 Ch.*x* Counter Data Registers (T16A_TC*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.*x* Counter Data Register (T16A_TC*x*)	**0x301162 0x301172 0x301182 0x301192** (16 bits)	D15–0	**T16ATC [15:0]**	Counter data T16ATC15 = MSB T16ATC0 = LSB	0x0 to 0xffff	0x0	R	

D[15:0] T16ATC[15:0]: Counter Data Bits

Counter data can be read out. (Default: 0x0)

The counter value can be read out even if the counter is running. However, the counter value should be read at once using a 16-bit transfer instruction. If data is read twice using an 8-bit transfer instruction, the correct value may not be obtained due to occurrence of count up between readings.

Note: The counter value must be read from the T16A_TC*x* register of the channel selected using T16SEL[1:0]/T16A_CTL*x* register.

T16A6 Ch.*x* Comparator/Capture Control Registers (T16A_CCCTL*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.*x* Comparator/ Capture Control Register (T16A_CCCTL*x*)	**0x301164 0x301174 0x301184 0x301194** (16 bits)	D15–14	**CAPBTRG [1:0]**	Capture B trigger select	CAPBTRG[1:0] Trigger edge 0x3 ↑ and ↓ 0x2 ↓ 0x1 ↑ 0x0 None	0x0	R/W	
		D13–12	**TOUTBMD [1:0]**	TOUT B mode select	TOUTBMD[1:0] Mode 0x3 cmp B: ↑ or ↓ 0x2 cmp A: ↑ or ↓ 0x1 cmp A: ↑, B: ↓ 0x0 Off	0x0	R/W	
		D11–10	–	reserved	–	–	–	0 when being read.
		D9	**TOUTBINV**	TOUT B invert	1 Invert 0 Normal	0	R/W	
		D8	**CCBMD**	T16A_CCB register mode select	1 Capture 0 Comparator	0	R/W	
		D7–6	**CAPATRG [1:0]**	Capture A trigger select	CAPATRG[1:0] Trigger edge 0x3 ↑ and ↓ 0x2 ↓ 0x1 ↑ 0x0 None	0x0	R/W	
		D5–4	**TOUTAMD [1:0]**	TOUT A mode select	TOUTAMD[1:0] Mode 0x3 cmp B: ↑ or ↓ 0x2 cmp A: ↑ or ↓ 0x1 cmp A: ↑, B: ↓ 0x0 Off	0x0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	**TOUTAINV**	TOUT A invert	1 Invert 0 Normal	0	R/W	
		D0	**CCAMD**	T16A_CCA register mode select	1 Capture 0 Comparator	0	R/W	

D[15:14] CAPBTRG[1:0]: Capture B Trigger Select Bits

Selects the trigger edge(s) of the external signal (T16A_ATMB_*x* input) at which the counter value is captured in the capture B register.

Table 16.8.6 Capture B Trigger Edge Selection

CAPBTRG[1:0]	Trigger edge
0x3	Falling edge and rising edge
0x2	Falling edge
0x1	Rising edge
0x0	Not triggered

(Default: 0x0)

CAPBTRG[1:0] are control bits for capture mode and are ineffective in comparator mode.

D[13:12] TOUTBMD[1:0]: TOUT B Mode Select Bits

Configures how the TOUT B signal waveform (T16A_ATMB_*x* output) is changed by the compare A and compare B signals. These bits are also used to turn the TOUT B output on and off.

Table 16.8.7 TOUT B Generation Mode

TOUTBMD[1:0]	When compare A occurs	When compare B occurs
0x3	No change	Toggle
0x2	Toggle	No change
0x1	Rise	Fall
0x0	Disable output	

(Default: 0x0)

TOUTBMD[1:0] are control bits for comparator mode and are ineffective in capture mode.

D[11:10] Reserved

D9 TOUTBINV: TOUT B Invert Bit

Selects the TOUT B signal (T16A_ATMB_*x* output) polarity.
1 (R/W): Inverted (active low)
0 (R/W): Normal (active high) (default)

Writing 1 to TOUTBINV generates an active low signal (off level = high) for the TOUT B output. When TOUTBINV is 0, an active high signal (off level = low) is generated.
TOUTBINV is a control bit for comparator mode and is ineffective in capture mode.

D8 CCBMD: T16A_CCB Register Mode Select Bit

Selects the T16A_CCB*x* register function (comparator mode or capture mode).
1 (R/W): Capture mode
0 (R/W): Comparator mode (default)

Writing 1 to CCBMD configures the T16A_CCB*x* register as the capture B register (capture mode) to which the counter data will be loaded by the external trigger signal. When CCBMD is 0, the T16A_CCB*x* register functions as the compare B register (comparator mode) for writing a comparison value to generate the compare B signal.

D[7:6] CAPATRG[1:0]: Capture A Trigger Select Bits

Selects the trigger edge(s) of the external signal (T16A_ATMA_*x* input) at which the counter value is captured in the capture A register.

Table 16.8.8 Capture A Trigger Edge Selection

CAPATRG[1:0]	Trigger edge
0x3	Falling edge and rising edge
0x2	Falling edge
0x1	Rising edge
0x0	Not triggered

(Default: 0x0)

CAPATRG[1:0] are control bits for capture mode and are ineffective in comparator mode.

D[5:4] TOUTAMD[1:0]: TOUT A Mode Select Bits

Configures how the TOUT A signal waveform (T16A_ATMA_*x* output) is changed by the compare A and compare B signals. These bits are also used to turn the TOUT A output on and off.

Table 16.8.9 TOUT A Generation Mode

TOUTAMD[1:0]	When compare A occurs	When compare B occurs
0x3	No change	Toggle
0x2	Toggle	No change
0x1	Rise	Fall
0x0	Disable output	

(Default: 0x0)

TOUTAMD[1:0] are control bits for comparator mode and are ineffective in capture mode.

D[3:2] Reserved

D1 TOUTAINV: TOUT A Invert Bit

Selects the TOUT A signal (T16A_ATMA_*x* output) polarity.
1 (R/W): Inverted (active low)
0 (R/W): Normal (active high) (default)

Writing 1 to TOUTAINV generates an active low signal (off level = high) for the TOUT A output. When TOUTAINV is 0, an active high signal (off level = low) is generated.
TOUTAINV is a control bit for comparator mode and is ineffective in capture mode.

D0 CCAMD: T16A_CCA Register Mode Select Bit

Selects the T16A_CCA*x* register function (comparator mode or capture mode).
1 (R/W): Capture mode
0 (R/W): Comparator mode (default)

Writing 1 to CCAMD configures the T16A_CCA*x* register as the capture A register (capture mode) to which the counter data will be loaded by the external trigger signal. When CCAMD is 0, the T16A_CCA*x* register functions as the compare A register (comparator mode) for writing a comparison value to generate the compare A signal.

T16A6 Ch.*x* Comparator/Capture A Data Registers (T16A_CCA*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.*x* Comparator/ Capture A Data Register (T16A_CCA*x*)	**0x301166** **0x301176** **0x301186** **0x301196** (16 bits)	D15–0	**CCA[15:0]**	Compare/capture A data CCA15 = MSB CCA0 = LSB	0x0 to 0xffff	0x0	R/W	

D[15:0] CCA[15:0]: Compare/Capture A Data Bits

In comparator mode (CCAMD/ T16A_CCCTL*x* register = 0)
Sets a compare A data, which will be compared with the counter value, through this register.
When CBUFEN/T16A_CTL*x* register is set to 0, accessing to this register directly read/write from/to the compare A register.
When CBUFEN is set to 1, accessing to this register read/write from/to the compare A buffer. The buffer contents are loaded into the compare A register when the counter is reset by the compare B signal.
The data set is compared with the counter data. When the counter reaches the comparison value set, the compare A signal is asserted and a cause of compare A interrupt occurs. Furthermore, the TOUT output waveform changes when TOUTAMD[1:0]/T16A_CCCTL*x* register or TOUTBMD[1:0]/T16A_CCCTL*x* register is set to 0x2 or 0x1. These processes do not affect the counter data and the count up operation.

In capture mode (CCAMD = 1)
When the counter value is captured at the external trigger signal (T16A_ATMA_*x* input) edge selected using CAPATRG[1:0]/T16A_CCCTL*x* register, the captured value is loaded to this register. At the same time a capture A interrupt can be generated, thus the captured counter value can be read out in the interrupt handler.

T16A6 Ch.*x* Comparator/Capture B Data Registers (T16A_CCB*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.*x* Comparator/ Capture B Data Register (T16A_CCB*x*)	**0x301168** **0x301178** **0x301188** **0x301198** (16 bits)	D15–0	**CCB[15:0]**	Compare/capture B data CCB15 = MSB CCB0 = LSB	0x0 to 0xffff	0x0	R/W	

D[15:0] CCB[15:0]: Compare/Capture B Data Bits

In comparator mode (CCBMD/ T16A_CCCTL*x* register = 0)

Sets a compare B data, which will be compared with the counter value, through this register.

When CBUFEN/T16A_CTL*x* register is set to 0, accessing to this register directly read/write from/to the compare B register.

When CBUFEN is set to 1, accessing to this register read/write from/to the compare B buffer. The buffer contents are loaded into the compare B register when the counter is reset by the compare B signal.

The data set is compared with the counter data. When the counter reaches the comparison value set, the compare B signal is asserted and a cause of compare B interrupt occurs. The counter is reset to 0. Furthermore, the TOUT output waveform changes when TOUTAMD[1:0]/T16A_CCCTL*x* register or TOUTBMD[1:0]/T16A_CCCTL*x* register is set to 0x3 or 0x1.

In capture mode (CCBMD = 1)

When the counter value is captured at the external trigger signal (T16A_ATMB_*x* input) edge selected using CAPBTRG[1:0]/T16A_CCCTL*x* register, the captured value is loaded to this register. At the same time a capture B interrupt can be generated, thus the captured counter value can be read out in the interrupt handler.

T16A6 Ch.*x* Comparator/Capture Interrupt Enable Registers (T16A_IEN*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.*x* Comparator/ Capture Interrupt Enable Register (T16A_IEN*x*)	**0x30116a** **0x30117a** **0x30118a** **0x30119a** (16 bits)	D15–6	–	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIE**	Capture B overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**CAPAOWIE**	Capture A overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D3	**CAPBIE**	Capture B interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**CAPAIE**	Capture A interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**CBIE**	Compare B interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**CAIE**	Compare A interrupt enable	1 Enable 0 Disable	0	R/W	

D[15:6] Reserved

D5 CAPBOWIE: Capture B Overwrite Interrupt Enable Bit

Enables or disables capture B overwrite interrupts.

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

Setting CAPBOWIE to 1 enables capture B overwrite interrupt requests to the ITC. Setting it to 0 disables interrupts.

D4 CAPAOWIE: Capture A Overwrite Interrupt Enable Bit

Enables or disables capture A overwrite interrupts.

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

Setting CAPAOWIE to 1 enables capture A overwrite interrupt requests to the ITC. Setting it to 0 disables interrupts.

D3 CAPBIE: Capture B Interrupt Enable Bit

Enables or disables capture B interrupts.

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

Setting CAPBIE to 1 enables capture B interrupt requests to the ITC. Setting it to 0 disables interrupts.

D2 CAPAIE: Capture A Interrupt Enable Bit

Enables or disables capture A interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting CAPAIE to 1 enables capture A interrupt requests to the ITC. Setting it to 0 disables interrupts.

D1 CBIE: Compare B Interrupt Enable Bit

Enables or disables compare B interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting CBIE to 1 enables compare B interrupt requests to the ITC. Setting it to 0 disables interrupts.

D0 CAIE: Compare A Interrupt Enable Bit

Enables or disables compare A interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting CAIE to 1 enables compare A interrupt requests to the ITC. Setting it to 0 disables interrupts.

T16A6 Ch.*x* Comparator/Capture Interrupt Flag Registers (T16A_IFLG*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
T16A6 Ch.*x* Comparator/ Capture Interrupt Flag Register (T16A_IFLG*x*)	0x30116c 0x30117c 0x30118c 0x30119c (16 bits)	D15–6	–	reserved	–				–	–	0 when being read.
		D5	CAPBOWIF	Capture B overwrite interrupt flag	1	Cause of interrupt occurred	0	Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D4	CAPAOWIF	Capture A overwrite interrupt flag					0	R/W	
		D3	CAPBIF	Capture B interrupt flag					0	R/W	
		D2	CAPAIF	Capture A interrupt flag					0	R/W	
		D1	CBIF	Compare B interrupt flag					0	R/W	
		D0	CAIF	Compare A interrupt flag					0	R/W	

D[15:6] Reserved

D5 CAPBOWIF: Capture B Overwrite Interrupt Flag Bit

Indicates whether the cause of capture B overwrite interrupt has occurred or not.
1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

CAPBOWIF is a T16A6 interrupt flag that is set to 1 when the capture B register is overwritten.
CAPBOWIF is reset by writing 1.

D4 CAPAOWIF: Capture A Overwrite Interrupt Flag Bit

Indicates whether the cause of capture A overwrite interrupt has occurred or not.
1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

CAPAOWIF is a T16A6 interrupt flag that is set to 1 when the capture A register is overwritten.
CAPAOWIF is reset by writing 1.

D3 CAPBIF: Capture B Interrupt Flag Bit

Indicates whether the cause of capture B interrupt has occurred or not.
1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

CAPBIF is a T16A6 interrupt flag that is set to 1 when the counter value is captured in the capture B register.
CAPBIF is reset by writing 1.

D2 CAPAIF: Capture A Interrupt Flag Bit

Indicates whether the cause of capture A interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

CAPAIF is a T16A6 interrupt flag that is set to 1 when the counter value is captured in the capture A register.
CAPAIF is reset by writing 1.

D1 CBIF: Compare B Interrupt Flag Bit

Indicates whether the cause of compare B interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

CBIF is a T16A6 interrupt flag that is set to 1 when the counter reaches the value set in the compare B register.
CBIF is reset by writing 1.

D0 CAIF: Compare A Interrupt Flag Bit

Indicates whether the cause of compare A interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

CAIF is a T16A6 interrupt flag that is set to 1 when the counter reaches the value set in the compare A register.
CAIF is reset by writing 1.

17 16-bit Audio PWM Timer (T16P)

17.1 T16P Module Overview

The S1C33L27 includes two channels of 16-bit audio PWM timers (T16P) that generate PWM pulses from PCM data. The pulses generated can be directly output to a low pass filter that eliminates quantization noise to shape the output signal into sound waveform. A stereo audio output system can be implemented simply without an external D/A converter.

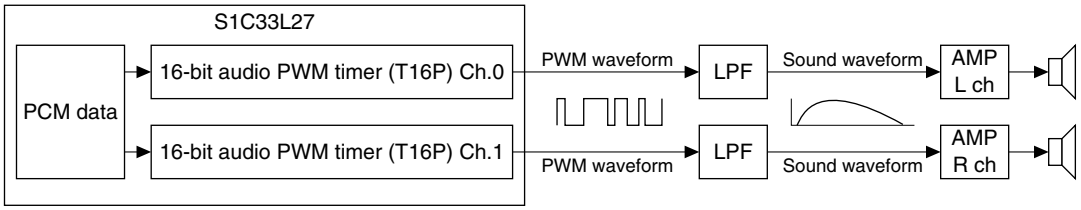


Figure 17.1.1 Audio Output Circuit Using T16P

If no audio output feature is required, T16P can be used as general-purpose 16-bit timers.
The following lists the main functions of T16P.

- Supports 8-bit and 16-bit PCM data with varied sample rates: 8k, 16k, 22.05k, 32k, 44.1k, and 48k.
- Supports both signed and unsigned PCM data.
- Supports split mode; 16-bit audio data can be split into 10 bits + 6 bits, 9 bits + 7 bits, or 8 bits + 8 bits.
- Supports fine mode to improve the precision of the pulse width.
- Includes a digital volume control unit.
- Channel sync function for controlling stereo outputs
- Programmable count clocks using the prescaler or an external clock
- Built-in two 16-bit data buffers for setting pulse widths (duty cycles) and pulse periods
- Can generate three different types of interrupts and invoke a DMA.

Figure 17.1.2 shows the T16P configuration.

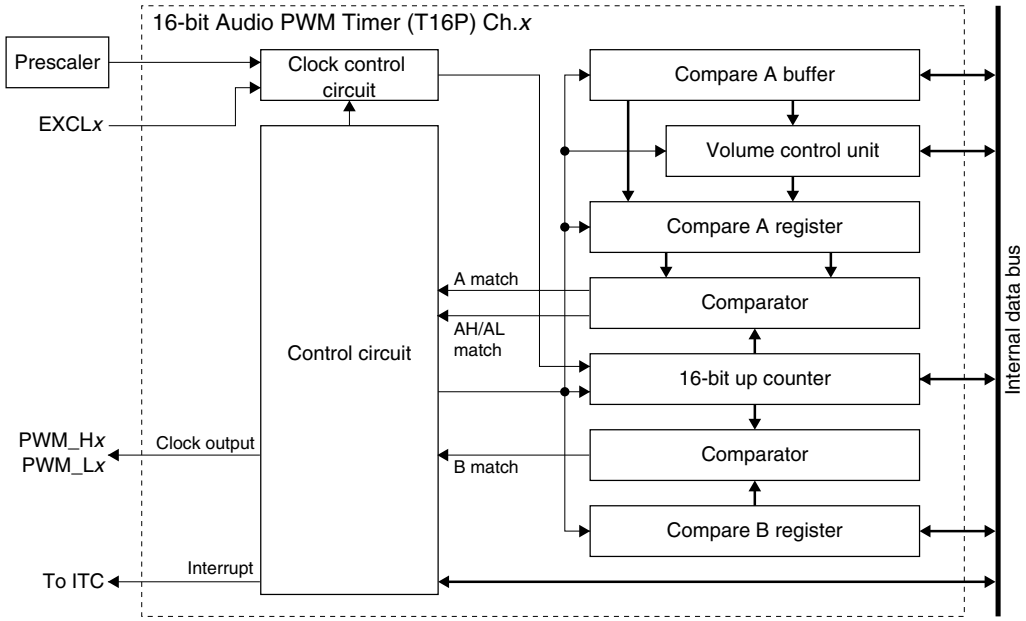


Figure 17.1.2 16-bit Audio PWM Timer (T16P) Configuration (One Channel)

Each T16P channel consists of a 16-bit up-counter and two 16-bit compare data buffers/registers.
The 16-bit counter can be reset to 0 via software and counts up using a prescaler output clock or an external clock input from the EXCL*x* pin.
The compare A buffer is used to store data (PCM data). The stored data is loaded to the compare A register and compared with the counter value to determine the output pulse width. The volume control unit multiplies the PCM data stored in the compare A buffer by the specified volume level set via software before loading to the compare A register. This makes it possible to adjust the volume level to 1/64 through 127/64.
The compare B buffer is used to store data to determine a pulse period. The stored data is loaded to the compare B register and compared with the counter value.
When the counter value reaches the compare data, the timer output signal is inverted to generate PWM waveform.
To control stereo outputs, T16P provides a channel sync function that allows the software to control two channels simultaneously using the registers for one channel.

Note: Both channels of the T16P module have the same functions except for the control register addresses. The description in this section applies to both channels of the T16P module unless otherwise specified. Letter '*x*' in the register name refers to a channel number (0 or 1).
Example: T16P_CTL*x* register
Ch.0: T16P_CTL0 register
Ch.1: T16P_CTL1 register

17.2 T16P Input/Output Pins

Table 17.2.1 lists the input/output pins for the T16P module.

Table 17.2.1 List of T16P Pins

Pin name	I/O	Qty	Function
EXCL0 (Ch.0) EXCL1 (Ch.1)	I	2	T16P external clock input pin Inputs an external clock as the count clock. The EXCL0 pin can also be used as the T16A6 Ch.0–1 external clock input pin. The EXCL1 pin can also be used as the T16A6 Ch.2–3 external clock input pin.
PWM_H0 (Ch.0) PWM_H1 (Ch.1)	O	2	PWM signal output pin In a split mode: Outputs the PWM signal generated from the high-order PCM data bits. In normal mode: Outputs the PWM signal generated from the PCM data.
PWM_L0 (Ch.0) PWM_L1 (Ch.1)	O	2	PWM signal output pin In a split mode: Outputs the PWM signal generated from the low-order PCM data bits. In normal mode: Fixed at the initial output level (or not used).

The T16P input/output pins (EXCL*x*, PWM_H*x*, PWM_L*x*) are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as T16P input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

17.3 Setting T16P Operating Conditions

Make the following settings before starting T16P.
1. Select the count clock. (See Section 17.3.1.)
2. Configure the PCM data (resolution, signed/unsigned data format). (See Section 17.3.2.)
3. Select the operating modes (split mode, fine mode). (See Section 17.3.3.)
4. Set the PWM output condition (initial signal level). (See Section 17.3.4.)
5. Set the initial volume level. (See Section 17.4.4.)
6. Set interrupt and/or DMA conditions. (See Section 17.5.)
7. Reset T16P. (See Section 17.4.1.)
8. Configure the T16P input/output pins. (See Section 17.2.)
9. Start T16P. (See Section 17.4.2.)

17.3.1 Count Clock

Either an internal clock or an external clock can be selected as the count clock using CLKSEL/T16P_CTL*x* register. When CLKSEL is set to 0 (default), an internal clock is used; when set to 1, the external clock input to the EXCL*x* pin is used.
When using an external clock, the external clock cycle must be at least two CPU operating clock cycles.
When an internal clock is used, it can be selected using CLKDIV[3:0]/T16P_CLK*x* register from the 13 types generated by the prescaler (PSC Ch.2) dividing the PCLK clock into 1/1 to 1/4,096.

Table 17.3.1.1 Internal Clock (PCLK Division Ratio) Selections

CLKDIV[3:0]	Division ratio	CLKDIV[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	Reserved	0x6	1/64
0xd	Reserved	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Note: Make sure the counter is halted before setting the count clock.

For controlling PSC Ch.2, refer to the "Prescaler (PSC)" chapter.

17.3.2 PCM Data Configuration

The resolution and data format must be specified for manipulating PCM data.

Data resolution

T16P supports 8-bit and 16-bit PCM data. Use RESSEL/T16P_CTL*x* register to select the resolution. When RESSEL is set to 1 (default), 16-bit resolution is selected; when set to 0, 8-bit resolution is selected.

Notes:
- 16-bit audio data must be written to addresses 0x301200 and 0x301210 (CMPA[15:0]/T16P_A*x* register) in 16-bit size.
- 8-bit audio data must be written to addresses 0x301201 and 0x301211 (CMPA[15:8]/T16P_A*x* register) in 8-bit size. Furthermore, select 8 bits + 8 bits split mode to use 8-bit audio data (RESSEL = 0). The PWM pulse will be output from the PWM_H*x* pin and the PWM_L*x* pin is fixed at the level set by INITOL. In this case, the PWM_L*x* pin can be used for a GPIO or other functions.

Data format

T16P supports signed and unsigned PCM data. Use SGNSEL/T16P_CTL*x* register to select the data format. When SGNSEL is set to 1 (default), signed data format is selected; when set to 0, unsigned data format is selected.

Note: When signed audio data is selected, CMPA15/T16P_A*x* register is treated as the sign bit for both 16-bit and 8-bit audio data.

17.3.3 Operating Mode Selection

Split mode

When 16-bit PCM data is used, it can be manipulated by splitting into two data units. Use SPLTMD[1:0]/T16P_CTL*x* register to select a split mode.

Table 17.3.3.1 Split Mode Selection

SPLTMD[1:0]	Split mode
0x3	10 bits + 6 bits split mode
0x2	9 bits + 7 bits split mode
0x1	8 bits + 8 bits split mode
0x0	16 bits normal mode

(Default: 0x0)

When a split mode is selected, the high-order bits (10, 9, or 8 high-order bits) of the compare A data and the low-order bits (6, 7, or 8 low-order bits) are compared with the counter data and the two comparison results generate two PWM output signals. The PWM signal generated from the high-order data bits is output from the PWM_H*x* pin and another generated from the low-order data bits is output from the PWM_L*x* pin. When a split mode or 8-bit PCM data resolution is selected, compare A interrupts cannot be generated.

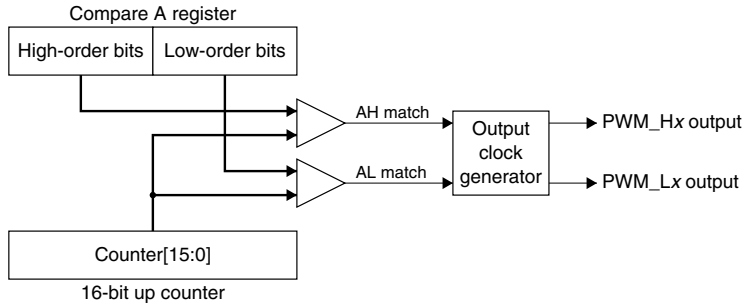


Figure 17.3.3.1 Split Mode

When normal mode is selected (SPLTMD[1:0] = 0x0), 16-bit PCM (compare A) data is compared with the 16-bit counter data and the PWM signal generated is output from the PWM_H*x* pin. If PWM_L*x* output function is enabled, the PWM_L*x* pin is fixed at the initial output level. If this mode and 16-bit PCM data resolution are selected, compare A interrupts can be generated when the counter reaches the compare A data.

Note: When using T16P as a 16-bit timer, set SPLTMD[1:0] to 0x0 (16 bits normal mode).
When using T16P for audio output, set SPLTMD[1:0] to 0x1 (8 bits + 8 bits), 0x2 (9 bits + 7 bits), or 0x3 (10 bits + 6 bits) split mode according to the audio sampling rate.

Fine mode

Normally, compare A data is compared with the counter data at the rising edge of the count clock. When T16P is set to fine mode, the comparisons are performed at both rising and falling edges of the count clock. At this time the compare A data is halved when compared.

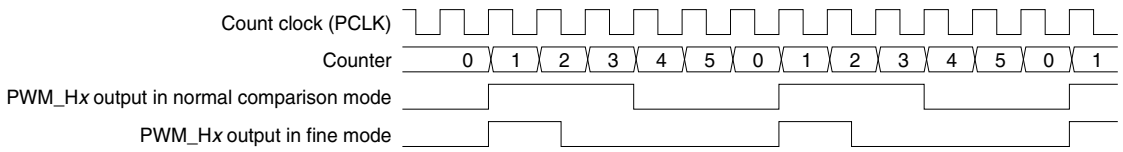


Figure 17.3.3.2 Fine Mode

The fine mode improves the precision of the pulse width. Note, however, that the PCLK/1 clock can only be used as the count clock in this mode. CLKSEL and CLKDIV[3:0] settings are ineffective.
Set SELFM/T16P_CTL*x* to 1 to set T16P to fine mode.
The fine mode does not affect the pulse period that is determined with compare B data.

Note: When using A match interrupts while T16P is placed into fine mode, the maximum value of CMPB[15:0] is limited to 2^{15} - 1 (= 32,767) and the CMPA[15:0] programmable range is limited to 0 to (2 × CMPB[15:0] - 1).
However, there is no such limitation when T16P is used only for generating PWM pulses with A match interrupt disabled.

17.3.4 PWM Output Condition Settings

Initial output level

The PWM_H*x* and PWM_L*x* output pins go to the initial output level when the pin function is switched for T16P before starting T16P or when T16P is stopped or reset. Use INITOL/T16P_CTL*x* register to select the initial output level.
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.

Note: Before the pin function is switched for T16P, be sure to set INITOL and then reset T16P (set PRESET to 1).

17.4 Control and T16P Operations

17.4.1 Resetting T16P

Writing 1 to PRESET/T16P_CTL*x* register resets T16P. The following operations are performed when PRESET is set to 1.

- The counter (CNT_DATA[15:0]/T16P_CNT_DATA*x* register) is reset to 0x0.
- The B match counter is reset to 0x0.
- The initial volume level (VOLSEL[6:0]/T16P_VOL_CTL*x* register) is loaded into the volume control circuit.
- The compare A and B buffers/registers (CMPA[15:0]/T16P_A*x* register, CMPB[15:0]/T16P_B*x* register) are reset to 0x0.
- The buffer empty flag (BUFEF/T16P_INT*x* register) is set to 1. (No interrupt occurs.)
- All other interrupt flags are reset to 0 and interrupt requests are canceled.
- DMA request is canceled if it has been issued.
- The PWM outputs go to the initial output level set by INITOL/T16P_CTL*x* register.

Note: Be sure to reset T16P before the GPIO pins are switched to the PWM_H*x* and PWM_L*x* pins, and before setting PRUN/T16P_RUN*x* register to 1 to start T16P.

17.4.2 Run/Stop Control

To start T16P, write 1 to PRUN/T16P_RUN*x* register.
T16P must be reset (write 1 to PRESET/T16P_CTL*x* register) before writing 1 to PRUN. Resetting the T16P sets the buffer empty flag to 1, but neither an interrupt request nor a DMA request is issued at this point even if the buffer empty interrupt is enabled. Writing 1 to PRUN enables T16P to issue buffer empty interrupts and DMA requests, so that the first audio data can be sent to the buffer in the interrupt handler routine or DMA.

Note: Writing 1 to PRUN does not actually start T16P, because the buffer is still empty. T16P will start after the buffer is filled by an interrupt or DMA.

To stop T16P being run, write 0 to PRUN. The compare data buffers/registers and counter retain the value at stop. The PWM output is fixed at the level set by INITOL. Note that T16P may not stop counting until B match conditions occur (BCNT[3:0] + 1) times.

17.4.3 Setting Compare Data

Compare A buffer

The compare A buffer (CMPA[15:0]/T16P_A*x* register) is used to specify output pulse widths (duty cycles). Set output audio data to this buffer. The buffer data is loaded to the compare A register when the timer starts counting or when a B match occurs specified number of times, and is compared with the counter value. The output signal level is inverted at the beginning of a pulse period and when the counter reaches the compare data stored in the compare A register. This operation converts audio data set to the compare A buffer into a pulse width.
When the data written to the compare A buffer is loaded to the compare A register, the buffer empty interrupt flag (BUFEF/T16P_INT*x* register) is set to 1 and an interrupt occurs if buffer empty interrupts are enabled.

Also this cause of interrupt can invoke a DMA transfer. By using this interrupt or DMA transfer, the next output data can be set to the compare A buffer.
When the counter reaches the compare data A, the A match interrupt flag (INTAF/T16P_INT*x* register) is set to 1 and an interrupt occurs if A match interrupts are enabled. This type of interrupts does not occur in split mode or when 8-bit PCM data resolution is selected.
The pulse width set by compare A data is as follows:

In normal comparison mode (SELFM/T16P_CTL*x* register = 0)
Output pulse width = CMPA × Count clock cycle

(CMPA: CMPA[15:0] in normal mode, CMPA[15:n] or CMPA[(n-1):0] in split mode)

In fine mode (SELFM = 1)
Output pulse width = CMPA × PCLK cycle × 1/2

(CMPA: CMPA[15:0] in normal mode, CMPA[15:n] or CMPA[(n-1):0] in split mode)

8-bit audio data should be written to CMPA[15:8] in 8-bit size.

Compare B buffer and B match counter

The compare B buffer (CMPB[15:0]/T16P_B*x* register) is used to specify pulse periods. The buffer data is loaded to the compare B register and is compared with the counter value. The output signal level is inverted when the counter reaches the compare data stored in the compare B register (B match). When a B match occurs the counter is reset to 0x0 to start the next pulse period. This operation generates a pulse period according to the compare B data specified.
When the counter reaches the compare data B, the B match interrupt flag (INTBF/T16P_INT*x* register) is set to 1 and an interrupt occurs if B match interrupts are enabled.

The T16P controller includes the B match counter (BCNT[3:0]/T16P_CTL*x* register) to set the sampling rate. Set BCNT[3:0] to 0 to 15. When a B match occurs (BCNT[3:0] + 1) times, the compare A and B buffer data are loaded into the compare A and B registers to start new sampling period.

The pulse period set by compare B data is as follows:

Output pulse period = (CMPB[15:0] + 1) × Count clock cycle

Sampling period = (CMPB[15:0] + 1) × Count clock cycle × (BCNT[3:0] + 1)

17.4.4 Volume Control

T16P includes a volume control function. To use this function, set VOLBPS/T16P_VOL_CTL*x* register to 0. The volume control unit multiplies the PCM data stored in the compare A buffer by the specified volume level set using VOLSEL[6:0]/T16P_VOL_CTL*x* register before loading to the compare A register. This makes it possible to adjust the volume level to 1/64 through 127/64 as well as muting.

Table 17.4.4.1 Volume Level Settings

VOLSEL[6:0]	Volume level
0x7f	× 127/64
0x7e	× 126/64
: 0x40 :	: × 64/64 :
0x2	× 2/64
0x1	× 1/64
0x0	× 0 (mute)

(Default: 0x40)

When VOLBPS is set to 1 (default), the volume control unit is bypassed and compare A data is directly loaded to the compare A register. When 8-bit PCM data is used, the volume control unit should be bypassed by setting VOLBPS to 1.

Table 17.4.4.2 Output Data with or without Volume Control

Input PCM data	Volume control circuit used	Volume control circuit bypassed
Signed 16-bit data: A	A × VOLSEL[6:0] + 0x8000	A + 0x8000
Unsigned 16-bit data: A	A × VOLSEL[6:0]	A

The volume controlled output data is limited to the range from 0x0 to 0xffff for both signed and unsigned PCM data.
When signed 16-bit PCM data is input, the output data becomes 0x0 if the multiplication results are less than -32768 or 0xffff if the multiplication results exceed +32767.
When unsigned 16-bit PCM data is input, the output data becomes 0xffff if the multiplication results exceed +65535.

Note: It is possible to alter the VOLSEL[6:0] values during playing sound, note, however, that set VOLSEL[6:0] before setting PRESET/T16P_CTL*x* register if the first audio data must be output with volume controlled.

17.4.5 Channel Sync Function

To control stereo outputs, T16P provides a channel sync function that allows the software to control two T16P channels simultaneously using the registers for one channel.
Each channel includes CHSYNC/T16P_CTL*x* register to enable or disable the channel sync function.
When CHSYNC is set to 1, the T16P channel operates in sync with another channel. In this case, the control bits listed below become ineffective and the T16P channel is controlled by the registers for another channel.

- VOLBPS/T16P_VOL_CTL*x* register (volume control enable)
- VOLSEL[6:0]/T16P_VOL_CTL*x* register (volume level setting)
- PRUN/T16P_RUN*x* register (timer run/stop)

Table 17.4.5.1 Channel Sync Control

CHSYNC settings		Control	
Ch.0	Ch.1	Ch.0	Ch.1
1	1	Controlled by the Ch.1 registers.	Controlled by the Ch.0 registers.
1	0	Controlled by the Ch.1 registers.	Controlled by the Ch.1 registers.
0	1	Controlled by the Ch.0 registers.	Controlled by the Ch.0 registers.
0	0	Controlled by the Ch.0 registers.	Controlled by the Ch.1 registers.

After an initial reset, the channel sync function is disabled.

Note: The T16P Ch.0 interrupt/DMA request takes precedence over that of Ch.1. When the channel sync function is used for stereo outputs, the T16P Ch.0 interrupt/DMA request will be accepted before Ch.1. Pay attention to the Left and Right data order when creating stereo PCM data transfer routine.

17.4.6 Counter Value

The counter data can be read out from CNT_DATA[15:0]/T16P_CNT_DATA*x* register at any time.
Counter data can also be written to CNT_DATA[15:0]. This makes it possible to change the interrupt and/or timer output cycles temporarily.

17.4.7 Timing Charts

Normal mode

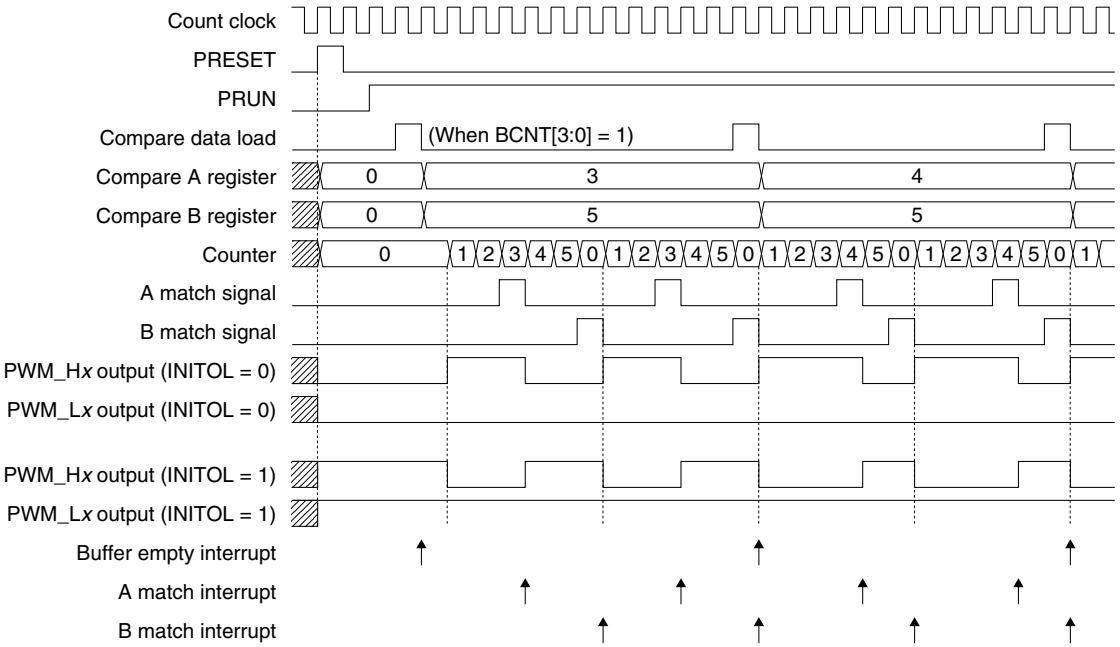


Figure 17.4.7.1 PWM Output Timing Chart 1 (normal mode)

Normal + fine mode

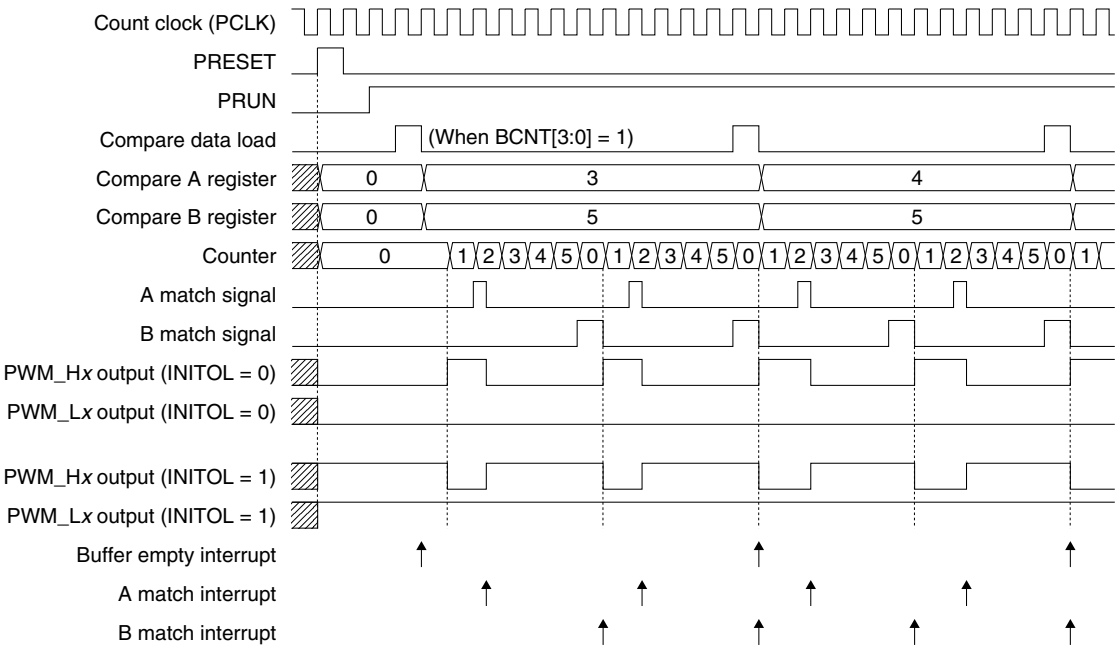


Figure 17.4.7.2 PWM Output Timing Chart 2 (normal + fine mode)

Split mode

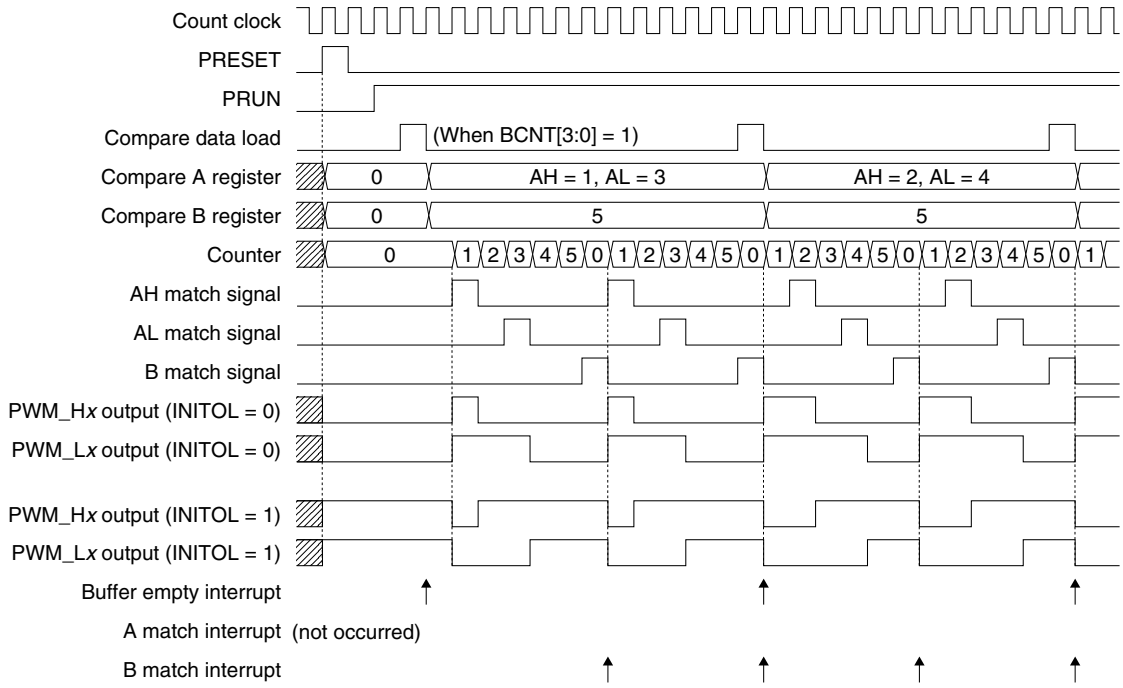


Figure 17.4.7.3 PWM Output Timing Chart 3 (split mode)

Split + fine mode

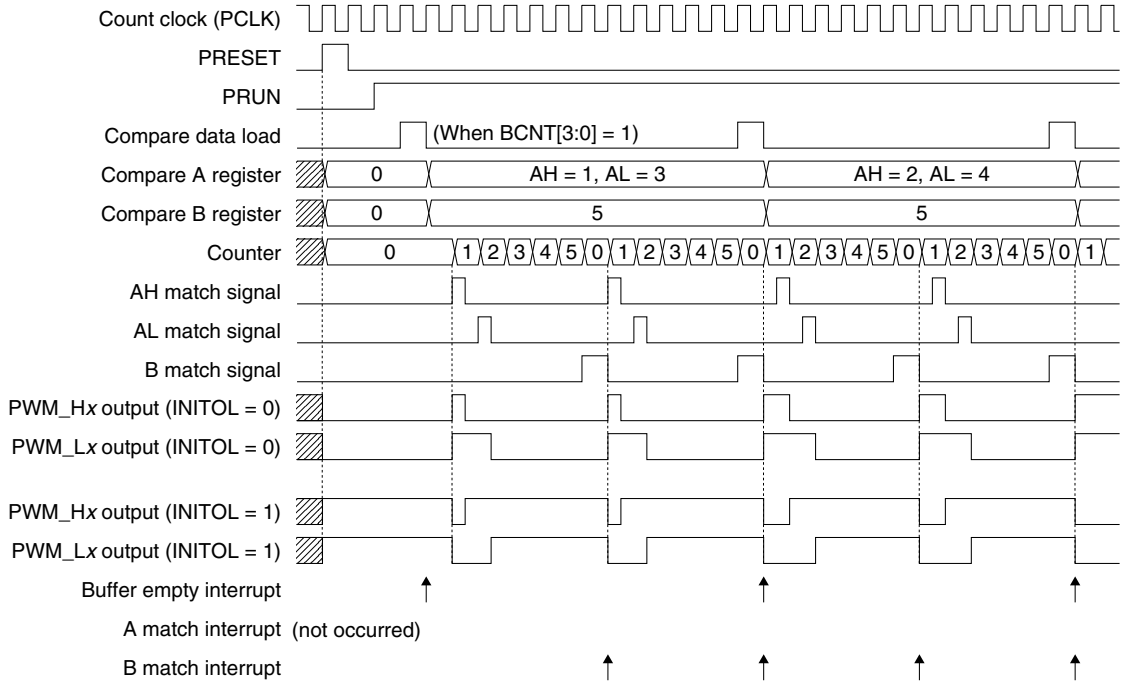


Figure 17.4.7.4 PWM Output Timing Chart 4 (split + fine mode)

17.5 T16P Interrupts and DMA

This section describes the T16P interrupts and invoking DMA.
For more information on interrupt processing and DMA transfer, see the "Interrupt Controller (ITC)" chapter and the "DMA Controller (DMAC)" chapter, respectively.

17.5.1 Interrupts

Each T16P channel can generate the following three kinds of interrupts:
• Buffer empty interrupt
• A match interrupt
• B match interrupt

A T16P channel outputs a single interrupt signal shared by the above interrupt causes to the interrupt controller (ITC). Read the interrupt flags in the T16P channel to identify the interrupt cause that has been occurred.

Buffer empty interrupt

This interrupt request is generated when compare A buffer data are loaded into the compare A registers. It sets the interrupt flag BUFEF/T16P_INT*x* register in the T16P module to 1. The flag is reset by writing 1. Note, however, that the flag will be set to 1 again after resetting if the compare A buffer is still empty. Therefore, write compare data to the compare A buffer before resetting BUFEF.
To use this interrupt, set INTBEEN/T16P_INT*x* register to 1. If INTBEEN is set to 0 (default), interrupt requests for this cause is not sent to the ITC.

Note: Resetting the T16P (writing 1 to PRESET) sets the buffer empty flag to 1, but neither an interrupt request nor a DMA request is issued at this point even if the buffer empty interrupt is enabled. Writing 1 to PRUN enables T16P to issue buffer empty interrupts and DMA requests, so that the first audio data can be sent to the buffer in the interrupt handler routine or DMA.

A match interrupt

This interrupt request is generated when the counter reaches the compare A register value during counting. It sets the interrupt flag INTAF/T16P_INT*x* register in the T16P module to 1.
To use this interrupt, set INTAEN/T16P_INT*x* register to 1. If INTAEN is set to 0 (default), interrupt requests for this cause is not sent to the ITC.
When a split mode or 8-bit PCM data resolution is selected, INTAF will not be set and A match interrupts will not be occurred.

B match interrupt

This interrupt request is generated when the counter reaches the compare B register value during counting. It sets the interrupt flag INTBF/T16P_INT*x* register in the T16P module to 1.
To use this interrupt, set INTBEN/T16P_INT*x* register to 1. If INTBEN is set to 0 (default), interrupt requests for this cause is not sent to the ITC.

If the interrupt flag is set to 1 when the interrupt has been enabled, the T16P module outputs an interrupt request to the ITC. An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied.
For more information on interrupt control registers and the operation when an interrupt occurs, see the "Interrupt Controller (ITC)" chapter.

Notes: • Reset the interrupt flag before enabling interrupts with the interrupt enable bit to prevent occurrence of unwanted interrupt. The interrupt flag is reset by writing 1.

• After an interrupt occurs, the interrupt flag in the T16P module must be reset in the interrupt handler routine.

17.5.2 DMA Transfer

The causes of buffer empty interrupts can invoke a DMA. This allows continuous data transfer via the DMAC between memory and the compare A buffer. The buffer empty interrupt signal is output to both the ITC and DMAC. Therefore, DMA transfer can be performed without generating a T16P interrupt.
For more information on DMA transfer, see the "DMA Controller (DMAC)" chapter.

Note: The T16P Ch.0 interrupt/DMA request takes precedence over that of Ch.1. When the channel sync function is used for stereo outputs, the T16P Ch.0 interrupt/DMA request will be accepted before Ch.1. Pay attention to the Left and Right data order when creating stereo PCM data transfer routine.

17.6 Control Register Details

Table 17.6.1 List of T16P Registers

Address	Register name		Function
0x301200	T16P_A0	T16P Ch.0 Compare A Buffer Register	Compare A data
0x301202	T16P_B0	T16P Ch.0 Compare B Buffer Register	Compare B data
0x301204	T16P_CNT_DATA0	T16P Ch.0 Counter Data Register	Counter data
0x301206	T16P_VOL_CTL0	T16P Ch.0 Volume Control Register	Enable volume control and set volume level
0x301208	T16P_CTL0	T16P Ch.0 Control Register	Set timer operating conditions
0x30120a	T16P_RUN0	T16P Ch.0 Running Control Register	Start/stop timer
0x30120c	T16P_CLK0	T16P Ch.0 Internal Clock Control Register	Select internal count clock
0x30120e	T16P_INT0	T16P Ch.0 Interrupt Control Register	Control T16P interrupts
0x301210	T16P_A1	T16P Ch.1 Compare A Buffer Register	Compare A data
0x301212	T16P_B1	T16P Ch.1 Compare B Buffer Register	Compare B data
0x301214	T16P_CNT_DATA1	T16P Ch.1 Counter Data Register	Counter data
0x301216	T16P_VOL_CTL1	T16P Ch.1 Volume Control Register	Enable volume control and set volume level
0x301218	T16P_CTL1	T16P Ch.1 Control Register	Set timer operating conditions
0x30121a	T16P_RUN1	T16P Ch.1 Running Control Register	Start/stop timer
0x30121c	T16P_CLK1	T16P Ch.1 Internal Clock Control Register	Select internal count clock
0x30121e	T16P_INT1	T16P Ch.1 Interrupt Control Register	Control T16P interrupts

The T16P registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

T16P Ch. *x* Compare A Buffer Registers (T16P_A*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.*x* Compare A Buffer Register (T16P_A*x*)	**0x301200 0x301210** (16 bits)	D15–0	**CMPA[15:0]**	Compare A data CMPA15 = MSB CMPA0 = LSB	0x0 to 0xffff	X	R/W	

D[15:0] CMPA[15:0]: Compare A Data Bits

Sets compare A data (PCM data) to be converted to a pulse width. (Default: undefined)
The buffer data is loaded to the compare A register when the timer starts counting or when a B match occurs specified number of times, and is compared with the counter value. The output signal level is inverted at the beginning of a pulse period and when the counter reaches the compare data stored in the compare A register. This operation converts audio data set to the compare A buffer into a pulse width.
When the data written to the compare A buffer is loaded to the compare A register, the buffer empty interrupt flag (BUFEF/T16P_INT*x* register) is set to 1 and an interrupt occurs if buffer empty interrupts are enabled. Also this cause of interrupt can invoke a DMA transfer. By using this interrupt or DMA transfer, the next output data can be set to the compare A buffer.
When the counter reaches the compare data A, the A match interrupt flag (INTAF/T16P_INT*x* register) is set to 1 and an interrupt occurs if A match interrupts are enabled. This type of interrupts does not occur in split mode or when 8-bit PCM data resolution is selected.
The pulse width set by compare A data is as follows:

In normal comparison mode (SELFM/T16P_CTL*x* register = 0)

Output pulse width = CMPA × Count clock cycle

(CMPA: CMPA[15:0] in normal mode, CMPA[15:n] or CMPA[(n-1):0] in split mode)

In fine mode (SELFM = 1)

Output pulse width = CMPA × PCLK cycle × 1/2

(CMPA: CMPA[15:0] in normal mode, CMPA[15:n] or CMPA[(n-1):0] in split mode)

8-bit audio data should be written to CMPA[15:8] in 8-bit size.

T16P Ch.*x* Compare B Buffer Registers (T16P_B*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.*x* Compare B Buffer Register (T16P_B*x*)	**0x301202** **0x301212** (16 bits)	D15–0	**CMPB[15:0]**	Compare B data CMPB15 = MSB CMPB0 = LSB	0x0 to 0xffff	X	R/W	

D[15:0] CMPB[15:0]: Compare B Data Bits

Sets compare B data to be converted to a pulse period. (Default: undefined)

The buffer data is loaded to the compare B register and is compared with the counter value. The output signal level is inverted when the counter reaches the compare data stored in the compare B register (B match). When a B match occurs the counter is reset to 0x0 to start the next pulse period. This operation generates a pulse period according to the compare B data specified.

When the counter reaches the compare data B, the B match interrupt flag (INTBF/T16P_INT*x* register) is set to 1 and an interrupt occurs if B match interrupts are enabled.

When a B match occurs the number of times (BCNT[3:0] + 1), the compare A and B buffer data are loaded into the compare A and B registers to start new sampling period.

The pulse period set by compare B data is as follows:

Output pulse period = (CMPB[15:0] + 1) × Count clock cycle

Sampling period = (CMPB[15:0] + 1) × Count clock cycle × (BCNT[3:0] + 1)

T16P Ch.*x* Counter Data Registers (T16P_CNT_DATA*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.*x* Counter Data Register (T16P_CNT_DATA*x*)	**0x301204** **0x301214** (16 bits)	D15–0	**CNT_DATA[15:0]**	Counter data CNT_DATA15 = MSB CNT_DATA0 = LSB	0x0 to 0xffff	X	R/W	

D[15:0] CNT_DATA[15:0]: Counter Data Bits

The counter data can be read from this register. (Default: undefined)

Furthermore, data can be set to the counter by writing it to this register.

The counter is reset to 0x0 when a B match occurs or when T16P Ch.*x* is reset by setting PRESET/T16P_CTL*x* register to 1.

T16P Ch.*x* Volume Control Registers (T16P_VOL_CTL*x*)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
T16P Ch.*x* Volume Control Register (T16P_VOL_CTL*x*)	**0x301206** **0x301216** (16 bits)	D15–8	**–**	reserved	–		–	–	0 when being read.
		D7	**VOLBPS**	Volume control enable	1 Disable	0 Enable	1	R/W	Effective only for 16-bit data
		D6–0	**VOLSEL[6:0]**	Volume level select	VOLSEL[6:0]	Volume level	0x40	R/W	
					0x7f	× 127/64			
					0x7e	× 126/64			
					:	:			
					0x40	× 64/64			
					:	:			
					0x2	× 2/64			
					0x1	× 1/64			
					0x0	× 0 (mute)			

Note: This register is ineffective when CHSYNC/T16P_CTL*x* register is set to 1 (channel sync enabled). In this case, the volume level is controlled by the T16P_VOL_CNT*x* register for another channel.

D[15:8] Reserved

D7 VOLBPS: Volume Control Enable Bit

Enables or disables the volume control function.
1 (R/W): Disabled (bypassed) (default)
0 (R/W): Enabled

When VOLBPS is set to 0, the volume control unit multiplies the PCM data stored in the compare A buffer by the specified volume level set using VOLSEL[6:0] before loading to the compare A register. If no volume control is required, set VOLBPS to 1.

Note: When 8-bit PCM data is used, the volume control unit should be bypassed by setting VOLBPS to 1.

D[6:0] VOLSEL[6:0]: Volume Level Select Bits

Selects a volume level when the volume control function is enabled.

Table 17.6.2 Volume Level Settings

VOLSEL[6:0]	Volume level
0x7f	× 127/64
0x7e	× 126/64
:	:
0x40	× 64/64
:	:
0x2	× 2/64
0x1	× 1/64
0x0	× 0 (mute)

(Default: 0x40)

T16P Ch.*x* Control Registers (T16P_CTL*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.*x* Control Register (T16P_CTL*x*)	0x301208 0x301218 (16 bits)	D15–12	BCNT[3:0]	B match count	0x0 to 0xf	0x0	R/W	
		D11	RESSEL	PCM data resolution select	1 16 bits 0 8 bits	1	R/W	
		D10	SGNSEL	PCM data format select	1 Signed 0 Unsigned	1	R/W	
		D9–8	SPLTMD [1:0]	Split mode select	SPLTMD[1:0] Split mode 0x3 10 bits + 6 bits 0x2 9 bits + 7 bits 0x1 8 bits + 8 bits 0x0 Normal (16 bits)	0x0	R/W	Effective only for 16-bit data
		D7	CHSYNC	Channel sync enable	1 Enable 0 Disable	0	R/W	
		D6	SELFM	Fine mode select	1 Fine mode 0 Normal	0	R/W	
		D5	–	reserved	–	–	–	0 when being read.
		D4	INITOL	Initial output level select	1 High 0 Low	0	R/W	
		D3	CLKSEL	Input clock select	1 External 0 Internal	0	R/W	
		D2	–	reserved	–	–	–	0 when being read.
		D1	PRESET	T16P reset	1 Reset 0 Ignored	0	W	
		D0	–	reserved	–	–	–	

D[15:12] BCNT[3:0]: B Match Count Bits

Sets the B match counter. (Default: 0x0)
When a B match occurs (BCNT[3:0] + 1) times, the compare A and B buffer data are loaded into the compare A and B registers.

D11 RESSEL: PCM Data Resolution Select Bit

Selects the PCM data resolution.
1 (R/W): 16 bits (default)
0 (R/W): 8 bits

Notes:
- When 8-bit PCM data resolution is selected, no A match interrupt will be generated.
- 16-bit audio data must be written to addresses 0x301200 and 0x301210 (CMPA[15:0]/T16P_A*x* register) in 16-bit size.

- 8-bit audio data must be written to addresses 0x301201 and 0x301211 (CMPA[15:8]/T16P_A*x* register) in 8-bit size. Furthermore, select 8 bits + 8 bits split mode to use 8-bit audio data (RESSEL = 0). The PWM pulse will be output from the PWM_H*x* pin and the PWM_L*x* pin is fixed at the level set by INITOL. In this case, the PWM_L*x* pin can be used for a GPIO or other functions.

D10 SGNSEL: PCM Data Format Select Bit

Selects the PCM data format.
1 (R/W): Signed data (default)
0 (R/W): Unsigned data

Note: When signed audio data is selected, CMPA15/T16P_A*x* register is treated as the sign bit for both 16-bit and 8-bit audio data.

D[9:8] SPLTMD[1:0]: Split Mode Select Bits

Selects the split mode for manipulating 16-bit PCM data.

Table 17.6.3 Split Mode Selection

SPLTMD[1:0]	Split mode
0x3	10 bits + 6 bits split mode
0x2	9 bits + 7 bits split mode
0x1	8 bits + 8 bits split mode
0x0	16 bits normal mode

(Default: 0x0)

When a split mode is selected, the high-order data bits (10, 9, or 8 high-order bits) and low-order data bits (6, 7, or 8 low-order bits) are compared with the counter data and the two comparison results generate two PWM output signals. The PWM signal generated from the high-order data bits is output from the PWM_H*x* pin and another generated from the low-order data bits is output from the PWM_L*x* pin. When a split mode or 8-bit PCM data resolution is selected, compare A interrupts cannot be generated.

Note: SPLTMD[1:0] does not affect 8-bit PCM data.

D7 CHSYNC: Channel Sync Enable Bit

Enables or disables the channel sync function.
1 (R/W): Enabled
0 (R/W): Disabled (default)

When CHSYNC is set to 1, the T16P channel operates in sync with another channel. In this case, the T16P_VOL_CTL*x* and T16P_RUN*x* registers become ineffective and the T16P channel is controlled by the registers for another channel. When CHSYNC is set to 0, the T16P channel is controlled with the registers in the same channel.

Table 17.6.4 Channel Sync Control

CHSYNC settings		Control	
Ch.0	Ch.1	Ch.0	Ch.1
1	1	Controlled by the Ch.1 registers.	Controlled by the Ch.0 registers.
1	0	Controlled by the Ch.1 registers.	Controlled by the Ch.1 registers.
0	1	Controlled by the Ch.0 registers.	Controlled by the Ch.0 registers.
0	0	Controlled by the Ch.0 registers.	Controlled by the Ch.1 registers.

D6 SELFM: Fine Mode Select Bit

Sets T16P to fine mode.
1 (R/W): Fine mode
0 (R/W): Normal comparison mode (default)

In normal comparison mode, compare A data is compared with the counter data at the rising edge of the count clock. When T16P is set to fine mode, the comparisons are performed at both rising and falling edges of the count clock. At this time the compare A data is halved when compared.

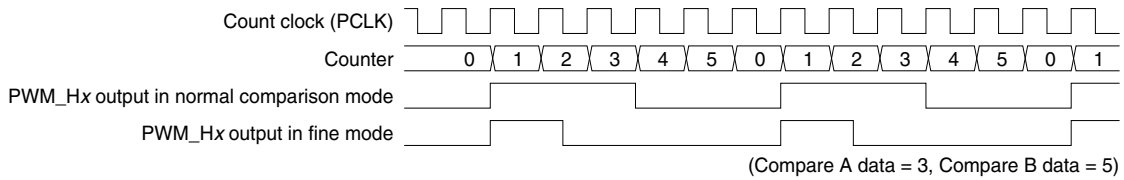


Figure 17.6.1 Fine Mode

The fine mode improves the precision of the pulse width. Note, however, that the PCLK/1 clock can only be used as the count clock in this mode. CLKSEL and CLKDIV[3:0] settings are ineffective.
The fine mode does not affect the pulse period that is determined with compare B data.

Note: When using A match interrupts while T16P is placed into fine mode, the maximum value of CMPB[15:0] is limited to 2^{15} - 1 (= 32,767) and the CMPA[15:0] programmable range is limited to 0 to (2 × CMPB[15:0] - 1).
However, there is no such limitation when T16P is used only for generating PWM pulses with A match interrupt disabled.

D5 Reserved

D4 INITOL: Initial Output Level Select Bit

Selects the initial output level for the PWM_H*x* and PWM_L*x* outputs.
1 (R/W): High
0 (R/W): Low (default)

The PWM_H*x* and PWM_L*x* output pins go to the initial output level when the pin function is switched for T16P before starting T16P or when T16P is stopped or reset. When INITOL is set to 0, the initial output level is set to low. When INITOL is set to 1, the initial output level is set to high.

Note: Before the pin function is switched for T16P, be sure to set INITOL and then reset the PWM (set PRESET to 1).

D3 CLKSEL: Input Clock Select Bit

Selects the input clock for the T16P.
1 (R/W): External clock
0 (R/W): Internal clock (default)

When CLKSEL is set to 0, the internal clock (prescaler output) is selected for the count clock. When CLKSEL is set to 1, an external clock (one that is fed from the EXCL*x* pin) is selected. When using an external clock, the external clock cycle must be at least two CPU operating clock cycles.

D2 Reserved

D1 PRESET: T16P Reset Bit

Resets T16P.
1 (W): Reset
0 (W): Has no effect
0 (R): Always 0 when read (default)

The following operations are performed when PRESET is set to 1.
- The counter (CNT_DATA[15:0]/T16P_CNT_DATA*x* register) is reset to 0x0.
- The B match counter is reset to 0x0.
- The initial volume level (VOLSEL[6:0]/T16P_VOL_CTL*x* register) is loaded into the volume control circuit.
- The compare A and B buffers/registers (CMPA[15:0]/T16P_A*x* register, CMPB[15:0]/T16P_B*x* register) are reset to 0x0.
- The buffer empty flag (BUFEF/T16P_INT*x* register) is set to 1. (No interrupt occurs.)
- All other interrupt flags are reset to 0 and interrupt requests are canceled.
- DMA request is canceled if it has been issued.
- The PWM outputs go to the initial output level set by INITOL register.

Note: Be sure to reset T16P before the GPIO pins are switched to the PWM_H*x* and PWM_L*x* pins, and before setting PRUN/T16P_RUN*x* register to 1 to start T16P.

D0 Reserved

T16P Ch.*x* Running Control Registers (T16P_RUN*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.*x* Running Control Register (T16P_RUN*x*)	0x30120a 0x30121a (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	PRUN	Timer run/stop control	1 Run 0 Stop	0	R/W	

Note: This register is ineffective when CHSYNC/T16P_CTL*x* register is set to 1 (channel sync enabled). In this case, the timer run/stop is controlled by the T16P_RUN*x* register for another channel.

D[15:1] Reserved

D0 PRUN: Timer Run/Stop Control Bit

Starts and stops T16P.
1 (R/W): Run
0 (R/W): Stop (default)

To start T16P, write 1 to PRUN.
T16P must be reset (write 1 to PRESET/T16P_CTL*x* register) before writing 1 to PRUN. Resetting the T16P sets the buffer empty flag to 1, but neither an interrupt request nor a DMA request is issued at this point even if the buffer empty interrupt is enabled. Writing 1 to PRUN enables T16P to issue buffer empty interrupts and DMA requests, so that the first audio data can be sent to the buffer in the interrupt handler routine or DMA.
To stop T16P being run, write 0 to PRUN. The compare data buffers/registers and counter retain the value at stop. The PWM output is fixed at the level set by INITOL. Note that T16P may not stop counting until B match conditions occur (BCNT[3:0] + 1) times.

T16P Ch.*x* Internal Clock Control Registers (T16P_CLK*x*)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
T16P Ch.*x* Internal Clock Control Register (T16P_CLK*x*)	0x30120c 0x30121c (16 bits)	D15–4	–	reserved	–		–	–	0 when being read.
		D3–0	CLKDIV [3:0]	Counter clock division ratio select (Prescaler output clock)	CLKDIV[3:0]	Division ratio	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
					0xf–0xd	reserved			
					0xc	1/4096			
					0xb	1/2048			
					0xa	1/1024			
					0x9	1/512			
					0x8	1/256			
					0x7	1/128			
					0x6	1/64			
					0x5	1/32			
					0x4	1/16			
					0x3	1/8			
					0x2	1/4			
					0x1	1/2			
					0x0	1/1			

D[15:4] Reserved

D[3:0] CLKDIV[3:0]: Counter Clock Division Ratio Select Bits

Selects the counter clock (PCLK division ratio) from the 13 different prescaler output clocks when an internal clock is used.

Table 17.6.5 Internal Clock (PCLK Division Ratio) Selection

CLKDIV[3:0]	Division ratio	CLKDIV[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	Reserved	0x6	1/64
0xd	Reserved	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Notes: • Make sure the counter is halted before setting the count clock.

• When T16P is set to fine mode, CLKDIV[3:0] is ineffective and PCLK is directly used as the count clock.

T16P Ch.*x* Interrupt Control Registers (T16P_INT*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.*x* Interrupt Control Register (T16P_INT*x*)	0x30120e 0x30121e (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10	**BUFEF**	Buffer empty interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	1	R/W	Reset by writing 1.
		D9	**INTBF**	B match interrupt flag		0	R/W	
		D8	**INTAF**	A match interrupt flag		0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2	**INTBEEN**	Buffer empty interrupt enable	1 Enable / 0 Disable	0	R/W	
		D1	**INTBEN**	B match interrupt enable	1 Enable / 0 Disable	0	R/W	
		D0	**INTAEN**	A match interrupt enable	1 Enable / 0 Disable	0	R/W	

D[15:11] Reserved

D10 BUFEF: Buffer Empty Interrupt Flag Bit

Indicates whether the cause of buffer empty interrupt has occurred or not.

1 (R): Cause of interrupt has occurred (default)
0 (R): No cause of interrupt has occurred
1 (W): Flag is reset
0 (W): Ignored

BUFEF is a T16P interrupt flag that is set to 1 when the compare A buffer data is loaded into the compare A register. BUFEF is reset by writing 1. Note, however, that the flag will be set to 1 again after resetting if the compare A buffer is still empty. Therefore, write compare data to the compare A buffer before resetting BUFEF.

The BUFEF value is undefined at initial reset. However, no buffer empty interrupt request is issued until T16P starts running by setting PRUN/T16P_RUN*x* register is set to 1 even if the interrupt is enabled when BUFEF has been set to 1 at initial reset .

D9 INTBF: B match Interrupt Flag Bit

Indicates whether the cause of B match interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

INTBF is a T16P interrupt flag that is set to 1 when the counter reaches the value set in the compare B register. INTBF is reset by writing 1.

D8 INTAF: A match Interrupt Flag Bit

Indicates whether the cause of A match interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

INTAF is a T16P interrupt flag that is set to 1 when the counter reaches the value set in the compare A register. INTAF is reset by writing 1.

D[7:3] Reserved

D2 INTBEEN: Buffer Empty Interrupt Enable Bit

Enables or disables buffer empty interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting INTBEEN to 1 enables buffer empty interrupt requests to the ITC. Setting it to 0 disables interrupts.

D1 INTBEN: B Match Interrupt Enable Bit

Enables or disables B match interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting INTBEN to 1 enables B match interrupt requests to the ITC. Setting it to 0 disables interrupts.

D0 INTAEN: A Match Interrupt Enable Bit

Enables or disables A match interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting INTAEN to 1 enables A match interrupt requests to the ITC. Setting it to 0 disables interrupts.

18 Watchdog Timer (WDT)

18.1 WDT Module Overview

The S1C33L27 incorporates a watchdog timer to detect the CPU running uncontrollably.
The features of WDT are listed below.

- 30-bit up counter with a comparator
- Reset or NMI can be generated when the counter reaches the specified value if WDT has not been reset.
- The count clock is selectable from the system clock (PCLK) and an external clock (WDT_EXCL).
- Can output the generated NMI signal (#WDT_NMI) and the comparator output (WDT_CLK).

Figure 18.1.1 shows the WDT configuration.

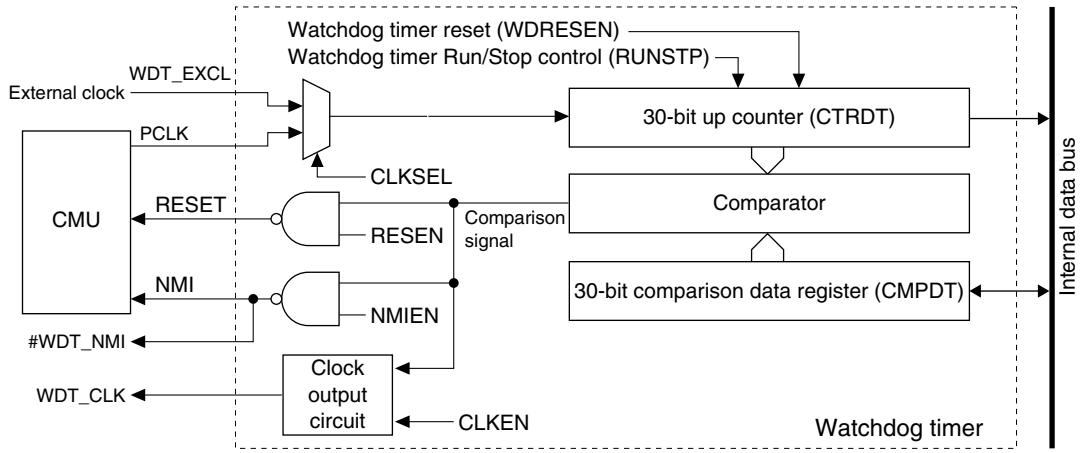


Figure 18.1.1 WDT Configuration

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 PCLK clock (= system clock) or the external clock input to WDT_EXCL 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.

18.2 WDT Input/Output Pins

Table 18.2.1 lists the input/output pins for the WDT module.

Table 18.2.1 List of WDT Pins

Pin name	I/O	Qty	Function
WDT_EXCL	I	1	WDT external clock input pin Inputs an external clock as the counter clock.
WDT_CLK	O	1	Watchdog timer clock output pin Outputs the reset/NMI cycle clock generated in the watchdog timer to external devices.
#WDT_NMI	O	1	Watchdog timer NMI output pin Outputs the NMI signal generated in the watchdog timer to external devices.

The WDT input/output pins (WDT_EXCL, WDT_CLK, #WDT_NMI) are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as WDT input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

18.3 WDT Operating Clock

The watchdog timer module is clocked by the PCLK clock (= system clock) supplied from the CMU. At initial reset, this clock is also selected as the count clock for the watchdog timer.
For more information on clock generation and control, see the "Clock Management Unit (CMU)" chapter.

Note: Even when using an external clock as the count clock for the watchdog timer, PCLK is required for watchdog timer operation and access to its control register.

18.4 Control of the Watchdog Timer

18.4.1 Setting Up the Watchdog Timer

Selecting the count clock

The internal clock (PCLK) or an external clock (WDT_EXCL) can be selected as the count clock for the 30-bit up-counter by using CLKSEL/WD_EN register.
Setting CLKSEL to 0 (default) selects the internal clock (PCLK); setting it to 1 selects an external clock (WDT_EXCL).

Setting the NMI/reset generation cycle

The watchdog timer has a 30-bit comparison data register (CMPDT[29:0]/WD_CMP_L/H registers) that can be used to set a cycle in which to generate an NMI or reset signal.
The data set to CMPDT[29:0] 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.

$$\text{NMI generating cycle} = \frac{\text{CMPDT} + 1}{f_{WDTIN}} \text{ [s]}$$

where
CMPDT = value set to CMPDT[29:0]
f_{WDTIN} = Input clock (PCLK or WDT_EXCL) frequency [Hz]

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/WD_EN register to 1. To output a reset signal instead, set RESEN/WD_EN register to 1.
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 WD_EN, WD_CMP_L, and WD_CMP_H 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 WDPTC[15:0]/WD_PROTECT register in 16-bit access only. Once the registers are rewritten, be sure to write other than 0x96 to WDPTC[15:0] to reapply write protection.

18.4.2 Starting/Stopping the Watchdog Timer

Writing 1 to RUNSTP/WD_EN register starts counting by the watchdog timer; writing 0 stops the watchdog timer. Since RUNSTP exists in the write-protected WD_EN register, write protection must be removed by writing 0x96 to WDPTC[15:0]/WD_PROTECT register before the content of RUNSTP can be altered.

18.4.3 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/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.
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 count value of the up-counter can be read out from CTRDT[29:0]/WD_CNT_L/H registers at any time.

18.4.4 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/WD_EN register or RESEN/WD_EN register to 0. Otherwise, write 0 to RUNSTP/WD_EN register to stop the watchdog timer before executing the halt instruction. When NMIEN or RESEN 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 PCLK 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/WD_EN register or RESEN/WD_EN register as required.

18.4.5 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/WD_EN register to 1 after setting up the WDT_CLK pin.
Since CLKEN also exists in the write-protected WDT_EN register, write protection must be removed by writing 0x96 to WDPTC[15:0]/WD_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.

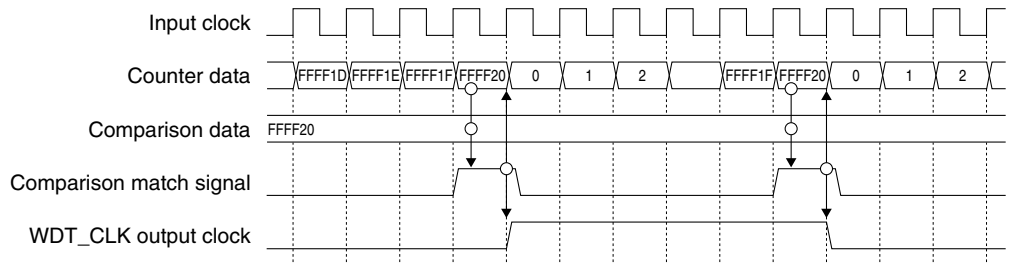


Figure 18.4.5.1 Clock Output of Watchdog Timer

18.4.6 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. Setting NMIEN/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 input clock cycles.

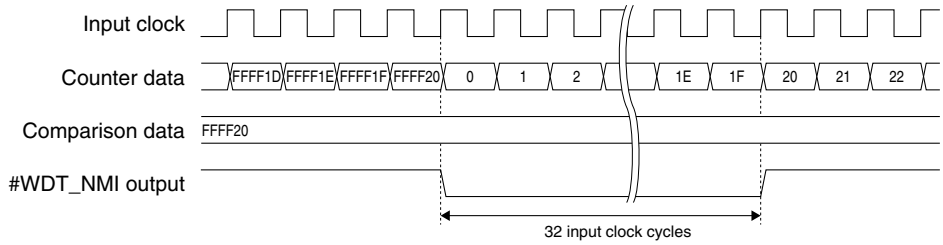


Figure 18.4.6.1 External NMI Output

18.5 Control Register Details

Table 18.5.1 List of WDT Registers

Address	Register name		Function
0x301000	WD_PROTECT	WDT Write Protect Register	Enable WDT register write protection
0x301002	WD_EN	WDT Enable and Setup Register	Configure and start watchdog timer
0x301004	WD_CMP_L	WDT Comparison Data L Register	Comparison data
0x301006	WD_CMP_H	WDT Comparison Data H Register	
0x301008	WD_CNT_L	WDT Count Data L Register	Watchdog timer counter data
0x30100a	WD_CNT_H	WDT Count Data H Register	
0x30100c	WD_CTL	WDT Control Register	Reset watchdog timer

The following describes each WDT register.

Notes: • When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

• The WD_PROTECT register (0x301000) allows 16-bit access only. Other registers (0x301002 to 0x30100c) allow 8-bit access as well as 16-bit access.

WDT Write Protect Register (WD_PROTECT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Write Protect Register (WD_ PROTECT)	**0x301000** (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 (0x301002–0x301006). 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 0x301002 to 0x301006.

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_EN, WDT_CMP_L, or WDT_CMP_H register, write 0x96 to WDPTC[15:0] to remove write protection. Setting WDPTC[15:0] 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 WDPTC[15:0], said registers can be rewritten any number of times until WDPTC[15:0] is set to other than 0x96. When the WDT_EN, WDT_CMP_L, or WDT_CMP_H have been rewritten, be sure to write other than 0x96 to WDPTC[15:0] to prevent erroneous writing to the registers.

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)	0x301002 (16 bits)	D15–7	–	reserved	–	–	–	0 when being read.
		D6	CLKSEL	WDT input clock select	1 External clk 0 Internal clk	0	R/W	Write-protected
		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	Write-protected
		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 WDPTC[15:0]/WD_PROTECT register. Once the register has been rewritten, be sure to write other than 0x96 to WDPTC[15:0] 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 (WDT_EXCL)
0 (R/W): Internal clock (PCLK) (default)

Setting this bit to 0 (default) selects the internal clock (PCLK); setting it to 1 selects the external clock (WDT_EXCL).

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.

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.

WDT Comparison Data L/H Registers (WD_CMP_L, WD_CMP_H)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Comparison Data L Register (WD_CMP_L)	**0x301004** (16 bits)	D15–0	**CMPDT [15:0]**	WDT comparison data CMPDT0 = LSB	0x0 to 0x3fffffff (low-order 16 bits)	0x0	R/W	Write-protected
WDT Comparison Data H Register (WD_CMP_H)	**0x301006** (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	Write-protected

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 WDPTC[15:0]/WD_PROTECT register. Once the registers have been rewritten, be sure to write other than 0x96 to WDPTC[15:0] to reapply write protection.

D[13:0]/0x301006, D[15:0]/0x301004

CMPDT[29:0]: WDT Comparison Data Bits

Sets comparison data. (Default: 0x0)

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.

Note: Do not set a value equal to or less than 0x1f as comparison data.

WDT Count Data L/H Registers (WD_CNT_L, WD_CNT_H)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Count Data L Register (WD_CNT_L)	**0x301008** (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)	**0x30100a** (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	

D[13:0]/0x30100a, D[15:0]/0x301008

CTRDT[29:0]: WDT Counter Data Bits

The current count value of the 30-bit up-counter can be read out from these registers.

(Default: indeterminate)

WDT Control Register (WD_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Control Register (WD_CTL)	**0x30100c** (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.

19 UART

19.1 UART Module Overview

The S1C33L27 incorporates a UART module. The UART transfers asynchronous data with external serial devices. The following shows the main features of the UART:

- Transfer rate: 150 to 460,800 bps (150 to 115,200 bps in IrDA mode)
- Transfer clock: Internal clock (T8F Ch.5 output) or an external clock input (SCLK input) can be selected.
- Character length: 7 or 8 bits (LSB first)
- Parity mode: Even, odd, or no parity
- Stop bit: 1 or 2 bits
- Start bit: 1 bit fixed
- Supports full-duplex communication.
- Includes a 2-byte receive data buffer and a 1-byte transmit data buffer.
- Includes an RZI modulator/demodulator circuit to support IrDA 1.0-compatible infrared communications.
- Can detect parity error, framing error, and overrun error during receiving.
- Can generate receive buffer full, transmit buffer empty and receive error interrupts.

Figure 19.1.1 shows the UART configuration.

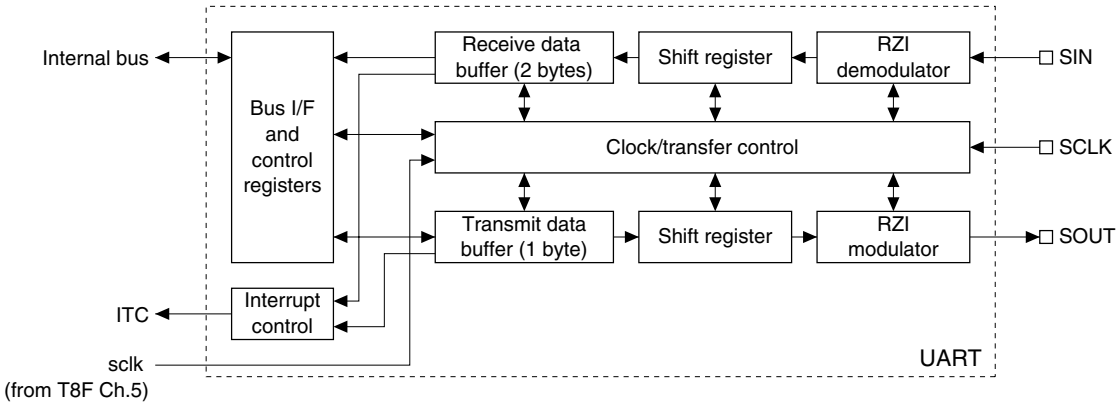


Figure 19.1.1 UART Configuration

19.2 UART Input/Output Pins

Table 19.2.1 lists the UART input/output pins.

Table 19.2.1 List of UART Pins

Pin name	I/O	Qty	Function
SIN	I	1	UART data input pin Inputs serial data sent from an external serial device.
SOUT	O	1	UART data output pin Outputs serial data sent to an external serial device.
SCLK	I	1	UART clock input pin Inputs the transfer clock when an external clock is used.

The UART input/output pins (SIN, SOUT, SCLK) are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as UART input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

19.3 Transfer Clock

The UART transfer clock can be set to internal or external using SSCK/UART_MOD register.

Note: Make sure the UART is halted (RXEN/UART_CTL register = 0) before changing SSCK.

Internal clock

Setting SSCK to 0 (default) selects the internal clock. The UART uses the T8F Ch.5 output clock. Thus, T8F Ch.5 must be programmed to output a clock suited to the transfer rate. For more information on controlling T8F Ch.5, see the "Fine Mode 8-bit Timers (T8F)" chapter.

External clock

Setting SSCK to 1 selects an external clock. In this case, input the external clock to the SCLK pin.

Notes: • The UART generates a sampling clock by dividing the T8F Ch.5 output clock or external clock by 16. Be careful when setting the transfer rate.

Example: Transfer rate = 115,200 bps, system clock = 48 MHz, DF[3:0]/T8F_CLK5 register setting (T8F Ch.5) = 1/1.

$f_{SOURCE} = f_{SYS\_CLK} \times DF / (TR + 1)$

$bps = f_{SOURCE} / 16$

$TR = (f_{SYS\_CLK} \times DF) / (bps \times 16) - 1 = 48{,}000{,}000 \times 1 / (115{,}200 \times 16) - 1 = 25$

f_{SOURCE}: T8F Ch.5 output clock frequency [Hz]
f_{SYS_CLK}: System clock frequency [Hz]
bps: Transfer rate [bps]
DF: Division ratio set by DF[3:0]/T8F_CLK5 register (T8F Ch.5)
TR: Reload data to be set to the T8F_TR5 register (T8F Ch.5)

• When inputting an external clock via the SCLK pin, the clock frequency must be half of the system clock or less and has a duty ratio of 50%.

19.4 Transfer Data Settings

Set the following conditions to configure the transfer data format.
• Data length: 7 or 8 bits
• Start bit: Fixed at 1 bit
• Stop bit: 1 or 2 bits
• Parity bit: Even, odd, or no parity

Note: Make sure the UART is halted (RXEN/UART_CTL register = 0) before changing transfer data format settings.

Data length

The data length is selected by CHLN/UART_MOD register. Setting CHLN to 0 (default) configures the data length to 7 bits. Setting CHLN to 1 configures it to 8 bits.

Stop bit

The stop bit length is selected by STPB/UART_MOD register. Setting STPB to 0 (default) configures the stop bit length to 1 bit. Setting STPB to 1 configures it to 2 bits.

Parity bit

Whether the parity function is enabled or disabled is selected by PREN/UART_MOD register. Setting PREN to 0 (default) disables the parity function. In this case, no parity bit is added to the transfer data and the data is not checked for parity when received. Setting PREN to 1 enables the parity function. In this case, a parity bit is added to the transfer data and the data is checked for parity when received.

When the parity function is enabled, the parity mode is selected by PMD/UART_MOD register. Setting PMD to 0 (default) adds a parity bit and checks for even parity. Setting PMD to 1 adds a parity bit and checks for odd parity.

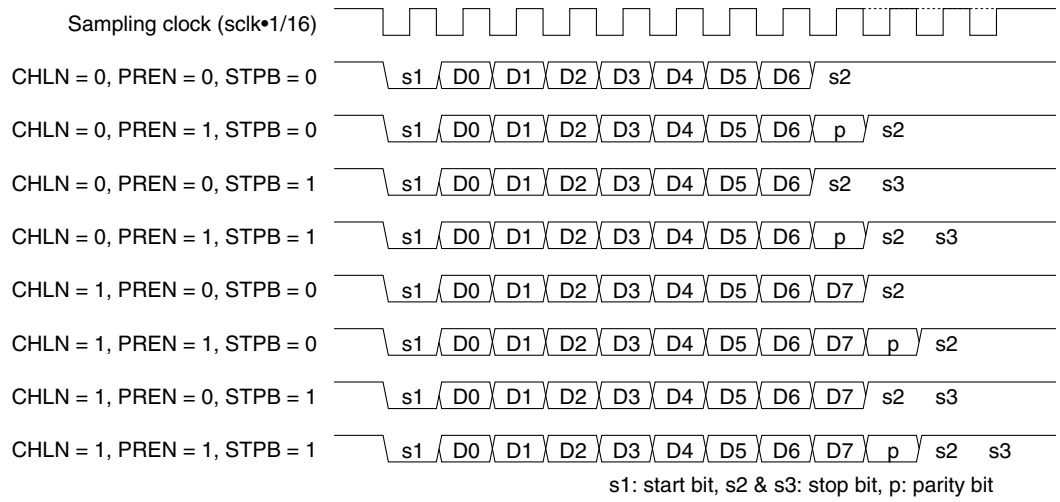


Figure 19.4.1 Transfer Data Format

19.5 Data Transfer Control

Make the following settings before starting data transfers.

(1) Select the input clock. (See Section 19.3.)
To use the internal clock, program T8F Ch.5 to output the transfer clock. See the T8F chapter.
(2) Set the transfer data format. (See Section 19.4.)
(3) To use the IrDA interface, set IrDA mode. (See Section 19.8.)
(4) Set interrupt conditions to use UART interrupts. (See Section 19.7.)

Note: Make sure the UART is halted (RXEN/UART_CTL register = 0) before changing the above settings.

Enabling data transfers

Set RXEN/UART_CTL register to 1 to enable data transfers. This puts the transmitter/receiver circuit in ready-to-transmit/receive status.

Notes: • Do not set RXEN to 0 while the UART is sending or receiving data.

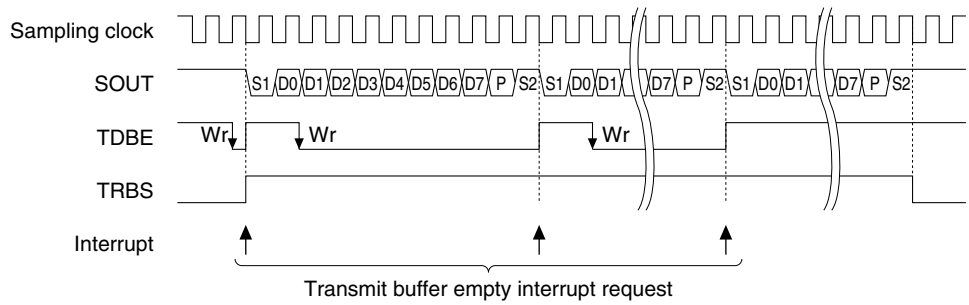
• Data transfer only for transmission or reception cannot be enabled.

Data transmission control

To start data transmission, write the transmit data to TXD[7:0]/UART_TXD register.
The data is written to the transmit data buffer, and the transmitter circuit starts sending data.
The buffer data is sent to the transmit shift register, and the start bit is output from the SOUT pin. The data in the shift register is then output from the LSB. The transfer data bit is shifted in sync with the sampling clock rising edge and output in sequence via the SOUT pin. Following output of MSB, the parity bit (if parity is enabled) and the stop bit are output.

The transmitter circuit includes two status flags: TDBE/UART_ST register and TRBS/UART_ST register.
The TDBE flag indicates the transmit data buffer status. This flag switches to 0 when the application program writes data to the transmit data buffer and reverts to 1 when the buffer data is sent to the transmit shift register. An interrupt can be generated when this flag is set to 1 (see Section 19.7). Subsequent data is sent after confirming that the transmit data buffer is empty either by using this interrupt or by reading the TDBE flag. The transmit data buffer size is 1 byte, but a transmit data buffer and a shift register are provided separately to allow data to be written while the previous data is being sent. Always confirm that the transmit data buffer is empty before writing transmit data. Writing data while the TDBE flag is 0 will overwrite earlier transmit data inside the transmit data buffer.

The TRBS flag indicates the shift register status. This flag switches to 1 when transmit data is loaded from the transmit data buffer to the shift register and reverts to 0 once the data is sent. Read this flag to check whether the transmitter circuit is operating or at standby.



S1: Start bit, S2: Stop bit, P: Parity bit, Wr: Data write to transmit data buffer

Figure 19.5.1 Data Transmission Timing Chart

Data reception control

When the external serial device sends a start bit, the receiver circuit detects its Low level and starts sampling the following data bits. The data bits are sampled at the sampling clock rising edge, and the lead bit is loaded into the receive shift register as LSB. Once the MSB has been received into the shift register, the received data is loaded into the receive data buffer. If parity checking is enabled, the receiver circuit checks the received data at the same time by checking the parity bit received immediately after the MSB.
The receive data buffer, a 2-byte FIFO, receives data until full.

Received data in the buffer can be read from RXD[7:0]/UART_RXD register. The oldest data is read out first and data is cleared by reading.

The receiver circuit includes two buffer status flags: RDRY/UART_ST register and RD2B/UART_ST register. The RDRY flag indicates that the receive data buffer still contains data. The RD2B flag indicates that the receive data buffer is full.

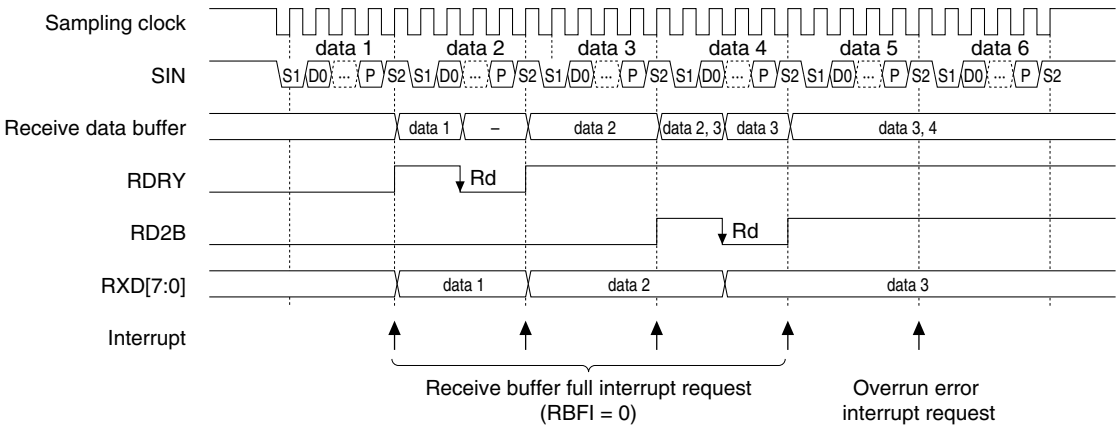
(1) RDRY = 0, RD2B = 0
The receive data buffer contents need not be read, since no data has been received.

(2) RDRY = 1, RD2B = 0
One 8-bit data has been received. Read the receive data buffer contents once. This clears the data inside the buffer and resets the RDRY flag. The buffer reverts to state (1) above.
If the receive data buffer contents are read twice, the second data read will be invalid.

(3) RDRY = 1, RD2B = 1
Two 8-bit data have been received. Read the receive data buffer contents twice. The receive data buffer outputs the oldest data first, clearing the buffer data read out and resetting the RD2B flag. The buffer then reverts to the state in (2) above. The second read outputs the most recent received data, after which the buffer reverts to the state in (1) above.
Even when the receive data buffer is full, the shift register can receive one more 8-bit data. An overrun error will occur if additional data is sent from the external serial device in this status, and the new data will overwrite the shift register data. The contents of the receive data buffer must be read out before an overrun error occurs. For detailed information on overrun errors, refer to Section 19.6.

The volume of data received can be checked by reading these flags.
The UART allows receive buffer full interrupts to be generated once data has been received in the receive data buffer. These interrupts can be used to read the receive data buffer. By default, a receive buffer full interrupt occurs when the receive data buffer receives one 8-bit data (status (2) above). This can be changed by setting RBFI/UART_CTL register to 1 so that an interrupt occurs when the receive data buffer receives two 8-bit data.

Three error flags are also provided in addition to the flags previously mentioned. See Section 19.6 for detailed information on flags and receive errors.



S1: Start bit, S2: Stop bit, P: Parity bit, Rd: Data read from RXD[7:0]

Figure 19.5.2 Data Receiving Timing Chart

Disabling data transfers

After a data transfer is completed (both transmission and reception), write 0 to RXEN to disable data transfers. Make sure that the TDBE flag is 1 and the TRBS and RDRY flags are both 0 before disabling data transfer.

Setting RXEN to 0 empties the transmit data buffer, clearing any remaining data. The receive data buffer is not cleared. The data being transferred cannot be guaranteed if RXEN is set to 0 while data is being sent or received.

19.6 Receive Errors

Three different receive errors may be detected while receiving data.

Since receive errors are interrupt causes, they can be processed by generating interrupts. For more information on UART interrupt control, see Section 19.7.

Parity error

If PREN/UART_MOD register has been set to 1 (parity enabled), data received is checked for parity.

Data received in the shift register is checked for parity when sent to the receive data buffer. The matching is checked against the PMD/UART_MOD register setting (odd or even parity). If the result is a non-match, a parity error is issued, and the parity error flag PER/UART_ST register is set to 1. Even if this error occurs, the data received is sent to the receive data buffer, and the receiving operation continues. However, the received data cannot be guaranteed if a parity error occurs. The PER flag is reset to 0 by writing 1.

Framing error

A framing error occurs if the stop bit is received as 0 and the UART determines loss of sync. If the stop bit is set to two bits, only the first bit is checked.

The framing error flag FER/UART_ST register is set to 1 if this error occurs. The received data is still transferred to the receive data buffer if this error occurs and the receiving operation continues, but the data cannot be guaranteed, even if no framing error occurs for subsequent data receiving. The FER flag is reset to 0 by writing 1.

Overrun error

Even if the receive data buffer is full (two 8-bit data already received), the third data can be received in 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 an overrun error will occur. If an overrun error occurs, the overrun error flag OER/UART_ST register is set to 1. The receiving operation continues even if this error occurs. The OER flag is reset to 0 by writing 1. Furthermore, set RXEN to 0 and read the receive data buffer successively until the RDRY flag is cleared.

19.7 UART Interrupts

The UART includes a function for generating the following three different types of interrupts.
- Transmit buffer empty interrupt
- Receive buffer full interrupt
- Receive error interrupt

The UART outputs one interrupt signal shared by the three above interrupt causes to the interrupt controller (ITC). Inspect the status flag and error flag to determine the interrupt cause occurred.

Transmit buffer empty interrupt

To use this interrupt, set TIEN/UART_CTL register to 1. If TIEN is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When transmit data written to the transmit data buffer is transferred to the shift register, the UART sets TDBE/UART_ST register to 1, indicating that the transmit data buffer is empty. If transmit buffer empty interrupts are enabled (TIEN = 1), an interrupt request is sent simultaneously to the ITC.

An interrupt occurs if other interrupt conditions are met. You can inspect the TDBE flag in the UART interrupt handler routine to determine whether the UART interrupt is attributable to a transmit buffer empty. If TDBE is 1, the next transmit data can be written to the transmit data buffer by the interrupt handler routine.

Receive buffer full interrupt

To use this interrupt, set RIEN/UART_CTL register to 1. If RIEN is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

If the specified volume of received data is loaded into the receive data buffer when a receive buffer full interrupt is enabled (RIEN = 1), the UART outputs an interrupt request to the ITC. If RBFI/UART_CTL register is 0, an interrupt request is output as soon as one received data is loaded into the receive data buffer (when RDRY/UART_ST register is set to 1). If RBFI is 1, an interrupt request is output as soon as two received data are loaded into the receive data buffer (when RD2B/UART_ST register is set to 1).

An interrupt occurs if other interrupt conditions are met. You can inspect the RDRY and RD2B flags in the UART interrupt handler routine to determine whether the UART interrupt is attributable to a receive buffer full. If RDRY or RD2B is 1, the received data can be read from the receive data buffer by the interrupt handler routine.

Receive error interrupt

To use this interrupt, set REIEN/UART_CTL register to 1. If REIEN is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

The UART sets an error flag, PER, FER, or OER/UART_ST register to 1 if a parity error, framing error, or overrun error is detected when receiving data. If receive error interrupts are enabled (REIEN = 1), an interrupt request is sent simultaneously to the ITC.

If other interrupt conditions are satisfied, an interrupt occurs. You can inspect the PER, FER, and OER flags in the UART interrupt handler routine to determine whether the UART interrupt was caused by a receive error. If any of the error flags has the value 1, the interrupt handler routine will proceed with error recovery.

For more information on interrupt processing, see the “Interrupt Controller (ITC)” chapter.

19.8 IrDA Interface

This UART module incorporates an RZI modulator/demodulator circuit enabling implementation of IrDA 1.0-compatible infrared communication function simply by adding basic external circuits.

The transmit data output from the UART transmit shift register is input to the modulator circuit and output from the SOUT pin after the Low pulse has been modulated to a 3/16 sclk cycle.

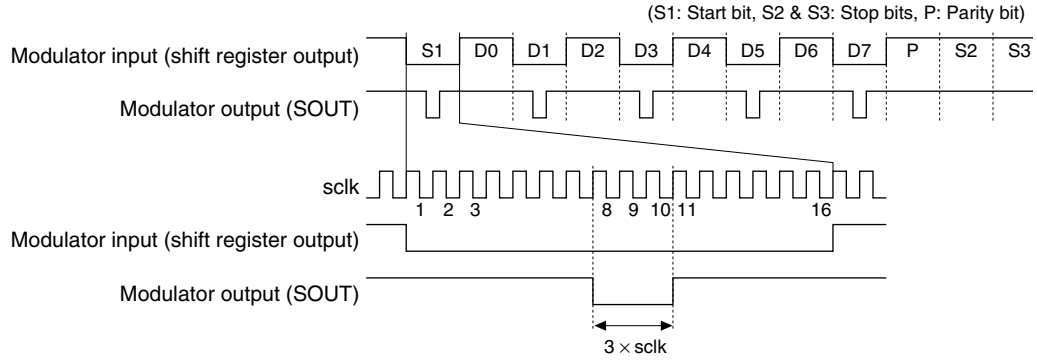


Figure 19.8.1 Transmission Signal Waveform

The received IrDA signal is input to the demodulator circuit and the Low pulse width is converted to 16 sclk cycles before entry to the receive shift register. The demodulator circuit uses the pulse detection clock selected from the prescaler output clock separately from the transfer clock to detect Low pulses input (when minimum pulse width = 1.41 µs/115,200 bps).

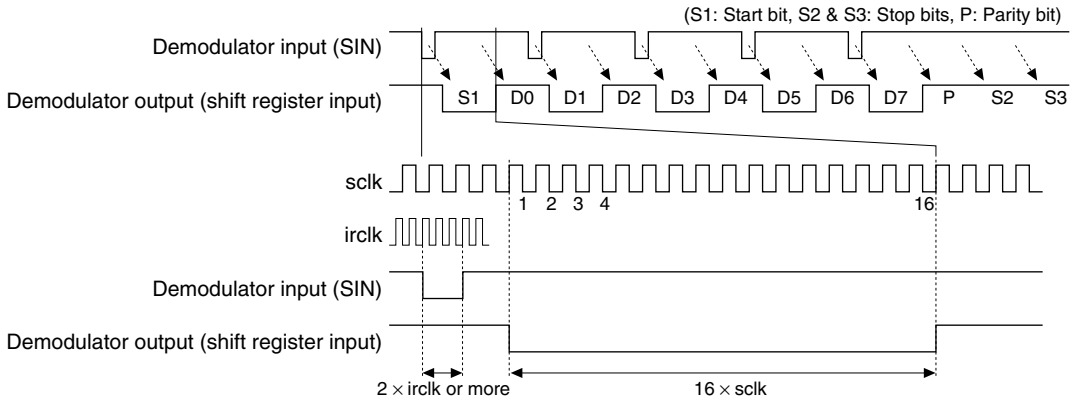


Figure 19.8.2 Receive Signal Waveform

IrDA enable

To use the IrDA interface function, set IRMD/UART_EXP register to 1. This enables the RZI modulator/demodulator circuit.

Note: This setting must be performed before setting other UART conditions.

IrDA receive detection clock selection

The input pulse detection clock is selected from among the prescaler output clock PCLK/1 to PCLK/128 using IRCLK[2:0]/UART_EXP register.

Table 19.8.1 IrDA Receive Detection Clock (PCLK Division Ratio) Selection

IRCLK[2:0]	Prescaler output clock
0x7	1/128
0x6	1/64
0x5	1/32
0x4	1/16
0x3	1/8
0x2	1/4
0x1	1/2
0x0	1/1

(Default: 0x0)

This clock must be selected as a clock faster than the T8F Ch.5 output clock or transfer clock sclk input via the SCLK pin.

The demodulator circuit treats Low pulses with a width of at least two IrDA receive detection clock cycles as valid and converts them to 16 sclk cycle width Low pulses. Select the prescaler output clock to enable detection of input pulses with a minimum width of 1.41 µs.

Serial data transfer control

Data transfer control in IrDA mode is identical to that for normal interfaces. For detailed information on data format settings and data transfer and interrupt control methods, refer to the preceding sections.

19.9 Control Register Details

Table 19.9.1 List of UART Registers

Address	Register name		Function
0x300e10	UART_ST	UART Status Register	Indicates transfer, buffer and error statuses.
0x300e11	UART_TXD	UART Transmit Data Register	Transmit data
0x300e12	UART_RXD	UART Receive Data Register	Receive data
0x300e13	UART_MOD	UART Mode Register	Sets transfer data format.
0x300e14	UART_CTL	UART Control Register	Controls data transfer.
0x300e15	UART_EXP	UART Expansion Register	Sets IrDA mode.

The UART registers are described in detail below.

Note: When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

UART Status Register (UART_ST)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
UART Status Register (UART_ST)	**0x300e10** (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): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

FER is set to 1 when a framing error occurs. Framing errors occur when data is received with the stop bit set to 0. FER is reset by writing 1 or by setting RXEN/UART_CTL register to 0.

D5 PER: Parity Error Flag Bit

Indicates whether a parity error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

PER is set to 1 when a parity error occurs. Parity checking is enabled only when PREN/ UART_MOD register is set to 1 and is performed when received data is transferred from the shift register to the receive data buffer. PER is reset by writing 1 or by setting RXEN/UART_CTL register to 0.

D4 OER: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

OER is set to 1 when an overrun error occurs. Overrun errors occur when data is received in the shift register when the receive data buffer is already full and additional data is sent. The receive data buffer is not overwritten even if this error occurs. The shift register is overwritten as soon as the error occurs.
OER is reset by writing 1 or by setting RXEN/UART_CTL register to 0. Furthermore, set RXEN to 0 and read the receive data buffer successively until the RDRY flag is cleared.

D3 RD2B: Second Byte Receive Flag Bit

Indicates that the receive data buffer contains two received data.

1 (R): Second byte can be read
0 (R): Second byte not received (default)

RD2B is set to 1 when the second byte of data is loaded into the receive data buffer and is reset to 0 when the first data is read from the receive data buffer.

D2 TRBS: Transmit Busy Flag Bit

Indicates the transmit shift register status.

1 (R): Operating
0 (R): Standby (default)

TRBS is set to 1 when transmit data is loaded from the transmit data buffer into the shift register and is reset to 0 when the data transfer is completed. Inspect TRBS to determine whether the transmit circuit is operating or at standby.

D1 RDRY: Receive Data Ready Flag Bit

Indicates that the receive data buffer contains valid received data.

1 (R): Data can be read
0 (R): Buffer empty (default)

RDRY is set to 1 when received data is loaded into the receive data buffer and is reset to 0 when all data has been read from the receive data buffer.

D0 TDBE: Transmit Data Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Buffer empty (default)
0 (R): Data exists

TDBE is reset to 0 when transmit data is written to the transmit data buffer and is set to 1 when the data is transferred to the shift register.

UART Transmit Data Register (UART_TXD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Transmit Data Register (UART_TXD)	**0x300e11** (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

Write transmit data to be set in the transmit data buffer. (Default: 0x0)
The UART starts transmitting when data is written to this register. Data written to TXD[7:0] is retained until sent to the transmit data buffer. Transmitting data from within the transmit data buffer generates a cause of transmit buffer empty interrupt.
TXD7 (MSB) is invalid in 7-bit mode.
Serial converted data is output from the SOUT pin beginning with the LSB, in which the bits set to 1 are output as High level and bits set to 0 as Low level signals.
This register can also be read.

UART Receive Data Register (UART_RXD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Receive Data Register (UART_RXD)	0x300e12 (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

Data in the receive data buffer is read out in sequence, starting with the oldest. Received data is placed in the receive data buffer. The receive data buffer is a 2-byte FIFO that allows proper data reception until it fills, even if data is not read out. If the buffer is full and the shift register also contains received data, an overrun error will occur, unless the data is read out before reception of the subsequent data starts.

The receive circuit includes two receive buffer status flags: RDRY/UART_ST register and RD2B/UART_ST register. The RDRY flag indicates the presence of valid received data in the receive data buffer, while the RD2B flag indicates the presence of two received data in the receive data buffer.

A receive buffer full interrupt occurs when the received data in the receive data buffer reaches the number specified by RBFI/UART_CTL register.

0 is loaded into RXD7 in 7-bit mode.

Serial data input via the SIN pin is converted to parallel, with the initial bit as LSB, the High level bit as 1, and the Low level bit as 0. This data is then loaded into the receive data buffer.

This register is read-only. (Default: 0x0)

UART Mode Register (UART_MOD)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
UART Mode Register (UART_MOD)	0x300e13 (8 bits)	D7–6	–	reserved	–				–	–	0 when being read.
		D5	–	reserved	–				0	–	Do not set to 1.
		D4	CHLN	Character length select	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[7:6] Reserved

D5 Reserved (Do not set to 1.)

D4 CHLN: Character Length Select Bit

Selects the serial transfer data length.

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 received data parity checking is performed and whether a parity bit is added to transmit data. Setting PREN to 1 parity-checks the received data. A parity bit is automatically added to the transmit data. If PREN is set to 0, no parity bit is checked or added.

D2 PMD: Parity Mode Select Bit

Selects the parity mode.

1 (R/W): Odd parity

0 (R/W): Even parity (default)

Writing 1 to PMD selects odd parity; writing 0 to it selects even parity. Parity checking and parity bit addition are enabled only when PREN is set to 1. The PMD setting is disabled if PREN is 0.

D1 STPB: Stop Bit Select Bit

Selects the stop bit length.
1 (R/W): 2 bits
0 (R/W): 1 bit (default)

Writing 1 to STPB selects 2 stop bits; writing 0 to it selects 1 bit. The start bit is fixed at 1 bit.

D0 SSCK: Input Clock Select Bit

Selects the input clock.
1 (R/W): External clock (SCLK)
0 (R/W): Internal clock (default)

Selects whether the internal clock (T8F Ch.5 output clock) or an external clock (input via SCLK pin) is used. Writing 1 to SSCK selects an external clock; Writing 0 to it selects the internal clock.

UART Control Register (UART_CTL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
UART Control Register (UART_CTL)	0x300e14 (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 setup	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 interrupt requests to the ITC when a receive error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process receive errors using interrupts.

D5 RIEN: Receive Buffer Full Interrupt Enable Bit

Enables interrupt requests to the ITC caused when the received data quantity in the receive data buffer reaches the quantity specified in RBFI.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to read received data using interrupts.

D4 TIEN: Transmit Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC caused when transmission data in the transmit data buffer is sent to the shift register (i.e. when data transmission begins).
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the transmit data buffer using interrupts.

D[3:2] Reserved

D1 RBFI: Receive Buffer Full Interrupt Condition Setup Bit

Sets the quantity 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)

If receive buffer full interrupts are enabled (RIEN = 1), the UART outputs an interrupt request to the ITC when the quantity of received data specified by RBFI is loaded into the receive data buffer.
If RBFI is 0, an interrupt request is output as soon as one received data is loaded into the receive data buffer (when RDRY/UART_ST register is set to 1). If RBFI is 1, an interrupt request is output as soon as two received data are loaded into the receive data buffer (when RD2B/UART_ST register is set to 1).

D0 RXEN: UART Enable Bit

Enables data transfer by the UART.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set RXEN to 1 before starting UART transfers. Setting RXEN to 0 disables data transfers. Set the transfer conditions while RXEN is 0.
Disabling transfers by writing 0 to RXEN also clears transmit data buffer. The receive data buffer is not cleared.

UART Expansion Register (UART_EXP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Expansion Register (UART_EXP)	0x300e15 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–4	IRCLK[2:0]	IrDA receive detection clock select	IRCLK[2:0] / Division ratio 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
		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

Selects the prescaler output clock used as the IrDA input pulse detection clock.

Table 19.9.2 IrDA Receive Detection Clock (PCLK Division Ratio) Selection

IRCLK[2:0]	Prescaler output clock
0x7	1/128
0x6	1/64
0x5	1/32
0x4	1/16
0x3	1/8
0x2	1/4
0x1	1/2
0x0	1/1

(Default: 0x0)

This clock must be selected as a clock faster than the transfer clock sclk input from the T8F Ch.5 or the SCLK pin.
The demodulator circuit treats Low pulses with a width of at least 2 IrDA receive detection clock cycles as valid. Select the appropriate prescaler output clock to enable detection of input pulses with a minimum width of 1.41 µs.

D[3:1] Reserved

D0 IRMD: IrDA Mode Select Bit

Switches the IrDA interface function on and off.
1 (R/W): On
0 (R/W): Off (default)

Set IRMD to 1 to use the IrDA interface. When IRMD is set to 0, this module functions as a normal UART, with no IrDA functions.

20 Universal Serial Interface (USI)

20.1 USI Module Overview

The S1C33L27 incorporates a universal serial interface (USI) module with three communication channels that can be configured as a UART, SPI, or I²C interface unit by the software switch.
The following shows the main features of USI:

- Supports five interface modes: UART, SPI master, SPI slave, I²C master, and I²C slave modes.
- Three channels can be configured to different interface modes.
- Contains one-byte receive data buffer and one-byte transmit buffer.
- Supports both MSB first and LSB first modes.
- UART mode
  - Character length: 7 or 8 bits
  - Parity mode: even, odd, or no parity
  - Stop bit: 1 or 2 bits
  - Start bit: 1 bit fixed
  - Parity error, framing error, and overrun error detectable
  - Can generate receive buffer full, transmit buffer empty, and receive error interrupts.
  - Supports DMA transfer.
- SPI master/slave mode
  - Data length: 8 or 9 bits (master mode), or 8 bits fixed (slave mode)
  - Supports both fast and normal modes (master mode), or normal mode only (slave mode).
  - Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
  - Can generate receive buffer full, transmit buffer empty, and receive error interrupts.
  - Supports DMA transfer.
- I²C master/slave mode
  - 7-bit addressing mode (10-bit addressing is possible by software control.)
  - Supports single master configuration only (master mode).
  - Supports clock stretch/wait functions.
  - Can generate operation (start/stop, data transfer, ACK/NAK transfer) completion interrupts and receive error interrupts.

Figure 20.1.1 shows the USI configuration.

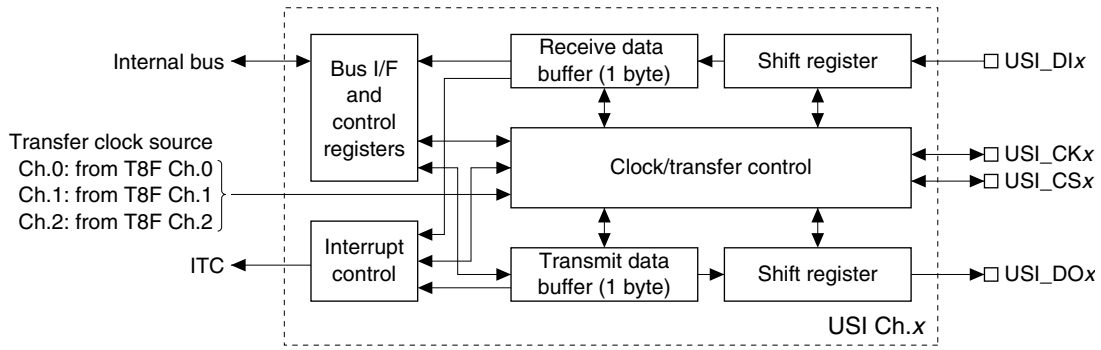


Figure 20.1.1 USI Configuration (one channel)

Note: Three channels in the USI module have the same functions except for control register addresses. For this reason, the description in this chapter applies to all USI channels. The '*x*' in the register name indicates a channel number (0 to 2).
Example: USI_GCFG*x* register
Ch.0: USI_GCFG0 register
Ch.1: USI_GCFG1 register
Ch.2: USI_GCFG2 register

20.2 USI Pins

Table 20.2.1 lists the USI input/output pins.

Table 20.2.1 List of USI Pins

Pin name	USI mode	Signal name	I/O	Function
USI_DI0 (Ch.0)	Software reset	–	I	Data input pin
USI_DI1 (Ch.1)	UART	uart_rx	I	Data input pin
USI_DI2 (Ch.2)	SPI master	spi_di	I	Inputs serial data sent from an external serial device.
	SPI slave	spi_di	I	
	I²C master	i2c_sda	I/O	Data input/output pin
	I²C slave	i2c_sda	I/O	Inputs/outputs serial data from/to the I²C bus. (∗1)
USI_DO0 (Ch.0)	Software reset	–	I	Data input pin
USI_DO1 (Ch.1)	UART	uart_tx	O	Data output pin
USI_DO2 (Ch.2)	SPI master	spi_do	O	Outputs serial data sent to an external serial device.
	SPI slave	spi_do	O	
	I²C master	–	–	Not used
	I²C slave	–	–	
USI_CK0 (Ch.0)	Software reset	–	I	Data input pin
USI_CK1 (Ch.1)	UART	–	–	Not used
USI_CK2 (Ch.2)	SPI master	spi_ck	O	Clock output pin Outputs the SPI clock.
	SPI slave	spi_ck	I	Clock input pin Inputs an external clock.
	I²C master	i2c_sck	I/O	SCL input/output pin Inputs SCL line status from the I²C bus. Also outputs the I²C clock.
	I²C slave	i2c_sck	I/O	SCL input/output pin Inputs SCL line status from the I²C bus. Also outputs a clock stretch condition.
USI_CS0 (Ch.0)	Software reset	–	I	Data input pin
USI_CS1 (Ch.1)	UART	–	–	Not used
USI_CS2 (Ch.2)	SPI master	–	–	
	SPI slave	#spi_ss	I	SPI slave select signal input pin Low level input to this pin selects USI (in SPI slave mode) as an SPI slave device.
	I²C master	i2c_sda	I/O	Data input/output pin
	I²C slave	i2c_sda	I/O	Inputs/outputs serial data from/to the I²C bus. (∗1)

∗1: When USI is configured to I²C master or slave mode, either the USI_DI*x* pin or the USI_CS*x* pin can be used as the data input/output pin. Note, however, that both the USI_DI*x* and USI_CS*x* pins cannot be used as the data input/output pin simultaneously.

∗2: After the pin functions are switched for USI, the pins are configured for input until the interface mode is set using the USI_GCFG*x* register. If USI software reset is performed when the pins are already configured for USI, all USI pins are configured for input. Before performing software reset while USI is operating, switch the pin functions to general-purpose input/output ports as necessary.

Note: Use a GPIO port to output the slave select signal when USI is configured to SPI master mode.

The USI input/output pins (USI_DI*x*, USI_DO*x*, USI_CK*x*, USI_CS*x*) are shared with I/O ports and are initially set as general-purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as USI input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

20.3 USI Clock Sources

Operating clock

The USI uses PCLK as the operating clock. Therefore, PCLK must be supplied from the CMU before starting the USI including setting the control registers. For more information on the PCLK supply, refer to the "Clock Management Unit (CMU)."

Transfer clock

When the USI Ch.*x* is configured to a UART, SPI master (normal mode), or I²C master device, the source clock for transfer is supplied by the fine mode 8-bit timer (T8F Ch.*x*). Program T8F Ch.*x* according to the transfer rate and enable supplying the source clock to the USI module. The USI module divides the source clock to generate the transfer clock (or sampling clock). Be aware that the division ratio in the USI depends on the interface mode.
When the USI is configured to an SPI master (fast mode) device, PCLK is used as the source clock.
When the USI is configured to an SPI slave or I²C slave device, the transfer clock is supplied from the external master device. However, SPI slave mode uses PCLK and I²C slave mode uses the T8F Ch.*x* output clock to generate the sampling signal.

Table 20.3.1 USI Clocks

Clock	Interface mode	USI
Operating clock	UART	PCLK
	SPI master	PCLK
	SPI slave	PCLK
	I²C master	PCLK
	I²C slave	PCLK
Transfer/sampling clock source (division ratio in USI)	UART	T8F Ch.*x* (f_{SOURCE}/8)
	SPI master	Normal mode: T8F Ch.*x* (f_{SOURCE}/2) Fast mode: PCLK (f_{PCLK})
	SPI slave	PCLK (f_{PCLK}/4) for sampling
	I²C master	T8F Ch.*x* (f_{SOURCE}/8)
	I²C slave	T8F Ch.*x* (f_{SOURCE}) for sampling

$f_{SOURCE} = f_{SYS\_CLK} \times DF / (TR + 1)$
bps = f_{SOURCE} / 8 (UART mode, I²C master mode)
bps = f_{SOURCE} / 2 (SPI master mode)

f_{SOURCE}: T8F Ch.*x* output clock frequency [Hz]
f_{SYS_CLK}: System clock frequency [Hz]
bps: Transfer rate [bps]
DF: Division ratio set by DF[3:0]/T8F_CLK*x* register (T8F Ch.*x*)
TR: Reload data to be set to the T8F_TR*x* register (T8F Ch.*x*)

Example: UART mode, transfer rate = 115,200 bps, system clock = 48 MHz, DF[3:0]/T8F_CLK*x* register setting (T8F Ch.*x*) = 1/1.

$TR = (f_{SYS\_CLK} \times DF) / (bps \times 8) - 1 = 48{,}000{,}000 \times 1 / (115{,}200 \times 8) - 1 = 51 (= 0x33)$

For more information on controlling the T8F module, refer to the "Fine Mode 8-bit Timers (T8F)" chapter.

Note: When the USI is set to I²C slave mode, i2c_sck (I²C clock) is supplied from the external I²C master. The T8F output clock frequency (f_{SOURCE}) should be determined according to the i2c_sck frequency.

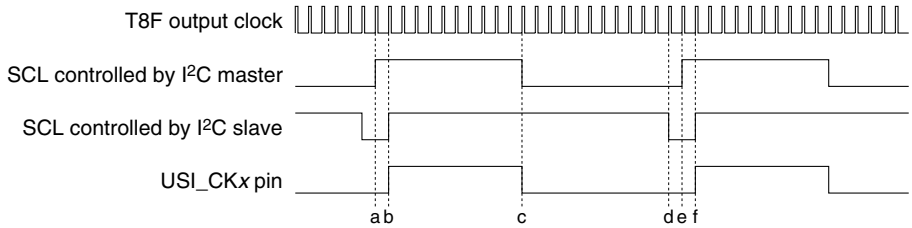


Figure 20.3.1 I²C Clock in I²C Slave Mode

Tbf = Ti2c_baud_rate
Tbc = Ti2c_baud_rate_high
Tcf = Ti2c_baud_rate_low

Tce: The I²C master occupies the SCL line by driving it to low.
Tac: The I²C master releases the SCL line.
Tdf: In order to finish the internal operations, the I²C slave occupies the SCL line for two source clock (T8F output clock) cycles by driving it to low after detecting that the I²C master drives the SCL line to low.

The T8F output clock frequency (f_{SOURCE}) must be set so that the conditions shown below are satisfied.

$f_{SOURCE} > 3/Tbc$
$f_{SOURCE} > 4/Tce$

Be aware that the actual SCL signal will be delayed, as the I²C slave forcibly drives the SCL line to low. The figure below shows an example in which the timing becomes worse.

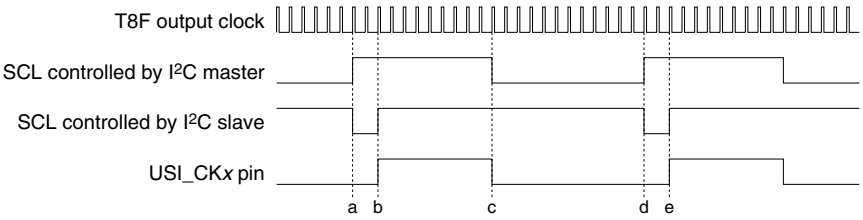


Figure 20.3.2 Example of Delayed I²C Clock

20.4 USI Module Settings

Make the following settings before starting data transfers using the USI module.
(1) Configure the pins to be used for USI according to the interface mode. (See Section 20.2.)
(2) Program the clock source module to supply the clock required to the USI module. (See Section 20.3.)
(3) Reset the USI module.
(4) Set the USI interface mode and a general condition (MSB first/LSB first) to be applied to all interface modes.
(5) Set the data format and operating conditions for the interface mode selected.
(6) Set interrupt and DMA transfer conditions if necessary. (See Section 20.7.)

20.4.1 USI Module Software Reset

Writing 0x0 to USIMOD[2:0]/USI_GCFG*x* register resets the USI module circuits. Be sure to perform software reset and then set the interface mode before changing other USI configurations. The software reset operation resets all the USI register contents to the initial values.

20.4.2 Interface Mode

The USI module provides five serial interface functions shown in Section 20.1. Each channel can be configured to one of them using the USIMOD[2:0]/USI_GCFG*x* register.

Table 20.4.2.1 Interface Mode Selection

USIMOD[2:0]	Interface mode
0x5	I²C slave
0x4	I²C master
0x3	SPI slave
0x2	SPI master
0x1	UART
0x0	Software reset

(Default: 0x0)

Note: Be sure to perform software reset and set the interface mode before changing other USI configurations.

20.4.3 General Settings for All Interface Modes

MSB first/LSB first selection

Use LSBFST/USI_GCFG*x* register to select whether the data MSB or LSB is input/output first.
LSB first is selected when LSBFST is set to 0 (default). MSB first is selected when LSBFST is set to 1.

20.4.4 Settings for UART Mode

When the USI is used in UART mode, configure the data length, stop bit, and parity bit. The start bit length is fixed at 1 bit.

Data length

Use UCHLN/USI_UCFG*x* register to select the data length. Setting UCHLN to 0 (default) configures the data length to 7 bits. Setting UCHLN to 1 configures it to 8 bits.

Stop bit

Use USTPB/USI_UCFG*x* register to select the stop bit length. Setting USTPB to 0 (default) configures the stop bit length to 1 bit. Setting USTPB to 1 configures it to 2 bits.

Parity bit

Use UPREN/USI_UCFG*x* register to select whether the parity function is enabled or not. Setting UPREN to 0 (default) disables the parity function. In this case, no parity bit will be added to transfer data and receive data will not be checked for parity. Setting UPREN to 1 enables the parity function. In this case, a parity bit will be added to transfer data and receive data will be checked for parity.
When the parity function is enabled, the parity mode should be selected using UPMD/USI_UCFG*x* register. Setting UPMD to 0 (default) adds a parity bit and checks for odd parity. Setting UPMD to 1 adds a parity bit and checks for even parity.

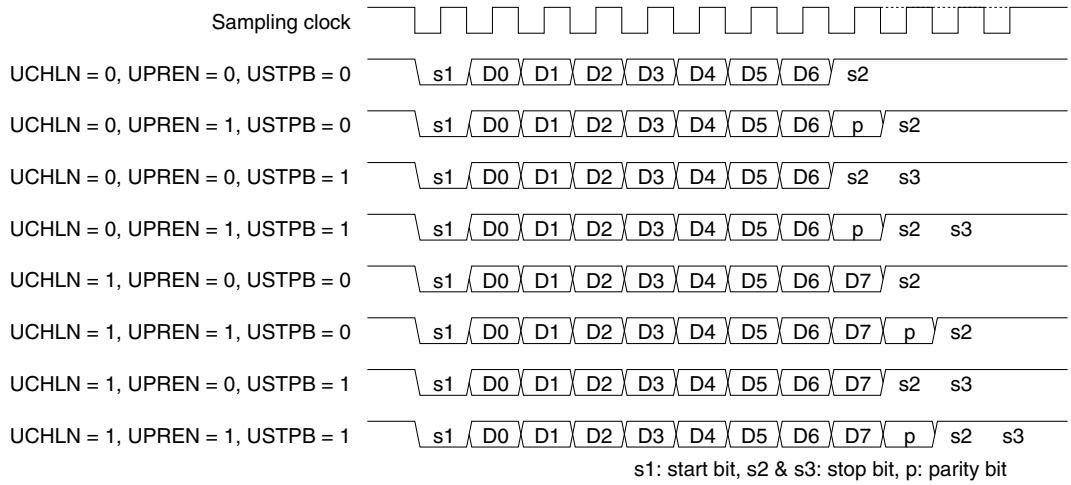


Figure 20.4.4.1 Transfer Data Format in UART Mode (LSB first)

20.4.5 Settings for SPI Mode

When the USI is used in SPI mode (master or slave), configure the SPI clock polarity/phase. When used in SPI master mode, select the clock mode and data length.
Note that the data length in SPI slave mode is fixed at 8 bits.

SPI clock polarity and phase settings (master mode and slave mode)

Use SCPOL/USI_SCFG*x* register to select the SPI clock polarity. Setting SCPOL to 1 treats the SPI clock as active low. Setting it to 0 (default) treats it as active high.
The SPI clock phase can be selected using SCPHA/USI_SCFG*x* register.
These control bits set transfer timing as shown in Figure 20.4.5.1.

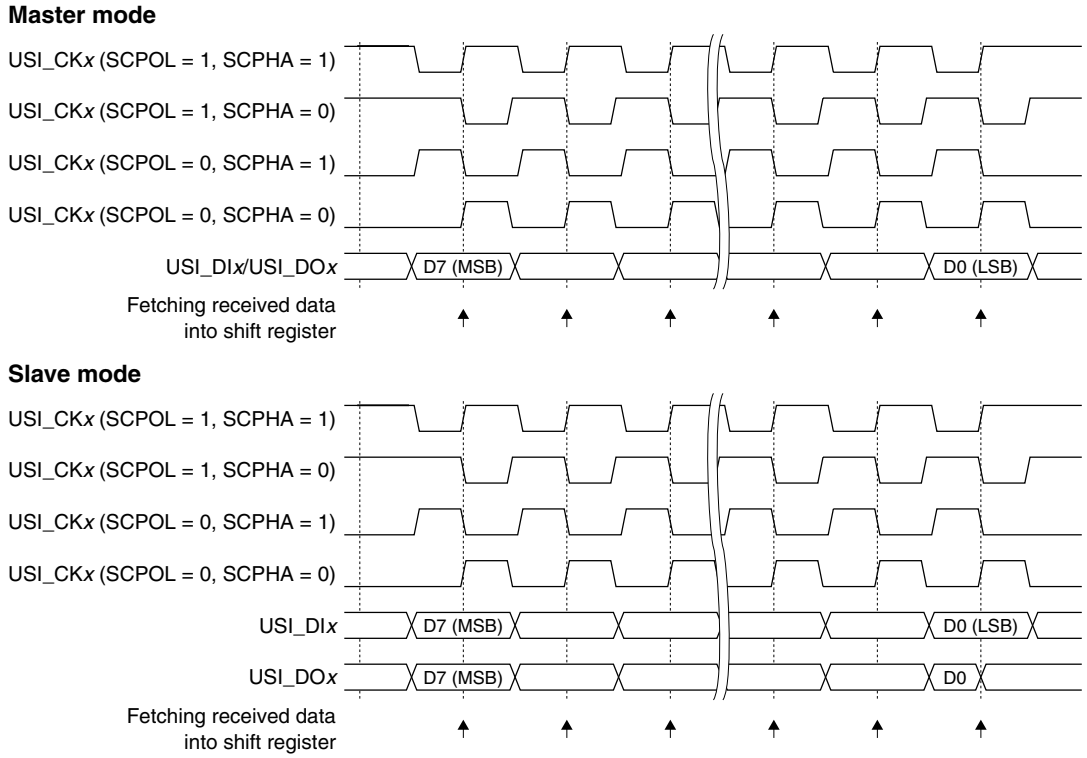


Figure 20.4.5.1 Clock and Data Transfer Timing (MSB first)

Clock mode (master mode only)

In SPI master mode, either normal or fast clock mode can be selected using SFSTMOD/USI_SCFG*x* register. Setting SFSTMOD to 0 (default) places the USI into normal mode and the USI generates the transfer clock by dividing the T8F output by 2. Setting SFSTMOD to 1 places the USI into fast mode and the USI uses PCLK supplied from the CMU directly as the transfer clock. The fast mode does not use the T8F.

The SPI slave mode uses the T8F output clock for generating the sampling clock.

Data length (master mode only)

In SPI master mode, the data length can be selected using SCHLN/USI_SCFG*x* register. Setting SCHLN to 0 (default) configures the data length to 8 bits.
Setting SCHLN to 1 configures the data length to 9 bits. In 9-bit mode, 8-bit data is prefixed with a command bit (1 bit). The command bit is used for controlling the SPI LCD controller connected to the USI. The command bit value to be transmitted can be specified using SCMD/USI_SCFG*x* register. Setting SCMD to 1 configures the command bit to high. Setting SCMD to 0 configures the command bit to low.

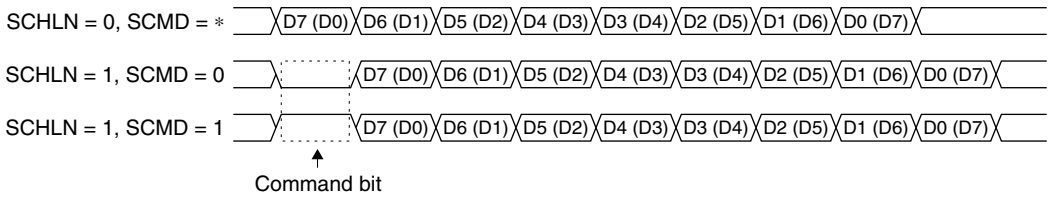


Figure 20.4.5.2 9-bit Transfer Data Format in SPI Master Mode

The data length in SPI slave mode is fixed at 8 bits.

20.4.6 Settings for I²C Mode

The I²C mode does not need to set data format and other conditions. The data length in I²C mode is fixed at 8 bits.

20.5 Data Transfer Control

This section describes how to control data transfers. The following explanations assume that the configurations described above and interrupt/DMA settings have already been finished.

20.5.1 Data Transfer in UART Mode

Data transmission

To start data transmission in UART mode, write the transmit data to the transmit data buffer (TD[7:0]/USI_TD*x* register).

The buffer data is sent to the transmit shift register, and the start bit is output from the USI_DO*x* pin. The data in the shift register is then output in sequence. Following output of the eighth data bit, the parity bit (if parity is enabled) and the stop bit are output.

The transmitter circuit includes two status flags: UTDIF/USI_UIF*x* register and UTBSY/USI_UIF*x* register.

The UTDIF flag indicates the transmit data buffer status. This flag is set to 1 indicating that the transmit data buffer becomes empty when data written to the transmit data buffer is sent to the transmit shift register. UTDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 20.7). Write subsequent data to the transmit data buffer to start the following transmission using this interrupt or DMA. The transmit data buffer size is 1 byte, but a shift register is provided separately to allow data to be written while the previous data is being sent. If an interrupt or DMA is not used for transmission, be sure to confirm that the transmit data buffer is empty before writing transmit data. Writing data before UTDIF has been set will overwrite earlier transmit data inside the transmit data buffer. After UTDIF is set to 1, it can be reset to 0 by writing 1.

The UTBSY flag indicates the USI status in UART mode. This flag switches to 1 when transmit data is written to the transmit buffer and reverts to 0 after both the shift register and transmit buffer become empty.

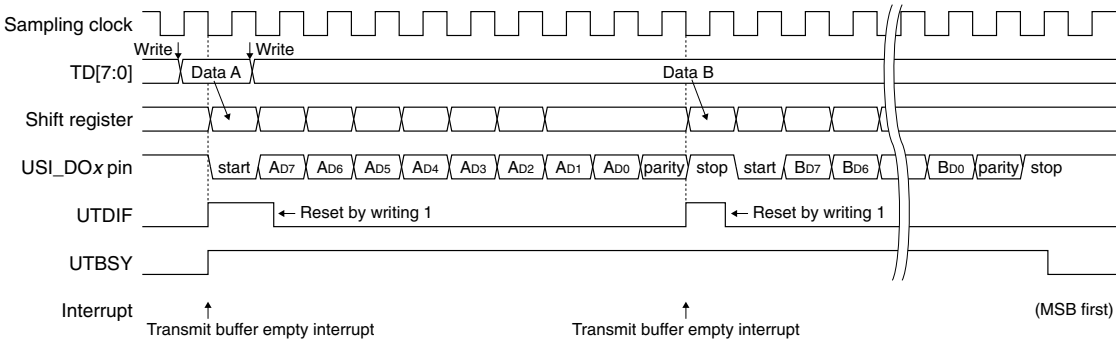


Figure 20.5.1.1 Data Transmission Timing Chart (UART mode)

Data reception

When the external serial device sends a start bit, the receiver circuit detects its low level and starts sampling the following data bits. Once the 8-bit data has been received into the shift register, the received data is loaded into the receive data buffer (RD[7:0]/USI_RD*x* register). If parity checking is enabled, the receiver circuit checks the received data at the same time by checking the parity bit received immediately after the eighth data bit.

The receiver circuit includes two status flags: URDIF/USI_UIF*x* register and URBSY/USI_UIF*x* register.

The URDIF flag indicates the receive data buffer status. This flag is set to 1 indicating that the received data can be read out when data received in the shift register is loaded to the receive data buffer. URDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 20.7). Read the received data from the receive data buffer using this interrupt or DMA. The receive data buffer size is 1 byte, therefore the received data must be read before the subsequent data reception has completed. Furthermore, URDIF must be reset by writing 1. If the next reception is completed when URDIF is 1 and the receive data buffer (USI_RD*x* register) is not read, an overrun error occurs (at the time stop bit has been received).

The URBSY flag indicates the shift register status. This flag is set to 1 while data is being received in the shift register and reverts to 0 once the received data is loaded to the receive data buffer. Read this flag to check whether the receiver circuit is operating or at standby.

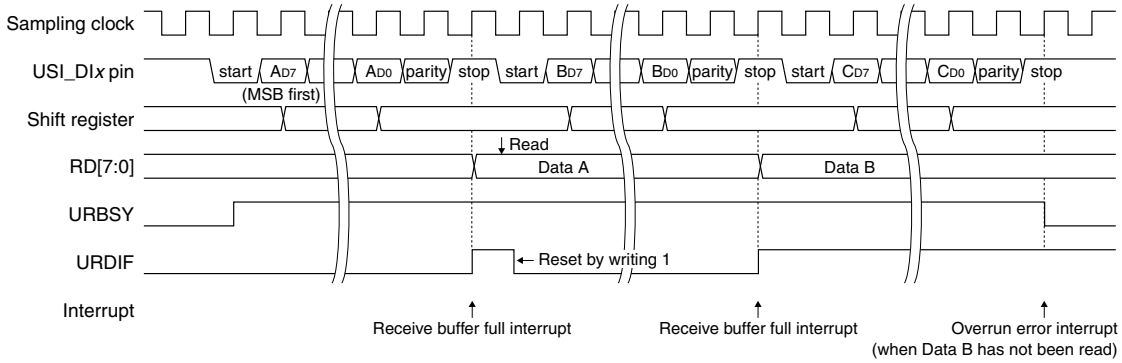


Figure 20.5.1.2 Data Receiving Timing Chart (UART mode)

20.5.2 Data Transfer in SPI Mode

Data transmission

To start data transmission in SPI mode, write the transmit data to the transmit data buffer (TD[7:0]/USI_TD*x* register).

The buffer data is sent to the transmit shift register. In SPI master mode, the module starts clock output from the USI_CK*x* pin. In SPI slave mode, the module awaits clock input from the USI_CK*x* pin. The data in the shift register is shifted in sequence at the clock rising or falling edge (see Figure 20.4.5.1) and sent from the USI_DO*x* pin.

The SPI controller includes two status flags for transfer control: STDIF/USI_SIF*x* register and SSIF/USI_SIF*x* register.

The STDIF flag indicates the transmit data buffer status. STDIF is set to 1 indicating that the transmit data buffer becomes empty when data written to the transmit data buffer is sent to the transmit shift register. STDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 20.7). Write subsequent data to the transmit data buffer to start the following transmission using this interrupt or DMA. The transmit data buffer size is 1 byte, but a shift register is provided separately to allow data to be written while the previous data is being sent. If an interrupt or DMA is not used for transmission, be sure to confirm that the transmit data buffer is empty before writing transmit data. Writing data before STDIF has been set will overwrite earlier transmit data inside the transmit data buffer.

In SPI master mode, the SSIF flag indicates the USI status. This flag switches to 1 when transmit data is written to the transmit buffer and reverts to 0 after both the shift register and transmit buffer become empty. Read this flag to check whether the SPI controller is operating or at standby.

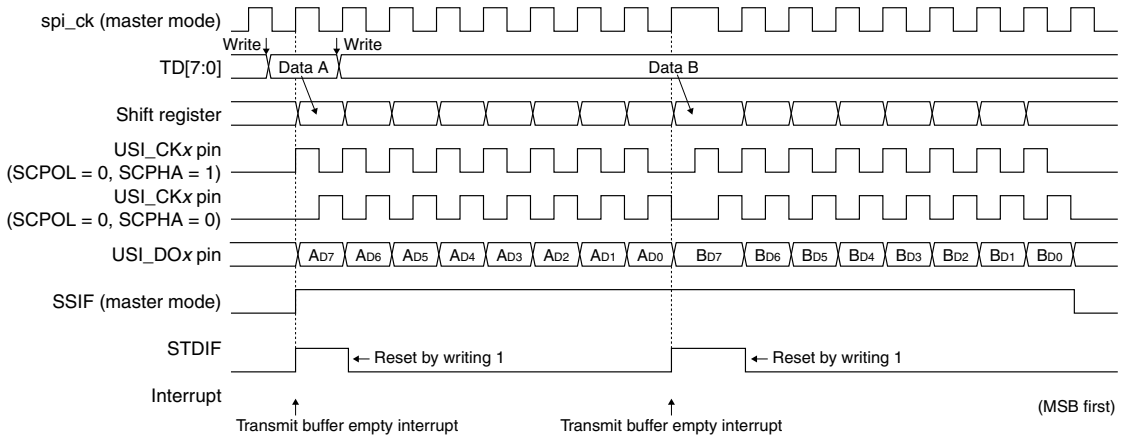


Figure 20.5.2.1 Data Transmission Timing Chart (SPI mode)

Data reception

In SPI master mode, write dummy data to the transmit data buffer. Writing to the transmit data buffer creates the trigger for reception as well as transmission start. Writing actual transmit data enables simultaneous transmission and reception. This starts the SPI clock output from the USI_CK*x* pin.

In SPI slave mode, the module waits until the clock is input from the USI_CK*x* pin. There is no need to write to the transmit data buffer if no transmission is required. The receiving operation is started by the clock input from the master device. If data is transmitted simultaneously, write transmit data to the transmit data buffer before the clock is input.

The data is received in sequence in the shift register at the SPI clock edge (see Figure 20.4.5.1). The received data is loaded into the receive data buffer once the 8 bits of data are received in the shift register.

The received data in the buffer can be read from RD[7:0]/USI_RD*x* register.

The SPI controller includes two status flags for transfer control: SRDIF/USI_SIF*x* register and SSIF/USI_SIF*x* register.

The SRDIF flag indicates the receive data buffer status. This flag is set to 1 when the data received in the shift register is loaded into the receive data buffer, indicating that the received data can be read out. SRDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 20.7). Read the received data from the receive data buffer using this interrupt or DMA. The receive data buffer size is 1 byte, therefore the received data must be read before the subsequent data reception has completed. Furthermore, SRDIF must be reset by writing 1. While SRDIF is set to 1, the next received data will not be transferred from the shift register to the receive data buffer (the first byte data exists in the receive data buffer and the second byte data exists in the shift register). An overrun error occurs if the third byte data is received in this condition, as the second byte data in the shift register is corrupted (an overrun error occurs at the time the first bit of the third byte is fetched).

In SPI master mode, the SSIF flag indicates the shift register status. This flag switches to 1 at the beginning of data reception and reverts to 0 once the data is received. Read this flag to check whether the SPI controller is operating or at standby.

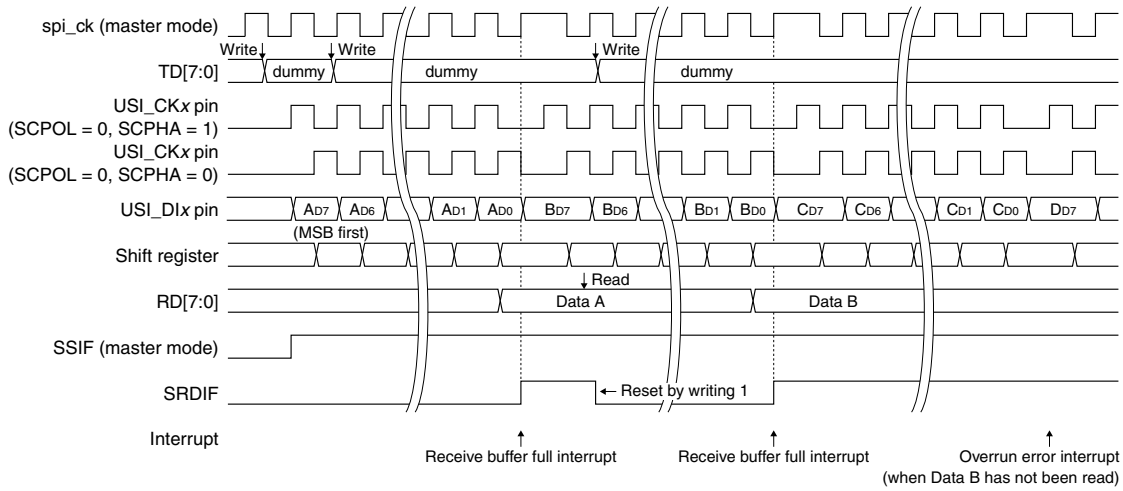


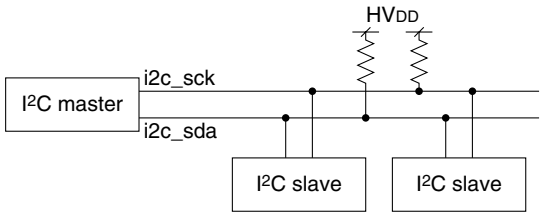
Figure 20.5.2.2 Data Receiving Timing Chart (SPI mode)

Slave select signal

In SPI slave mode, data transmission/receiving operations are enabled when the master device's slave select signal input to the USI_CS*x* pin is low. When the slave select signal is high, the SPI controller does not start data transfer even if the clock is input to the USI_CK*x* pin from the master device. The slave select signal status can be checked using SSIF/USI_SIF*x* register (it functions as the shift register status flag in SPI master mode). SSIF goes 1 when the slave select signal is inactive (high); it goes 0 when the slave select signal is active (low).

If a slave select output is required in SPI master mode, use a general-purpose I/O port and control its output with software.

20.5.3 Data Transfer in I²C Mode



The i2c_sck and i2c_sda pins go to a low level or high impedance status when the USI module is set to I²C mode. These pins do not output a high level, so the i2c_sck and i2c_sda lines should be pulled up to HVDD with an external pull-up resistor.

Note: Be sure to avoid pulling these pins up to a voltage that exceeds the HVDD level.

Figure 20.5.3.1 I²C Mode Connection Example

Control method in I²C master mode

Data transfer in I²C master mode is controlled using IMTGMOD[2:0]/USI_IMTG*x* register and IMTG/USI_IMTG*x* register. Select an I²C master operation using IMTGMOD[2:0] and write 1 to IMTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

Table 20.5.3.1 Trigger List in I²C Master Mode

IMTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception
0x2	Data transmission
0x1	Stop condition
0x0	Start condition

(Default: 0x0)

Writing 1 to IMTG sets IMBSY/USI_IMIF*x* register to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, IMBSY is reset to 0. At the same time, the interrupt flag (IMIF/USI_IMIF*x* register) is also set to 1. After an interrupt occurs, read the status bits (IMSTA[2:0]/USI_IMIF*x* register) to check the operation finished. Then clear IMIF by writing 1. IMSTA[2:0] will be automatically cleared to 0x0.

Table 20.5.3.2 I²C Master Status Bits

IMSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been generated.
0x0	Start condition has been generated.

(Default: 0x0)

Data transmission in I²C master mode

The following describes the data transmission procedure in I²C master mode.

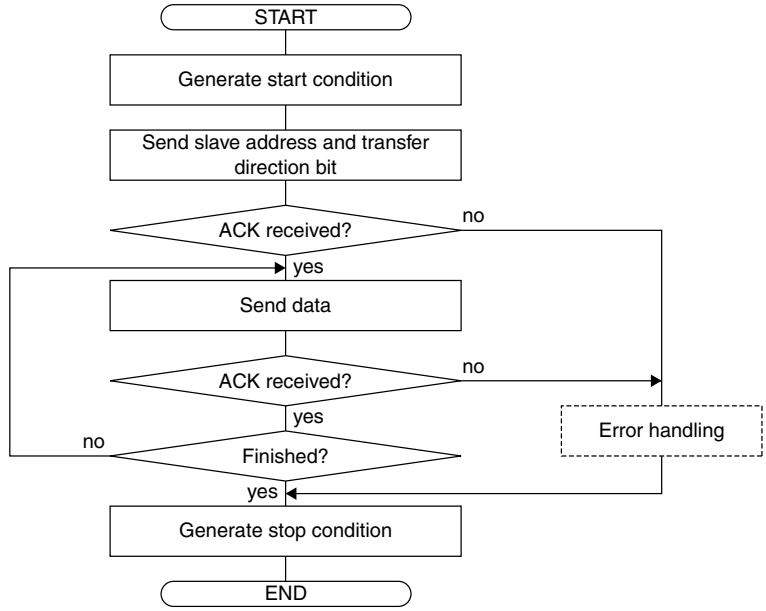
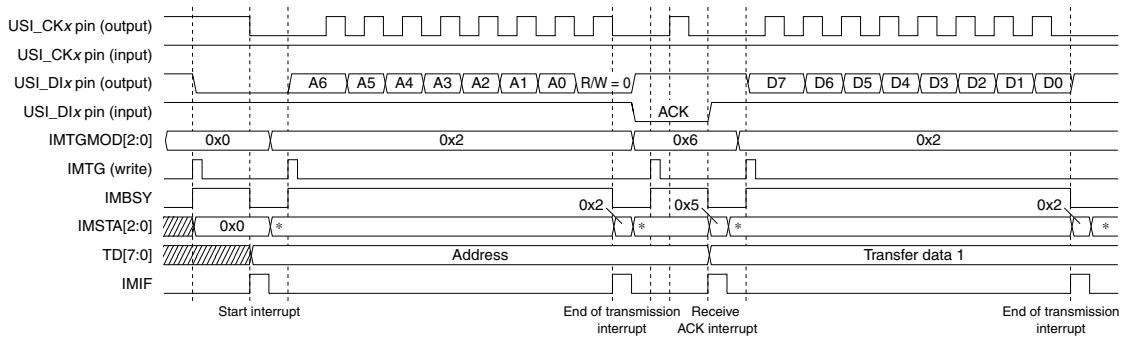
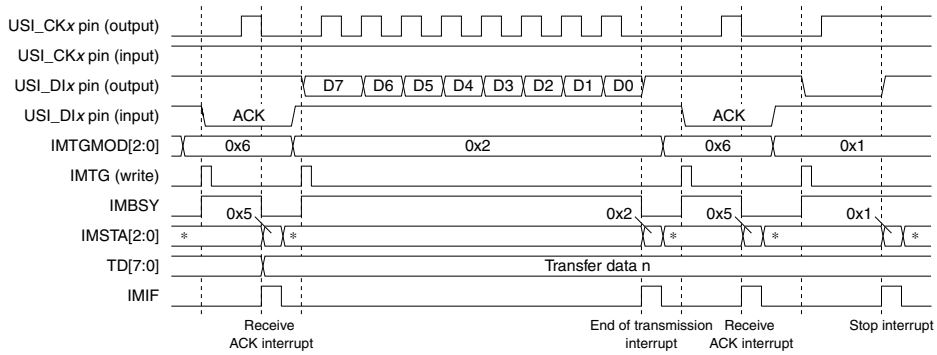


Figure 20.5.3.2 I²C Master Data Transmission Flow Chart



(1) Start condition → Data transmission



(2) Data transmission → Stop condition

∗ When IMIF is cleared via software, IMSTA[2:0] is also cleared to 0x0.

Figure 20.5.3.3 I²C Master Data Transmission Timing Chart

(1) Generating start condition

I²C data transfer starts when the I²C master device generates a start condition. The start condition applies when the SCL line is maintained at high and the SDA line is pulled down to low.

To generate a start condition in this I²C master, set IMTGMOD[2:0] to 0x0 (default) and write 1 to IMTG.

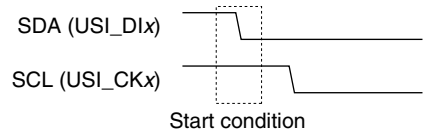


Figure 20.5.3.4 Start Condition

IMBSY is set to 1 while a start condition is being generated. When the start condition is generated, IMBSY is reset to 0 and IMSTA[2:0] is set to 0x0. The I²C bus is busy from this point on.

Note: Other operations cannot be started before a start condition is generated.

(2) Sending slave address and transfer direction bit

After a start condition has been generated, send the address of the slave device to be communicated and a transfer direction bit. I²C slave addresses are either 7-bit or 10-bit. This module uses an 8-bit transfer data buffer to send the slave address and transfer direction bit, enabling single transfers in 7-bit address mode. In 10-bit mode, data is sent twice or three times under software control. Figure 20.5.3.5 shows the configuration of the address data.

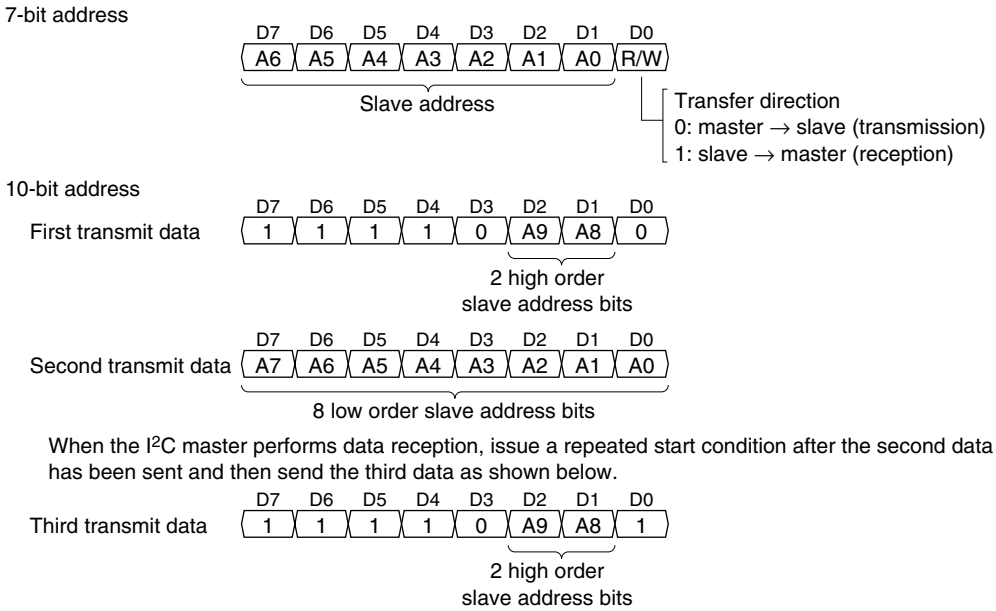


Figure 20.5.3.5 Transmit Data Specifying Slave Address and Transfer Direction

The transfer direction bit indicates the data transfer direction after the slave address has been sent. Set this bit to 0 when sending data from the master to the slave.

To send a slave address, set the address with the transfer direction bit to the transmit data buffer (TD[7:0]/USI_TD*x* register). Then set IMTGMOD[2:0] to 0x2 and write 1 to IMTG.

To send a 10-bit address, execute this procedure twice or three times as shown in Figure 20.5.3.5.

Writing 1 to IMTG sets IMBSY to 1. When data in the transmit data buffer is sent to the transmit shift register, IMBSY reverts to 0 and IMSTA[2:0] is set to 0x2. Confirm that the slave address (each byte) has been sent by reading IMBSY or using an interrupt.

After a slave address has been sent, the selected slave device sends back an ACK by pulling down the SCL line to low. If the SCL line maintains high, it is regarded as a NAK. In this case, the I²C controller cannot communicate with the slave device specified.

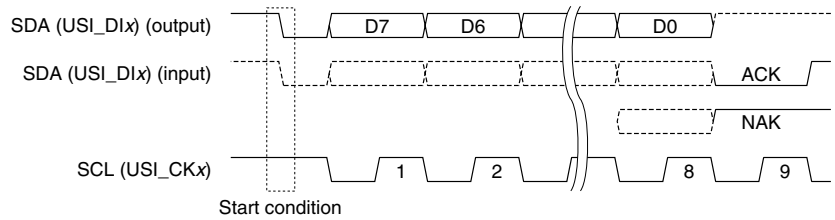


Figure 20.5.3.6 ACK and NAK

It is necessary to check that an ACK has been received before sending data. To do this, set IMTGMOD[2:0] to 0x6 and write 1 to IMTG after the slave address has been sent.
IMBSY is set to 1 while an ACK/NAK is being detected and it reverts to 0 when the detection has completed. Receiving an ACK sets IMSTA[2:0] to 0x5; receiving a NAK sets it to 0x6. Check IMSTA[2:0] after confirming IMBSY or using an interrupt. When an ACK has been received, perform data transmission. When a NAK has been received, perform an error handling.

(3) Data transmission

The data transmission procedure is the same as that of the slave address transmission.
1. Write an 8-bit transmit data to the transmit data buffer (TD[7:0]).
2. Set IMTGMOD[2:0] to 0x2 and IMTG to 1.

This trigger transfers the buffer data to the transmit shift register to start transmission. The module starts clock output from the USI_CK*x* pin. The data in the shift register is shifted in sequence with the clock and sent from the USI_DO*x* pin.
Writing 1 to IMTG sets IMBSY to 1. When data in the transmit data buffer is sent to the transmit shift register, IMBSY reverts to 0 and IMSTA[2:0] is set to 0x2 (end of transmit data). An interrupt request can be generated at this point. Write subsequent data to the transmit data buffer to start the following transmission using this interrupt.
However, as in the case of the slave address transmission, check that the slave device has sent back an ACK (by setting IMTGMOD[2:0] to 0x6 and IMTG to 1) before starting the following 8-bit data transmission.
Repeat an 8-bit data transmission and ACK receiving check for the required number of times.

(4) Generating stop condition

To end I²C communication after all data has been sent, the I²C master must generate a stop condition. The stop condition applies when the SCL line is maintained at high and the SDA line is pulled up from low to high. To generate a stop condition in this I²C master, set IMTGMOD[2:0] to 0x1 and write 1 to IMTG.

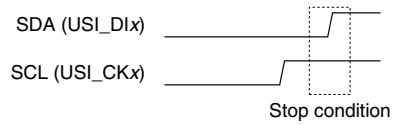


Figure 20.5.3.7 Stop Condition

IMBSY is set to 1 while a stop condition is being generated. When the stop condition is generated, IMBSY is reset to 0 and IMSTA[2:0] is set to 0x1. Read IMBSY or use an interrupt to check that a stop condition has been generated. The I²C bus subsequently switches to free state.

(5) Generating repeated start condition

To make it possible to continue with a different data transfer after a data transmission has completed, the I²C master can omit stop condition generation and generate a repeated start condition. To generate a repeated start condition, perform a start condition generation procedure described in Step (1). Slave address transmission is subsequently possible with the I²C bus remaining in the busy state.

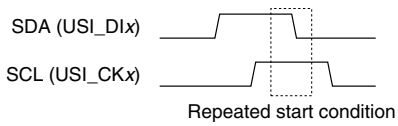


Figure 20.5.3.8 Repeated Start Condition

Data reception in I²C master mode

The following describes the data receiving procedure in I²C master mode.

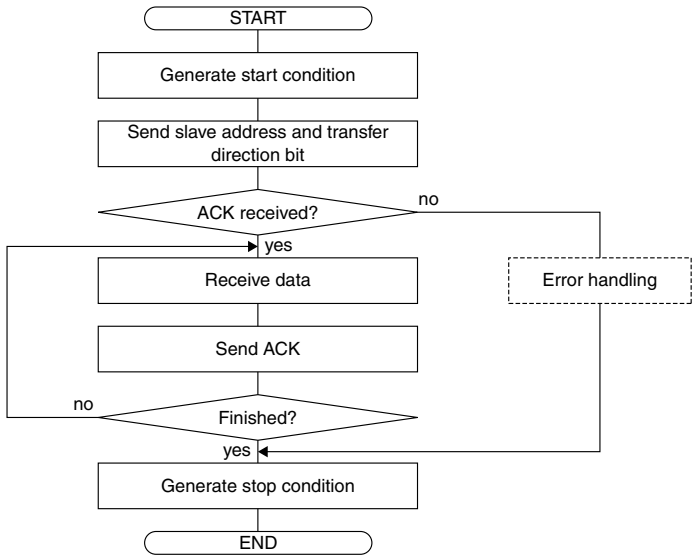
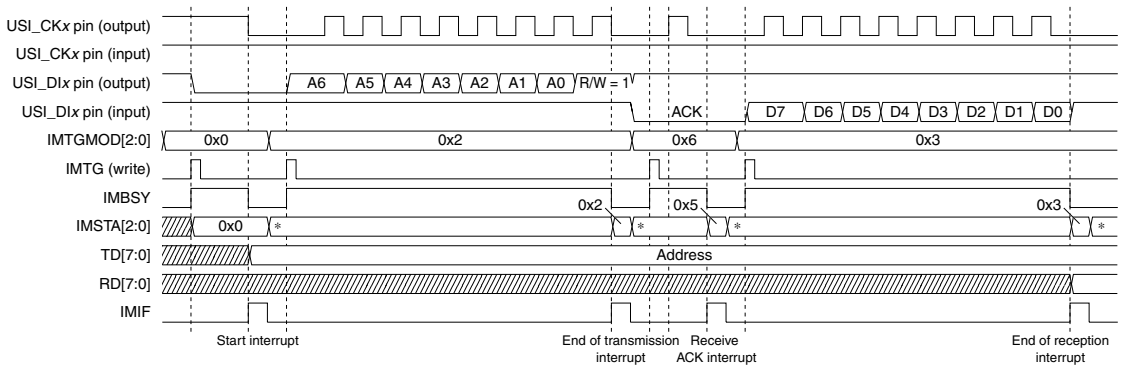
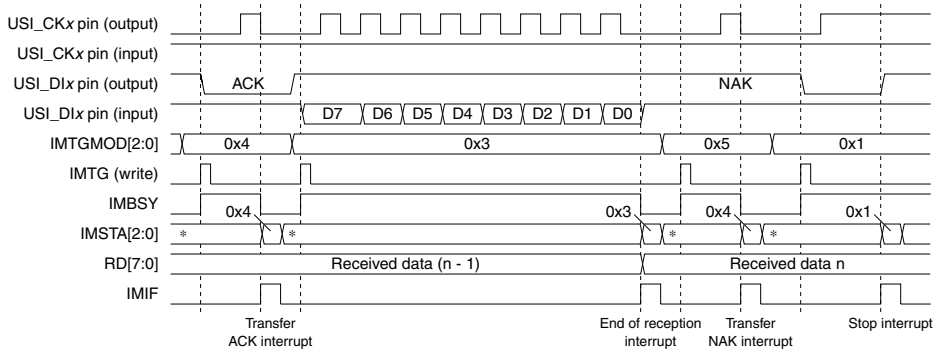


Figure 20.5.3.9 I²C Master Data Receiving Flow Chart



(1) Start condition → Data reception



(2) Data reception → Stop condition

∗ When IMIF is cleared via software, IMSTA[2:0] is also cleared to 0x0.

Figure 20.5.3.10 I²C Master Data Receiving Timing Chart

Note: The timing chart above shows a basic transfer operation that does not include an actual I²C transfer procedure. See "Receiving control byte in I²C slave mode" in "20.9 Precautions."

(1) Generating start condition

The procedure is the same as that of data transmission in I²C master mode.

(2) Sending slave address and transfer direction bit
The procedure is the same as that of data transmission in I²C master mode. However, send the slave address with the transfer direction bit set to 1. Then check that the slave device sends back an ACK.

(3) Data reception
To start data reception, set IMTGMOD[2:0] to 0x3 and write 1 to IMTG.
This trigger starts outputting 8 clocks from the USI_CK*x* pin. The USI_DO*x* pin status is sampled in sync with the clock and loaded to the shift register. The received data is loaded to the receive data buffer (RD[7:0]/USI_RD*x* register) once the 8-bit data has been received in the shift register.
Writing 1 to IMTG sets IMBSY to 1. When the received data is loaded to the receive data buffer, IMBSY reverts to 0 and IMSTA[2:0] is set to 0x3 (end of receive data). An interrupt request can be generated at this point. Read the received data from the receive data buffer using this interrupt.

It is necessary to send back an ACK or NAK to the slave device after an 8-bit data has been received. To send back an ACK, set IMTGMOD[2:0] to 0x4 and write 1 to IMTG. To send back a NAK, set IMTGMOD[2:0] to 0x5 and write 1 to IMTG.
IMBSY is set to 1 while an ACK/NAK is being sent and it reverts to 0 when the transmission has completed. An interrupt can be generated at this point. When an ACK or NAK has been sent, IMSTA[2:0] is set to 0x4.
Repeat an 8-bit data reception and ACK (NAK) transmission for the required number of times.

(4) Generating stop condition
The procedure is the same as that of data transmission in I²C master mode.

(5) Generating repeated start condition
The procedure is the same as that of data transmission in I²C master mode.

Clock stretch function

During transmitting/receiving data, the slave device may issue a wait request to the master device by pulling down the SCL line to low until the slave device becomes ready to transmit/receive the subsequent data. The master device enters a standby state until the wait request is canceled (the SCL line goes high).
This I²C controller supports this clock stretch function. When a clock stretch condition is detected after a slave address or data has been sent/received, this module enters a waiting status and it does not start operating even if it accepts a trigger for data transfer until the clock stretch status is canceled. IMBSY is maintained at 1 until the triggered operation has completed including a waiting status.

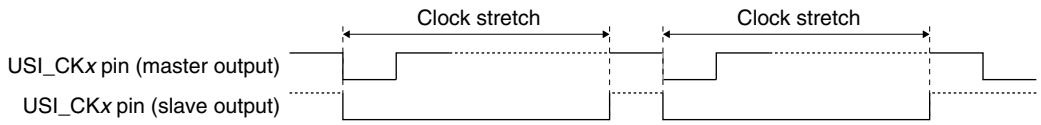


Figure 20.5.3.11 Clock Stretch

Control method in I²C slave mode

Data transfer in I²C slave mode is controlled using ISTGMOD[2:0]/USI_ISTG*x* register and ISTG/USI_ISTG*x* register. Select an I²C slave operation using ISTGMOD[2:0] and write 1 to ISTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

Table 20.5.3.3 Trigger List in I²C Slave Mode

ISTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception/stop condition detection
0x2	Data transmission
0x1	Reserved
0x0	Wait for start condition

(Default: 0x0)

Writing 1 to ISTG sets ISBSY/USI_ISIF*x* register to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, ISBSY is reset to 0. At the same time, the interrupt flag (ISIF/USI_ISIF*x* register) is also set to 1. After an interrupt occurs, read the status bits (ISSTA[2:0]/USI_ISIF*x* register) to check the operation finished. Then, clear ISIF by writing 1. This also automatically reset ISSTA[2:0] to 0x0.

Table 20.5.3.4 I²C Slave Status Bits

ISSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been detected.
0x0	Start condition has been detected.

(Default: 0x0)

Data transmission in I²C slave mode

The following describes the data transmission procedure in I²C slave mode.

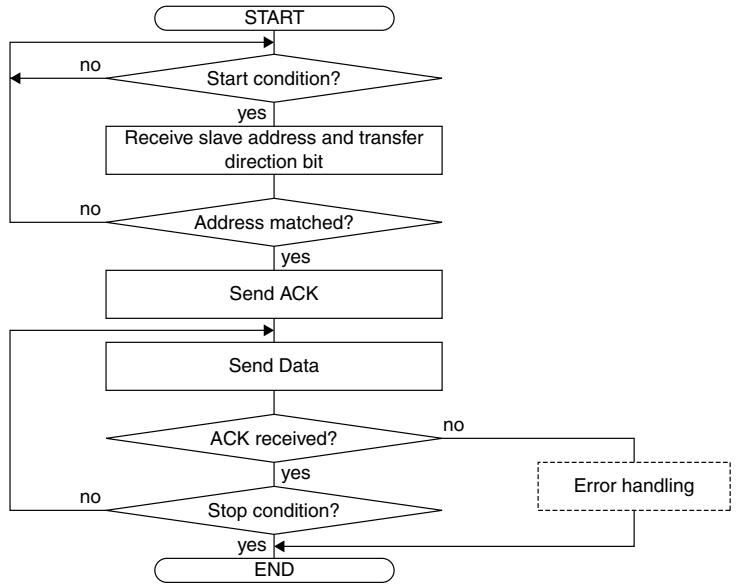
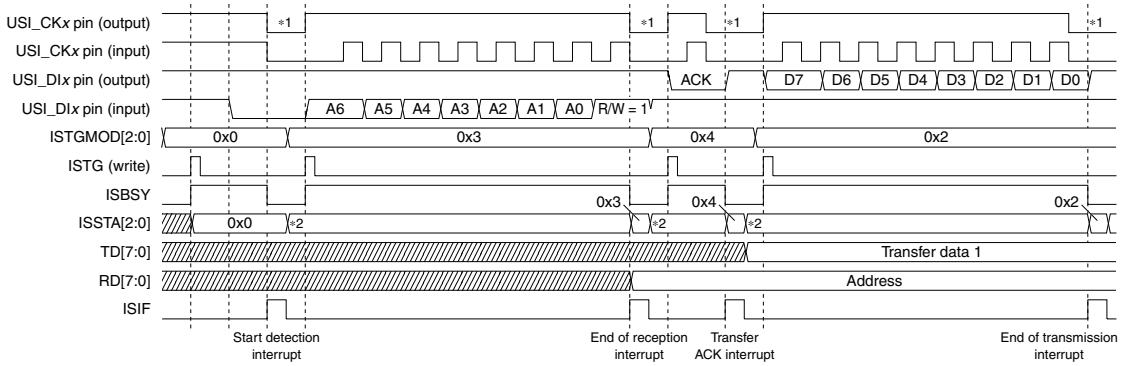
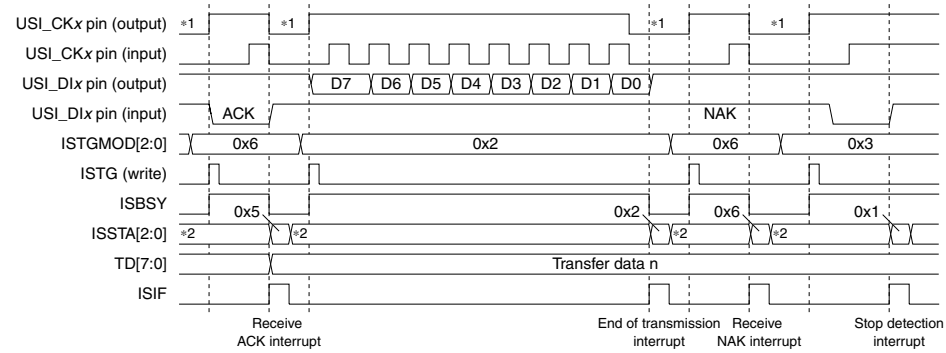


Figure 20.5.3.12 I²C Slave Data Transmission Flow Chart



(1) Start condition → Data transmission



(2) Data transmission → Stop condition

∗1 When the USI_CK*x* input is detected as low after the operation selected by ISTGMOD[2:0] has finished, the USI I²C slave controller pulls down the USI_CK*x* pin to low to places the external I²C master into wait state. This pull-down is canceled to release the I²C master from wait state when the subsequent operation is triggered by ISTG.

∗2 When ISIF is cleared via software, ISSTA[2:0] is also cleared to 0x0.

Figure 20.5.3.13 I²C Slave Data Transmission Timing Chart

Note: The timing chart above shows a basic transfer operation that does not include an actual I²C transfer procedure. See "Receiving control byte in I²C slave mode" in "20.9 Precautions."

(1) Waiting for start condition

I²C data transfer starts when the I²C master device generates a start condition (see Figure 20.5.3.4).

First enable this I²C slave to detect a start condition by setting ISTGMOD[2:0] to 0x0 (default) and writing 1 to ISTG. The I²C controller starts detecting a start condition and sets ISBSY to 1. ISBSY is set to 1 while a start condition is being detected. ISBSY reverts to 0 and ISSTA[2:0] is set to 0x0 when the detection has completed. Check if a start condition is generated by reading ISBSY or using an interrupt.

Note: Other operations cannot be started before a start condition is detected.

(2) Receiving slave address and transfer direction data bit

The I²C master sends the address of the slave device to be communicated and a transfer direction bit (see Figure 20.5.3.5) after it has generated a start condition. Set this I²C slave into receiving status to receive the slave address. To start reception, set ISTGMOD[2:0] to 0x3 and write 1 to ISTG.

This trigger starts sampling clocks input from the USI_CK*x* pin. When clocks are input, the I²C controller loads the USI_DO*x* pin status to the shift register in sync with each clock. The received data is loaded to the receive data buffer (RD[7:0]/USI_RD*x* register) once the 8-bit data has been received in the shift register.

Writing 1 to ISTG sets ISBSY to 1. When the received data is loaded to the receive data buffer, ISBSY reverts to 0 and ISSTA[2:0] is set to 0x3 (end of receive data). An interrupt request can be generated at this point. Read the received data from the receive data buffer using this interrupt.

When a 7-bit address is used, the slave address and transfer direction bit can be obtained in one operation. When a 10-bit address is used, save the first data received in the receive data buffer into the memory and perform data reception again to obtain the remaining address bits.

Check whether the received address is matched to this I²C slave address or not. When they are matched, send back an ACK to the I²C master by setting ISTGMOD[2:0] to 0x4 and write 1 to ISTG. ISBSY is set to 1 while an ACK is being sent and it reverts to 0 when the transmission has completed. An interrupt request can be generated at this point. When an ACK has been sent, ISSTA[2:0] is set to 0x4.

If the received address is not for this I²C slave, abort data reception and return to Step (1) to wait the subsequent start condition.

(3) Data transmission

When the transfer direction bit received with the slave address in Step (2) is 1, start data transmission by the following procedure:

1. Write an 8-bit transmit data to the transmit data buffer (TD[7:0]).
2. Set ISTGMOD[2:0] to 0x2 and ISTG to 1.

This trigger transfers the buffer data to the transmit shift register to start transmission. When clocks are input from the USI_CK*x* pin, the data in the shift register is shifted in sequence with the clock and sent from the USI_DO*x* pin.
Writing 1 to ISTG sets ISBSY to 1. When data in the transmit data buffer is sent to the transmit shift register, ISBSY reverts to 0 and ISSTA[2:0] is set to 0x2 (end of transmit data). An interrupt request can be generated at this point. Write subsequent data to the transmit data buffer to start the following transmission using this interrupt.
However, check that the master device has sent back an ACK or NAK (by setting ISTGMOD[2:0] to 0x6 and ISTG to 1) before starting the following 8-bit data transmission.
ISBSY is set to 1 while an ACK/NAK is being detected and it reverts to 0 when the detection has completed. Receiving an ACK sets ISSTA[2:0] to 0x5; receiving a NAK sets it to 0x6. Check ISSTA[2:0] after confirming ISBSY or using an interrupt. When an ACK has been received, perform data transmission. When a NAK has been received, perform the appropriate handling.

(4) When a stop condition is received
If the ISSTA[2:0] value read during data transmission is 0x1, the I²C master device has generated a stop condition (see Figure 20.5.3.7). In this case, abort data transmission.

Data reception in I²C slave mode

The following describes the data receiving procedure in I²C slave mode.

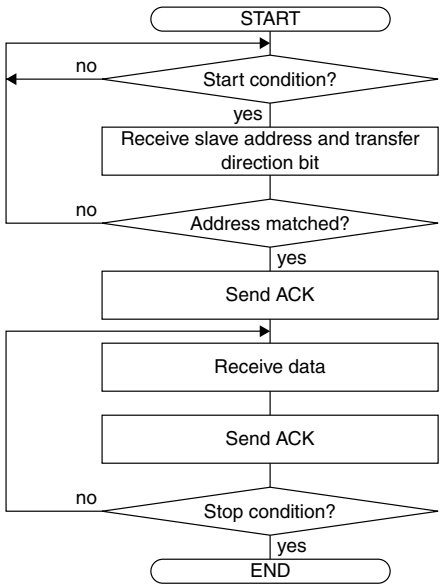
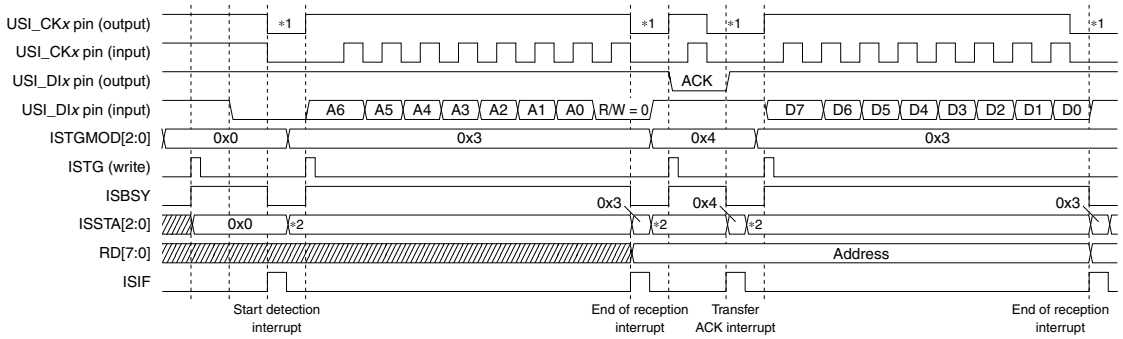
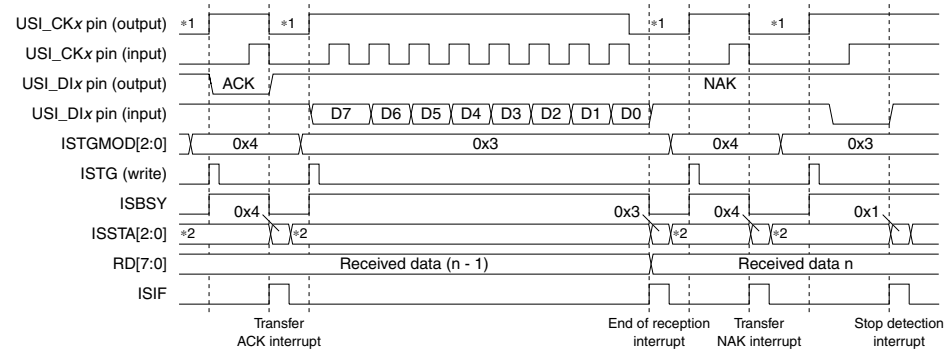


Figure 20.5.3.14 I²C Slave Data Receiving Flow Chart



(1) Start condition → Data reception



(2) Data reception → Stop condition

∗1 When the USI_CK*x* input is detected as low after the operation selected by ISTGMOD[2:0] has finished, the USI I²C slave controller pulls down the USI_CK*x* pin to low to places the external I²C master into wait state. This pull-down is canceled to release the I²C master from wait state when the subsequent operation is triggered by ISTG.

∗2 When ISIF is cleared via software, ISSTA[2:0] is also cleared to 0x0.

Figure 20.5.3.15 I²C Slave Data Receiving Timing Chart

Note: The timing chart above shows a basic transfer operation that does not include an actual I²C transfer procedure. See "Receiving control byte in I²C slave mode" in "20.9 Precautions."

(1) Waiting for start condition
The procedure is the same as that of data transmission in I²C slave mode.

(2) Receiving slave address and transfer direction data bit
The procedure is the same as that of data transmission in I²C slave mode.

(3) Data reception
When the transfer direction bit received with the slave address in Step (2) is 0, start data reception by setting ISTGMOD[2:0] to 0x3 and writing 1 to ISTG.
When clocks are input, the I²C controller loads the USI_DO*x* pin status to the shift register in sync with each clock. The received data is loaded to the receive data buffer (RD[7:0]/USI_RD*x* register) once the 8-bit data has been received in the shift register.
Writing 1 to ISTG sets ISBSY to 1. When the received data is loaded to the receive data buffer, ISBSY reverts to 0 and ISSTA[2:0] is set to 0x3 (end of receive data). An interrupt request can be generated at this point. Read the received data from the receive data buffer using this interrupt.

It is necessary to send back an ACK or NAK to the master device after an 8-bit data has been received. To send back an ACK, set ISTGMOD[2:0] to 0x4 and write 1 to ISTG. To send back a NAK, set ISTGMOD[2:0] to 0x5 and write 1 to ISTG.
ISBSY is set to 1 while an ACK/NAK is being sent and it reverts to 0 when the transmission has completed. An interrupt request can be generated at this point. When an ACK or NAK has been sent, ISSTA[2:0] is set to 0x4.
Repeat an 8-bit data reception and ACK (NAK) transmission for the required number of times.

(4) When a stop condition is received
If the ISSTA[2:0] value read during data reception is 0x1, the I²C master device has generated a stop condition (see Figure 20.5.3.7). In this case, abort data reception.

Clock stretch function

While data is being sent/received, this I²C slave generates a clock stretch status by pulling down the SCL line to low to make a wait request to the master device after an ACK is sent/received until the following data transfer is started.

20.6 Receive Errors

In UART mode, three different receive errors (overrun error, framing error, and parity error) may be detected while receiving data. In SPI and I²C modes, overrun errors may be detected while receiving data.
Since receive errors are interrupt causes, they can be processed by generating interrupts. For more information on interrupt control, see Section 20.7.

Overrun error (all interface modes)

UART mode

An overrun error occurs if the next reception is completed when URDIF is 1 and the receive data buffer (USI_RD*x* register) is not read (an overrun error occurs at the time stop bit has been received).
When an overrun error occurs, the overrun error flag (UOEIF/USI_UIF*x* register) is set to 1. The receiving operation continues even if this error occurs. To reset UOEIF, perform USI software reset (write 0x0 to USIMOD[2:0]/USI_GCFG*x* register) to initialize USI.

SPI mode

An overrun error occurs if data are received successively when SRDIF is 1. While SRDIF is set to 1, the next received data will not be transferred from the shift register to the receive data buffer (the first byte data exists in the receive data buffer and the second byte data exists in the shift register). An overrun error occurs if the third byte data is received in this condition, as the second byte data in the shift register is corrupted (an overrun error occurs at the time the first bit of the third byte is fetched).
When an overrun error occurs, the overrun error flag (SEIF/USI_SIF*x* register) is set to 1. The receiving operation continues even if this error occurs. SEIF is reset by writing 1. To reset an overrun error, write 1 to SEIF and then read the receive data buffer (USI_RD*x* register) twice. The procedure that writes 1 to SEIF and reads USI_RD*x* register twice can be reversed.

I²C master/slave mode

An overrun error occurs when a transmit or receive trigger is issued after two-byte data has been received (the first byte data exists in the receive data buffer and the second byte data exists in the shift register) without the receive data buffer being read.
When an overrun error occurs, the overrun error flag (IMEIF/USI_IMIF*x* register for I²C master mode or ISEIF/USI_ISIF*x* register for I²C slave mode) is set to 1. The receiving operation continues even if this error occurs. IMEIF/ISEIF is reset by writing 1. To reset an overrun error, write 1 to IMEIF/ISEIF and then read the receive data buffer (USI_RD*x* register) twice.

Framing error (UART mode only)

If the stop bit is received as 0 in UART mode, the UART controller determines loss of sync and a framing error occurs. If the stop bit is configured to two bits, only the first bit is checked.
The framing error flag (USEIF/USI_UIF*x* register) is set to 1 if this error occurs. The received data is still transferred to the receive data buffer if this error occurs and the receiving operation continues, but the data cannot be guaranteed, even if no framing error occurs for subsequent data receiving. The framing error flag is reset to 0 by writing 1.

Parity error (UART mode only)

If UPREN/USI_UCFG*x* register has been set to 1 (parity enabled), data received is checked for parity in UART mode. Data received in the shift register is checked for parity when sent to the receive data buffer. The matching is checked against the UPMD/USI_UCFG*x* register setting (odd or even parity). If the result is a non-match, a parity error is issued, and the parity error flag (UPEIF/USI_UIF*x* register) is set to 1. Even if this error occurs, the data received is sent to the receive data buffer, and the receiving operation continues. However, the received data cannot be guaranteed if a parity error occurs. The UPEIF flag is reset to 0 by writing 1.

20.7 USI Interrupts and DMA

This section describes the USI interrupts generated in each interface mode and invoking DMA.
For more information on interrupt processing and DMA transfer, see the "Interrupt Controller (ITC)" chapter and the "DMA Controller (DMAC)" chapter, respectively.

Each USI channel outputs one interrupt signal shared by the all interrupt causes to the interrupt controller (ITC). Inspect the interrupt flags available in each mode to determine the interrupt cause occurred.

20.7.1 Interrupts in UART Mode

The UART mode includes a function for generating the following three different types of interrupts.
- Transmit buffer empty interrupt
- Receive buffer full interrupt
- Receive error interrupt

Transmit buffer empty interrupt

To use this interrupt, set UTDIE/USI_UIE*x* register to 1. If UTDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When transmit data written to the transmit data buffer is transferred to the shift register, the USI module sets UTDIF/USI_UIF*x* register to 1, indicating that the transmit data buffer is empty. If transmit buffer empty interrupts are enabled (UTDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the UTDIF flag in the interrupt handler routine to determine whether the USI (UART mode) interrupt is attributable to a transmit buffer empty. If UTDIF is 1, the next transmit data can be written to the transmit data buffer by the interrupt handler routine.

Receive buffer full interrupt

To use this interrupt, set URDIE/USI_UIE*x* register to 1. If URDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
If a received data is loaded into the receive data buffer, the USI module sets URDIF/USI_UIF*x* register to 1. If receive buffer full interrupts are enabled (URDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the URDIF flag in the interrupt handler routine to determine whether the USI (UART mode) interrupt is attributable to a receive buffer full. If URDIF is 1, the received data can be read from the receive data buffer by the interrupt handler routine. However, be sure to check whether a receive error has occurred or not.

Receive error interrupt

To use this interrupt, set UEIE/USI_UIE*x* register to 1. If UEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
The USI module sets an error flag (UPEIF/USI_UIF*x* register, USEIF/USI_UIF*x* register, or UOEIF/USI_UIF*x* register) to 1 if a parity error, framing error, or overrun error is detected when receiving data. If receive error interrupts are enabled (UEIE = 1), an interrupt request is sent simultaneously to the ITC.
If other interrupt conditions are satisfied, an interrupt occurs. You can inspect the UPEIF, USEIF, and UOEIF flags in the interrupt handler routine to determine whether the USI (UART mode) interrupt was caused by a receive error. If any of the error flags has the value 1, the interrupt handler routine will proceed with error recovery.
To reset an overrun error, perform USI software reset (write 0x0 to USIMOD[2:0]/USI_GCFG*x* register) to initialize USI.

20.7.2 Interrupts in SPI Mode

The SPI master/slave modes include a function for generating the following three different types of interrupts.
- Transmit buffer empty interrupt
- Receive buffer full interrupt
- Receive error interrupt

Transmit buffer empty interrupt

To use this interrupt, set STDIE/USI_SIE*x* register to 1. If STDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When transmit data written to the transmit data buffer is transferred to the shift register, the USI module sets STDIF/USI_SIF*x* register to 1, indicating that the transmit data buffer is empty. If transmit buffer empty interrupts are enabled (STDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the STDIF flag in the interrupt handler routine to determine whether the USI (SPI master/slave mode) interrupt is attributable to a transmit buffer empty. If STDIF is 1, the next transmit data can be written to the transmit data buffer by the interrupt handler routine.

Receive buffer full interrupt

To use this interrupt, set SRDIE/USI_SIE*x* register to 1. If SRDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

If a received data is loaded into the receive data buffer, the USI module sets SRDIF/USI_SIF*x* register to 1. If receive buffer full interrupts are enabled (SRDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the SRDIF flag in the interrupt handler routine to determine whether the USI (SPI master/slave mode) interrupt is attributable to a receive buffer full. If SRDIF is 1, the received data can be read from the receive data buffer by the interrupt handler routine. However, be sure to check whether a receive error has occurred or not.

Receive error interrupt

To use this interrupt, set SEIE/USI_SIE*x* register to 1. If SEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

The USI module sets SEIF/USI_SIF*x* register to 1 if an overrun error is detected when receiving data. If receive error interrupts are enabled (SEIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the SEIF flags in the interrupt handler routine to determine whether the USI (SPI master/slave mode) interrupt was caused by a receive error. If SEIF is 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, clear SEIF by writing 1 and then read the receive data buffer (USI_RD*x* register) twice.

20.7.3 Interrupts in I²C Master Mode

The I²C master mode includes a function for generating the following two different types of interrupts.

- Operation completion interrupt
- Receive error interrupt

Operation completion interrupt

To use this interrupt, set IMIE/USI_IMIE*x* register to 1. If IMIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When the operation that initiated by a software trigger has completed, the USI module sets IMIF/USI_IMIF*x* register to 1. If operation completion interrupts are enabled (IMIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the IMSTA[2:0]/USI_IMIF*x* register in the interrupt handler routine to determine the I²C operation/status that causes the interrupt.

Table 20.7.3.1 I²C Master Status Bits

IMSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been generated.
0x0	Start condition has been generated.

(Default: 0x0)

Receive error interrupt

To use this interrupt, set IMEIE/USI_IMIE*x* register to 1. If IMEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

An overrun error occurs at the time a transmit or receive trigger is issued after two-byte data has been received without reading the receive data buffer.

The USI module sets IMEIF/USI_IMIF*x* register to 1 if an overrun error is detected when receiving data. If receive error interrupts are enabled (IMEIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the IMEIF flags in the interrupt handler routine to determine whether the USI (I^2C master mode) interrupt was caused by a receive error. If IMEIF is 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, clear IMEIF by writing 1, and then read the receive data buffer (USI_RD*x* register) twice.

20.7.4 Interrupts in I^2C Slave Mode

The I^2C slave mode includes a function for generating the following two different types of interrupts.

- Operation completion interrupt
- Receive error interrupt

Operation completion interrupt

To use this interrupt, set ISIE/USI_ISIE*x* register to 1. If ISIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When the operation that initiated by a software trigger has completed, the USI module sets ISIF/USI_ISIF*x* register to 1. If operation completion interrupts are enabled (ISIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the ISSTA[2:0]/USI_ISIF*x* register in the interrupt handler routine to determine the I^2C operation/status that causes the interrupt.

Table 20.7.4.1 I^2C Slave Status Bits

ISSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been detected.
0x0	Start condition has been detected.

(Default: 0x0)

Receive error interrupt

To use this interrupt, set ISEIE/USI_ISIE*x* register to 1. If ISEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

An overrun error occurs at the time a transmit or receive trigger is issued after two-byte data has been received without reading the receive data buffer.

The USI module sets ISEIF/USI_ISIF*x* register to 1 if an overrun error is detected when receiving data. If receive error interrupts are enabled (ISEIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the ISEIF flags in the interrupt handler routine to determine whether the USI (I^2C slave mode) interrupt was caused by a receive error. If ISEIF is 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, clear ISEIF by writing 1, and then read the receive data buffer (USI_RD*x* register) twice.

20.7.5 DMA Transfer

The causes of receive buffer full and transmit buffer empty interrupts in UART and SPI master/slave modes can invoke a DMA. This allows continuous data transmission/reception through DMA transfer between memory and transmit/receive data buffers. These interrupt signals are output to both the ITC and DMAC. Therefore, DMA transfer can be performed without generating any USI interrupt.
The following lists the DMAC channels that allow selection of a USI interrupt cause as the trigger.

USI Ch.0 receive buffer full: DMAC Ch.2
USI Ch.0 transmit buffer empty: DMAC Ch.3
USI Ch.1 receive buffer full: DMAC Ch.4
USI Ch.1 transmit buffer empty: DMAC Ch.5
USI Ch.2 receive buffer full: DMAC Ch.6
USI Ch.2 transmit buffer empty: DMAC Ch.7

For more information on DMA transfer, see the "DMA Controller (DMAC)" chapter.

Note: The USI module cannot invoke a DMA in I²C master and slave mode.

20.8 Control Register Details

Table 20.8.1 List of USI Registers

Address	Register name		Function
0x300400	USI_GCFG0	USI Ch.0 Global Configuration Register	Set interface and MSB/LSB modes
0x300401	USI_TD0	USI Ch.0 Transmit Data Buffer Register	Transmit data buffer
0x300402	USI_RD0	USI Ch.0 Receive Data Buffer Register	Receive data buffer
0x300440	USI_UCFG0	USI Ch.0 UART Mode Configuration Register	Set UART transfer conditions
0x300441	USI_UIE0	USI Ch.0 UART Mode Interrupt Enable Register	Enable/disable UART interrupts
0x300442	USI_UIF0	USI Ch.0 UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
0x300450	USI_SCFG0	USI Ch.0 SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
0x300451	USI_SIE0	USI Ch.0 SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
0x300452	USI_SIF0	USI Ch.0 SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
0x300460	USI_IMTG0	USI Ch.0 I²C Master Mode Trigger Register	Start I²C master operations
0x300461	USI_IMIE0	USI Ch.0 I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
0x300462	USI_IMIF0	USI Ch.0 I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
0x300470	USI_ISTG0	USI Ch.0 I²C Slave Mode Trigger Register	Start I²C slave operations
0x300471	USI_ISIE0	USI Ch.0 I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
0x300472	USI_ISIF0	USI Ch.0 I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status
0x300500	USI_GCFG1	USI Ch.1 Global Configuration Register	Set interface and MSB/LSB modes
0x300501	USI_TD1	USI Ch.1 Transmit Data Buffer Register	Transmit data buffer
0x300502	USI_RD1	USI Ch.1 Receive Data Buffer Register	Receive data buffer
0x300540	USI_UCFG1	USI Ch.1 UART Mode Configuration Register	Set UART transfer conditions
0x300541	USI_UIE1	USI Ch.1 UART Mode Interrupt Enable Register	Enable/disable UART interrupts
0x300542	USI_UIF1	USI Ch.1 UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
0x300550	USI_SCFG1	USI Ch.1 SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
0x300551	USI_SIE1	USI Ch.1 SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
0x300552	USI_SIF1	USI Ch.1 SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
0x300560	USI_IMTG1	USI Ch.1 I²C Master Mode Trigger Register	Start I²C master operations
0x300561	USI_IMIE1	USI Ch.1 I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
0x300562	USI_IMIF1	USI Ch.1 I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
0x300570	USI_ISTG1	USI Ch.1 I²C Slave Mode Trigger Register	Start I²C slave operations
0x300571	USI_ISIE1	USI Ch.1 I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
0x300572	USI_ISIF1	USI Ch.1 I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status
0x300700	USI_GCFG2	USI Ch.2 Global Configuration Register	Set interface and MSB/LSB modes
0x300701	USI_TD2	USI Ch.2 Transmit Data Buffer Register	Transmit data buffer
0x300702	USI_RD2	USI Ch.2 Receive Data Buffer Register	Receive data buffer
0x300740	USI_UCFG2	USI Ch.2 UART Mode Configuration Register	Set UART transfer conditions
0x300741	USI_UIE2	USI Ch.2 UART Mode Interrupt Enable Register	Enable/disable UART interrupts
0x300742	USI_UIF2	USI Ch.2 UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
0x300750	USI_SCFG2	USI Ch.2 SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
0x300751	USI_SIE2	USI Ch.2 SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
0x300752	USI_SIF2	USI Ch.2 SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
0x300760	USI_IMTG2	USI Ch.2 I²C Master Mode Trigger Register	Start I²C master operations
0x300761	USI_IMIE2	USI Ch.2 I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
0x300762	USI_IMIF2	USI Ch.2 I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
0x300770	USI_ISTG2	USI Ch.2 I²C Slave Mode Trigger Register	Start I²C slave operations
0x300771	USI_ISIE2	USI Ch.2 I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
0x300772	USI_ISIF2	USI Ch.2 I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status

The USI registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

USI Ch.*x* Global Configuration Registers (USI_GCFG*x*)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
USI Ch.*x* Global Configuration Register (USI_GCFG*x*)	0x300400 0x300500 0x300700 (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3	LSBFST	MSB/LSB first mode select	1 MSB first	0 LSB first	0	R/W	
		D2–0	USIMOD [2:0]	Interface mode configuration	USIMOD[2:0]	I/F mode	0x0	R/W	
					0x7–0x6	reserved			
					0x5	I²C slave			
					0x4	I²C master			
					0x3	SPI slave			
					0x2	SPI master			
					0x1	UART			
					0x0	Software reset			

Note: This register must be configured before setting other USI registers.

D[7:4] Reserved

D3 LSBFST: MSB/LSB First Mode Select Bit

Selects whether serial data will be transferred from the MSB or LSB.
1 (R/W): MSB first
0 (R/W): LSB first (default)

This setting affects all interface modes.

D[2:0] USIMOD[2:0]: Interface Mode Configuration Bits

Selects an interface mode.

Table 20.8.2 Interface Mode Selection

USIMOD[2:0]	Interface mode
0x5	I²C slave
0x4	I²C master
0x3	SPI slave
0x2	SPI master
0x1	UART
0x0	Software reset

(Default: 0x0)

Perform software reset (set USIMOD[2:0] to 0x0) and then set the interface mode before changing other USI configurations.

Notes: 1. Setting USIMOD[2:0] to 0x0 (software reset) resets all the USI register contents to the initial values.

2. The USI transmitter and receiver circuits will be able to operate after USIMOD[2:0] is set to a value (0x1 to 0x5) other than 0x0.

3. The USI registers for an interface mode are available only when USIMOD[2:0] is set to the mode setting value.

Table 20.8.3 Corresponding Between USIMOD[2:0] Settings and Available Registers

USI register	USIMOD[2:0]				
	0x1	0x2	0x3	0x4	0x5
I²C slave mode registers	–	–	–	–	Available
I²C master mode registers	–	–	–	Available	–
SPI master/slave mode registers	–	Available	Available	–	–
UART mode registers	Available	–	–	–	–

USI Ch.*x* Transmit Data Buffer Registers (USI_TD*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x*	**0x300401**	D7–0	**TD[7:0]**	USI transmit data buffer	0x0 to 0xff	0x0	R/W	
Transmit Data	**0x300501**			TD7 = MSB				
Buffer Register	**0x300701**			TD0 = LSB				
(USI_TD*x*)	(8 bits)							

D[7:0] TD[7:0]: USI Transmit Data Buffer Bits

Sets transmit data to be written to the transmit data buffer. (Default: 0x0)

In UART and SPI master modes, transmission begins immediately after writing data to this register. In SPI slave mode, transmission will begin when the clock is input from the SPI master device.

In I^2C master/slave mode, transmission begins by the software trigger for data transmission.

The data written to this register is converted into serial data through the shift register and is output from the USI_DO*x* pin with the bit set to 1 as high level and the bit set to 0 as low level.

A transmit buffer empty interrupt can be generated when data written to this register has been transferred to the shift register. The subsequent transmit data can then be written, even while data is being sent.

USI Ch.*x* Receive Data Buffer Registers (USI_RD*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x*	**0x300402**	D7–0	**RD[7:0]**	USI receive data buffer	0x0 to 0xff	0x0	R	
Receive Data	**0x300502**			RD7 = MSB				
Buffer Register	**0x300702**			RD0 = LSB				
(USI_RD*x*)	(8 bits)							

D[7:0] RD[7:0]: USI Receive Data Buffer Bits

Contains the received data. (Default: 0x0)

Serial data input from the USI_DI*x* pin is converted to parallel, with the high level bit set to 1 and the low level bit set to 0, and then it is loaded to this register.

A receive buffer full interrupt can be generated when the data received in the shift register has been loaded to this register. Data can then be read until subsequent data is received.

This register is read-only.

USI Ch.*x* UART Mode Configuration Registers (USI_UCFG*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.*x*	**0x300440**	D7–4	**–**	reserved	–				–	–	0 when being read.
UART Mode	**0x300540**	D3	**UCHLN**	Character length select	1	8 bits	0	7 bits	0	R/W	
Configuration	**0x300740**	D2	**USTPB**	Stop bit select	1	2 bits	0	1 bit	0	R/W	
Register		D1	**UPMD**	Parity mode select	1	Even	0	Odd	0	R/W	
(USI_UCFG*x*)	(8 bits)	D0	**UPREN**	Parity enable	1	With parity	0	No parity	0	R/W	

Note: This register is effective only in UART mode. Configure USI to UART mode before setting this register.

D[7:4] Reserved

D3 UCHLN: Character Length Select Bit

Selects the serial transfer data length.

1 (R/W): 8 bits

0 (R/W): 7 bits (default)

When 7-bit data length is selected, D7 in the transmit data buffer is ignored and D7 in the receive data buffer is always set to 0.

D2 USTPB: Stop Bit Select Bit

Selects the stop bit length.

1 (R/W): 2 bits

0 (R/W): 1 bit (default)

Writing 1 to USTPB selects 2 stop bits; writing 0 to it selects 1 bit. The start bit is fixed at 1 bit.

D1 UPMD: Parity Mode Select Bit

Selects the parity mode.
1 (R/W): Even parity
0 (R/W): Odd parity (default)

Parity checking and parity bit addition are enabled only when UPREN is set to 1. The UPMD setting is disabled if UPREN is 0.

D0 UPREN: Parity Enable Bit

Enables the parity function.
1 (R/W): With parity
0 (R/W): No parity (default)

UPREN is used to select whether received data parity checking is performed and whether a parity bit is added to transmit data. Setting UPREN to 1 parity-checks the received data. A parity bit is automatically added to the transmit data. If UPREN is set to 0, no parity bit is checked or added.

USI Ch.*x* UART Mode Interrupt Enable Registers (USI_UIE*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.*x* UART Mode Interrupt Enable Register (USI_UIE*x*)	0x300441 0x300541 0x300741 (8 bits)	D7–3	–	reserved	–				–	–	0 when being read.
		D2	UEIE	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	URDIE	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	UTDIE	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in UART mode. Configure USI to UART mode before this register can be used.

D[7:3] Reserved

D2 UEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when a receive error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process receive errors using interrupts.

D1 URDIE: Receive Buffer Full Interrupt Enable Bit

Enables interrupt requests to the ITC when received data is loaded to the receive data buffer.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to read received data using interrupts.

D0 UTDIE: Transmit Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC when data written to the transmit data buffer is sent to the shift register (i.e. when data transmission begins).
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the transmit data buffer using interrupts.

USI Ch.*x* UART Mode Interrupt Flag Registers (USI_UIF*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.*x* UART Mode Interrupt Flag Register (USI_UIF*x*)	0x300442 0x300542 0x300742 (8 bits)	D7	–	reserved	–				–	–	0 when being read.
		D6	URBSY	Receive busy flag	1	Busy	0	Idle	0	R	
		D5	UTBSY	Transmit busy flag	1	Busy	0	Idle	0	R	
		D4	UPEIF	Parity error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D3	USEIF	Framing error flag	1	Error	0	Normal	0	R/W	
		D2	UOEIF	Overrun error flag	1	Error	0	Normal	0	R/W	
		D1	URDIF	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	UTDIF	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	

Note: This register is effective only in UART mode. Configure USI to UART mode before this register can be used.

D7 Reserved

D6 URBSY: Receive Busy Flag Bit

Indicates the receive shift register status.

1 (R): Busy
0 (R): Idle (default)

URBSY is set to 1 when the first start bit is detected (when data reception begins) and is reset to 0 when the data received in the shift register is loaded into the receive data buffer. Inspect URBSY to determine whether the receiving circuit is operating or at standby.

D5 UTBSY: Transmit Busy Flag Bit

Indicates the USI status in UART mode.

1 (R): Busy
0 (R): Idle (default)

UTBSY switches to 1 when transmit data is written to the transmit buffer and reverts to 0 after both the shift register and transmit buffer become empty.

D4 UPEIF: Parity Error Flag Bit

Indicates whether a parity error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

UPEIF is set to 1 when a parity error occurs. At the same time a receive error interrupt request is sent to the ITC if UEIE/USI_UIE*x* register is 1. Parity checking is enabled only when UPREN/USI_UCFG*x* register is set to 1 and is performed when received data is transferred from the shift register to the receive data buffer. UPEIF is reset by writing 1.

D3 USEIF: Framing Error Flag Bit

Indicates whether a framing error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

USEIF is set to 1 when a framing error occurs. At the same time a receive error interrupt request is sent to the ITC if UEIE/USI_UIE*x* register is 1. A framing error occurs when data is received with the stop bit set to 0. USEIF is reset by writing 1.

D2 UOEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

UOEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if UEIE/USI_UIE*x* register is 1. An overrun error occurs if the next reception is completed when URDIF is 1 and the receive data buffer (USI_RD*x* register) is not read (an overrun error occurs at the time stop bit has been received). To reset UOEIF, perform USI software reset (write 0x0 to USIMOD[2:0]/USI_GCFG*x* register) to initialize USI.

D1 URDIF: Receive Buffer Full Flag Bit

Indicates the receive data buffer status.

1 (R): Data full
0 (R): No data (default)
1 (W): Reset to 0
0 (W): Ignored

URDIF is set to 1 when data received in the shift register is sent to the receive data buffer (when receiving is completed), indicating that the data can be read. At the same time a receive buffer full interrupt request is sent to the ITC if URDIE/USI_UIE*x* register is 1. URDIF is reset by writing 1.

D0 UTDIF: Transmit Data Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Empty (default)
0 (R): Data exists
1 (W): Reset to 0
0 (W): Ignored

UTDIF is set to 1 when the transmit data written to the transmit data buffer is transferred to the shift register (when transmission starts), indicating that the next transmit data can be written to. At the same time a transmit buffer empty interrupt request is sent to the ITC if UTDIE/USI_UIE*x* register is 1. UTDIF is reset by writing 1.

USI Ch.*x* SPI Master/Slave Mode Configuration Registers (USI_SCFG*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.*x* SPI Master/Slave Mode Configuration Register (USI_SCFG*x*)	0x300450 0x300550 0x300750 (8 bits)	D7–6	–	reserved	–				–	–	0 when being read.
		D5	SCMD	Command bit (for 9-bit data)	1	High	0	Low	0	R/W	
		D4	SCHLN	Character length select	1	9 bits	0	8 bits	0	R/W	
		D3	SCPHA	Clock phase select	1	Phase 1	0	Phase 0	0	R/W	
		D2	SCPOL	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	–	reserved	–				–	–	Do not set to 1.
		D0	SFSTMOD	Fast mode select	1	Fast	0	Normal	0	R/W	

Note: This register is effective only in SPI master and slave modes. Configure USI to SPI master/slave mode before this register can be used.

D[7:6] Reserved

D5 SCMD: Command Bit (for 9-bit data in SPI master mode)

Sets the command bit value for 9-bit data (see SCHLN below).

1 (R/W): High
0 (R/W): Low (default)

D4 SCHLN: Character Length Select Bit (for SPI master mode)

Selects the serial transfer data length.

1 (R/W): 9 bits
0 (R/W): 8 bits (default)

In 9-bit mode, 8-bit data is prefixed with a command bit (1 bit). The command bit is used for controlling the SPI LCD controller connected to the USI. The command bit value to be transmitted can be specified using SCMD.

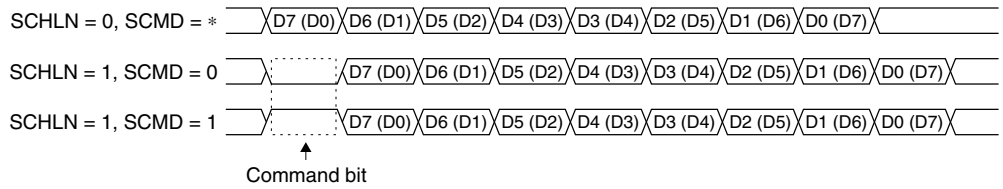


Figure 20.8.1 9-bit Transfer Data Format in SPI Master Mode

This bit is effective only in SPI master mode. The data length in SPI slave mode is fixed at 8 bits.

D3 SCPHA: Clock Phase Select Bit

Selects the SPI clock phase.

1 (R/W): Phase 1
0 (R/W): Phase 0 (default)

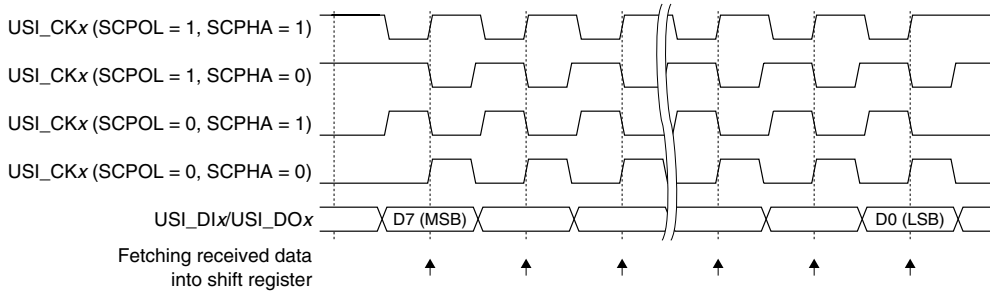
Set the data transfer timing together with SCPOL. (See Figure 20.8.2.)

D2 SCPOL: Clock Polarity Select Bit

Selects the SPI clock polarity.
1 (R/W): Active low
0 (R/W): Active high (default)

Set the data transfer timing together with SCPHA. (See Figure 20.8.2.)

Master mode



Slave mode

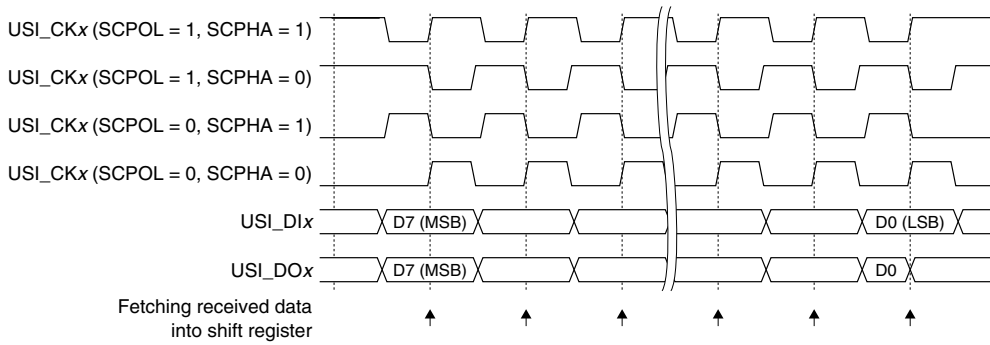


Figure 20.8.2 Clock and Data Transfer Timing

D1 Reserved (Do not set to 1.)

D0 SFSTMOD: Fast Mode Select Bit (for SPI master mode)

Selects Fast mode.
1 (R/W): Fast mode
0 (R/W): Normal mode (default)

In SPI master mode, either normal or fast clock mode can be selected using SFSTMOD. Setting SFSTMOD to 0 (default) places the USI into normal mode and the USI generates the transfer clock by dividing the T8F output by 2. Setting SFSTMOD to 1 places the USI into fast mode and the USI uses PCLK supplied from the CMU directly as the transfer clock. The fast mode does not use the T8F.
The SPI slave mode uses the T8F output clock for generating the sampling clock.

USI Ch.*x* SPI Master/Slave Mode Interrupt Enable Registers (USI_SIE*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.*x* SPI Master/Slave Mode Interrupt Enable Register (USI_SIE*x*)	0x300451 0x300551 0x300751 (8 bits)	D7–3	–	reserved	–				–	–	0 when being read.
		D2	SEIE	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	SRDIE	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	STDIE	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in SPI master and slave modes. Configure USI to SPI master/slave mode before this register can be used.

D[7:3] Reserved

D2 SEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when an overrun error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process overrun errors using interrupts.

D1 SRDIE: Receive Buffer Full Interrupt Enable Bit

Enables interrupt requests to the ITC when received data is loaded to the receive data buffer.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to read received data using interrupts.

D0 STDIE: Transmit Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC when data written to the transmit data buffer is sent to the shift register (i.e. when data transmission begins).
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the transmit data buffer using interrupts.

USI Ch.*x* SPI Master/Slave Mode Interrupt Flag Registers (USI_SIF*x*)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.*x* SPI Master/Slave Mode Interrupt Flag Register (USI_SIF*x*)	0x300452 0x300552 0x300752 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SSIF	Transfer busy flag (master)	1	Busy	0	Idle	0	R	
				ss signal low flag (slave)	1	ss = H	0	ss = L			
		D2	SEIF	Overrun error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D1	SRDIF	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	STDIF	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	

Note: This register is effective only in SPI master and slave modes. Configure USI to SPI master/slave mode before this register can be used.

D[7:4] Reserved

D3 SSIF: Transfer Busy Flag Bit (Master Mode)/ss Signal Low Flag Bit (Slave Mode)

Master mode

Indicates the SPI transfer status.
1 (R): Operating
0 (R): Standby (default)

SSIF is set to 1 when the SPI starts data transfer in master mode and is maintained at 1 while transfer is underway. It is cleared to 0 once the transfer is completed.

Slave mode

Indicates the slave select (USI_CS*x*) signal status.
1 (R): High level (this SPI is not selected)
0 (R): Low level (this SPI is selected) (default)

SSIF is set to 0 when the master device asserts the slave select (USI_CS*x*) signal to select this SPI controller (slave device). It is returned to 1 when the master device clears the SPI controller selection by negating the slave select (USI_CS*x*) signal.

D2 SEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.
1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

SEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if SEIE/USI_SIE*x* register is 1. An overrun error occurs if data are received successively when SRDIF is 1. While SRDIF is set to 1, the next received data will not be transferred from the shift register to the receive data buffer (the first byte data exists in the receive data buffer and the second byte data exists in the shift register). An overrun error occurs if the third byte data is received in this condition, as the second byte data in the shift register is corrupted (an overrun error occurs at the time the first bit of the third byte is fetched).
SEIF is reset by writing 1. To reset an overrun error, write 1 to SEIF and then read the receive data buffer (USI_RD*x* register) twice. The procedure that writes 1 to SEIF and reads USI_RD*x* register twice can be reversed.

D1 SRDIF: Receive Buffer Full Flag Bit

Indicates the receive data buffer status.

1 (R): Data full
0 (R): No data (default)
1 (W): Reset to 0
0 (W): Ignored

SRDIF is set to 1 when data received in the shift register is sent to the receive data buffer (when receiving is completed), indicating that the data can be read. At the same time a receive buffer full interrupt request is sent to the ITC if SRDIE/USI_SIE*x* register is 1. SRDIF is reset by writing 1.

D0 STDIF: Transmit Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Empty (default)
0 (R): Data exists
1 (W): Reset to 0
0 (W): Ignored

STDIF is set to 1 when the transmit data written to the transmit data buffer is transferred to the shift register (when transmission starts), indicating that the next transmit data can be written to. At the same time a transmit buffer empty interrupt request is sent to the ITC if STDIE/USI_SIE*x* register is 1. STDIF is reset by writing 1.

USI Ch.*x* I²C Master Mode Trigger Registers (USI_IMTG*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x* I²C Master Mode Trigger Register (USI_IMTG*x*)	0x300460 0x300560 0x300760 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	IMTG	I²C master operation trigger	1 Trigger / 0 Ignored 1 Waiting / 0 Finished	0	W R	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	IMTGMOD [2:0]	I²C master trigger mode select	IMTGMOD[2:0] / Trigger mode: 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data 0x2 Transmit data 0x1 Stop condition 0x0 Start condition	0x0	R/W	

Note: This register is effective only in I²C master mode. Configure USI to I²C master mode before this register can be used.

D[7:5] Reserved

D4 IMTG: I²C Master Operation Trigger Bit

Starts an I²C master operation.

1 (W): Trigger
0 (W): Ignored
1 (R): Waiting for starting operation
0 (R): Trigger has finished (default)

Select an I²C master operation using IMTGMOD[2:0] and write 1 to IMTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

D3 Reserved

D[2:0] IMTGMOD[2:0]: I²C Master Trigger Mode Select Bits

Selects an I²C master operation.

Table 20.8.4 Trigger List in I²C Master Mode

IMTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception
0x2	Data transmission
0x1	Stop condition
0x0	Start condition

(Default: 0x0)

USI Ch.*x* I²C Master Mode Interrupt Enable Registers (USI_IMIE*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x* I²C Master Mode Interrupt Enable Register (USI_IMIE*x*)	0x300461 0x300561 0x300761 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	IMEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	IMIE	Operation completion int. enable	1 Enable 0 Disable	0	R/W	

Note: This register is effective only in I²C master mode. Configure USI to I²C master mode before this register can be used.

D[7:2] Reserved

D1 IMEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when an overrun error occurs.

1 (R/W): Enabled

0 (R/W): Disabled (default)

Set this bit to 1 to process overrun errors using interrupts.

D0 IMIE: Operation Completion Interrupt Enable Bit

Enables interrupt requests to the ITC when the triggered operation has completed.

1 (R/W): Enabled

0 (R/W): Disabled (default)

Set this bit to 1 to confirm whether the triggered operation has completed or not using interrupts.

USI Ch.*x* I²C Master Mode Interrupt Flag Registers (USI_IMIF*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x* I²C Master Mode Interrupt Flag Register (USI_IMIF*x*)	0x300462 0x300562 0x300762 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	IMBSY	I²C master busy flag	1 Busy 0 Standby	0	R	
		D4–2	IMSTA[2:0]	I²C master status	IMSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop generated 0x0 Start generated	0x0	R	
		D1	IMEIF	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	IMIF	Operation completion flag	1 Completed 0 Not completed	0	R/W	

Note: This register is effective only in I²C master mode. Configure USI to I²C master mode before this register can be used.

D[7:6] Reserved

D5 IMBSY: I²C Master Busy Flag Bit

Indicates the I²C master operation status.

1 (R): Busy
0 (R): Standby (default)

Writing 1 to IMTG/USI_IMTG*x* register (starting an I²C master operation) sets IMBSY to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, IMBSY is reset to 0.

D[4:2] IMSTA[2:0]: I²C Master Status Bits

Indicates the I²C master status.

Table 20.8.5 I²C Master Status Bits

IMSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been generated.
0x0	Start condition has been generated.

(Default: 0x0)

When an operation completion interrupt occurs, read IMSTA[2:0] to check the operation that has been finished. IMSTA[2:0] is automatically reset to 0x0 by writing 1 to IMIF.

D1 IMEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

IMEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if IMEIE/USI_IMIE*x* register is 1.
An overrun error occurs when a transmit or receive trigger is issued after two-byte data has been received (the first byte data exists in the receive data buffer and the second byte data exists in the shift register) without the receive data buffer being read.
IMEIF is reset by writing 1.
To reset an overrun error, clear IMEIF by writing 1, and then read the receive data buffer (USI_RD*x* register) twice.

D0 IMIF: Operation Completion Flag Bit

Indicates whether the triggered operation has completed or not.

1 (R): Completed
0 (R): Not completed (default)
1 (W): Reset to 0
0 (W): Ignored

IMIF is set to 1 when the operation that is specified and triggered using the USI_IMTG*x* register has completed. At the same time an operation completion interrupt request is sent to the ITC if IMIE/USI_IMIE*x* register is 1. IMIF is reset by writing 1.

USI Ch.*x* I²C Slave Mode Trigger Registers (USI_ISTG*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x* I²C Slave Mode Trigger Register (USI_ISTG*x*)	**0x300470** **0x300570** **0x300770** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**ISTG**	I²C slave operation trigger	1 Trigger 0 Ignored 1 Waiting 0 Finished	0	W R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**ISTGMOD [2:0]**	I²C slave trigger mode select	ISTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data/ Detect stop 0x2 Transmit data 0x1 reserved 0x0 Wait for start	0x0	R/W	

Note: This register is effective only in I²C slave mode. Configure USI to I²C slave mode before this register can be used.

D[7:5] Reserved

D4 ISTG: I²C Slave Operation Trigger Bit

Starts an I²C slave operation.

1 (W): Trigger
0 (W): Ignored
1 (R): Waiting for starting operation
0 (R): Trigger has finished (default)

Select an I²C slave operation using ISTGMOD[2:0] and write 1 to ISTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

D3 Reserved

D[2:0] ISTGMOD[2:0]: I²C Slave Trigger Mode Select Bits

Selects an I²C slave operation.

Table 20.8.6 Trigger List in I²C Slave Mode

ISTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception/stop condition detection
0x2	Data transmission
0x1	Reserved
0x0	Wait for start condition

(Default: 0x0)

USI Ch.*x* I²C Slave Mode Interrupt Enable Registers (USI_ISIE*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x* I²C Slave Mode Interrupt Enable Register (USI_ISIE*x*)	**0x300471** **0x300571** **0x300771** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**ISEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**ISIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	

Note: This register is effective only in I²C slave mode. Configure USI to I²C slave mode before this register can be used.

D[7:2] Reserved

D1 ISEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when an overrun error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process overrun errors using interrupts.

D0 ISIE: Operation Completion Interrupt Enable Bit

Enables interrupt requests to the ITC when the triggered operation has completed.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to confirm whether the triggered operation has completed or not using interrupts.

USI Ch.*x* I²C Slave Mode Interrupt Flag Registers (USI_ISIF*x*)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.*x* I²C Slave Mode Interrupt Flag Register (USI_ISIF*x*)	0x300472 0x300572 0x300772 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	ISBSY	I²C slave busy flag	1 Busy 0 Standby	0	R	
		D4–2	ISSTA[2:0]	I²C slave status	ISSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop detected 0x0 Start detected	0x0	R	
		D1	ISEIF	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	ISIF	Operation completion flag	1 Completed 0 Not completed	0	R/W	

Note: This register is effective only in I²C slave mode. Configure USI to I²C slave mode before this register can be used.

D[7:6] Reserved

D5 ISBSY: I²C Slave Busy Flag Bit

Indicates the I²C slave operation status.
1 (R): Busy
0 (R): Standby (default)

Writing 1 to ISTG/USI_ISTG*x* register (starting an I²C slave operation) sets ISBSY to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, ISBSY is reset to 0.

D[4:2] ISSTA[2:0]: I²C Slave Status Bits

Indicates the I²C slave status.

Table 20.8.7 I²C Slave Status Bits

ISSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been detected.
0x0	Start condition has been detected.

(Default: 0x0)

When an operation completion interrupt occurs, read ISSTA[2:0] to check the operation that has been finished. ISSTA[2:0] is automatically reset to 0x0 by writing 1 to ISIF.

D1 ISEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

ISEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if ISEIE/USI_ISIE*x* register is 1. An overrun error occurs when a transmit or receive trigger is issued after two-byte data has been received (the first byte data exists in the receive data buffer and the second byte data exists in the shift register) without the receive data buffer being read.
ISEIF is reset by writing 1.
To reset an overrun error, clear ISEIF by writing 1, and then read the receive data buffer (USI_RD*x* register) twice.

D0 ISIF: Operation Completion Flag Bit

Indicates whether the triggered operation has completed or not.

1 (R): Completed
0 (R): Not completed (default)
1 (W): Reset to 0
0 (W): Ignored

ISIF is set to 1 when the operation that is specified and triggered using the USI_ISTG*x* register has completed. At the same time an operation completion interrupt request is sent to the ITC if ISIE/USI_ISIE*x* register is 1. ISIF is reset by writing 1.

20.9 Precautions

Interface mode setting

Be sure to perform software reset (USIMOD[2:0]/USI_GCFG*x* register = 0x0) and set the interface mode (USIMOD[2:0]/USI_GCFG*x* register = 0x1 to 0x5) before changing other USIL configurations.

Receiving control byte in I²C slave mode

The external I²C master device sends a control byte to the I²C slave device when an ACK has been received after sending a slave address. The subsequent operations of the slave device are determined by the control byte.

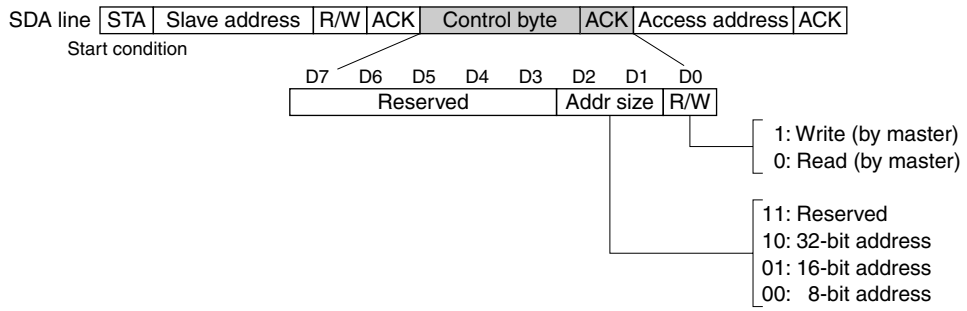


Figure 20.9.1 Control Byte Sent from I²C Master

I²C master write (data receiving from master)

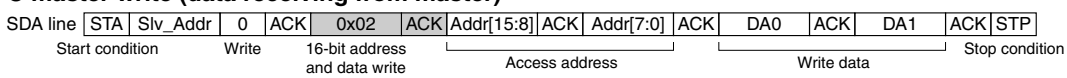


Figure 20.9.2 I²C Master Write (Data Receiving from Master)

The control byte specifies the access address size and writing operations. The received data that follow the control byte should be used as the address and the data to be written according to the access address size.

I²C master read (data transmission to master)

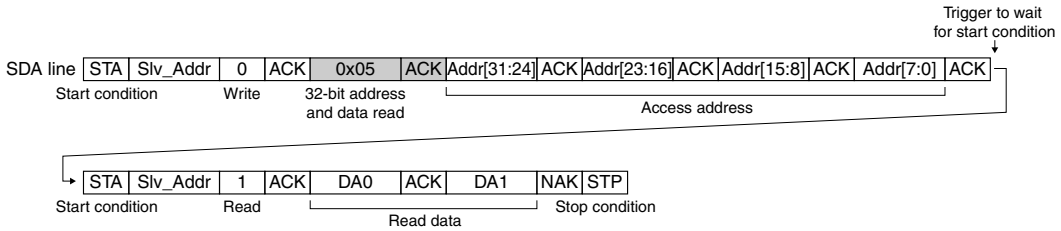


Figure 20.9.3 I²C Master Read (Data Transmission to Master)

The master sends the access address following the control byte. Perform data reception for the control byte and address data to determine the address from which transmit data is read. After sending an ACK for Addr 0, set ISTGMOD[2:0]/USIL_ISTG*x* register to 0x0 and ISTG/USIL_ISTG*x* register to 1 to wait for a start condition that will be sent from the master for reading data (for the slave to sent the read data).

21 Universal Serial Interface with LCD Interface (USIL)

21.1 USIL Module Overview

The S1C33L27 incorporates a USIL module that can be configured as a UART, SPI, I²C, LCD SPI, and LCD parallel interface unit by the software switch.
The following shows the main features of USIL:

- Supports seven interface modes: UART, SPI master, SPI slave, I²C master, I²C slave, LCD SPI, and LCD parallel modes.
- Contains one-byte receive data buffer and one-byte transmit buffer.
- Supports both MSB first and LSB first modes.
- UART mode
  - Character length: 7 or 8 bits
  - Parity mode: even, odd, or no parity
  - Stop bit: 1 or 2 bits
  - Start bit: 1 bit fixed
  - Parity error, framing error, and overrun error detectable
  - Can generate receive buffer full, transmit buffer empty, and receive error interrupts.
  - Supports DMA transfer.
- SPI master/slave mode
  - Data length: 8 bits fixed
  - Supports both fast and normal modes (master mode), or normal mode only (slave mode).
  - Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
  - Can generate receive buffer full, transmit buffer empty, and receive error interrupts.
  - Supports DMA transfer.
- I²C master/slave mode
  - 7-bit addressing mode (10-bit addressing is possible by software control.)
  - Supports single master configuration only (master mode).
  - Supports clock stretch/wait functions.
  - Can generate operation (start/stop, data transfer, ACK/NAK transfer) completion interrupts and receive error interrupts.
- LCD SPI mode
  - Data format: 8 bits, 16 bits, 18 bits (4 data format), and 24 bits + CMD bit
  - Supports normal mode only.
  - Supports transmission only.
  - Data transfer timing (clock phase and polarity variations) is selectable from among 4 types.
  - Can generate transmit buffer empty interrupts.
  - Supports DMA transfer.
- LCD parallel mode
  - Data bus width: 8 bits
  - Control signals: A0, write, read and chip select signals can be output.
  - Supports byte read and byte write.
  - Configurable access timing parameters (setup, hold, and wait cycles)
  - Can generate write buffer empty and read buffer full interrupts.
  - Supports DMA transfer.

Figure 21.1.1 shows the USIL configuration.

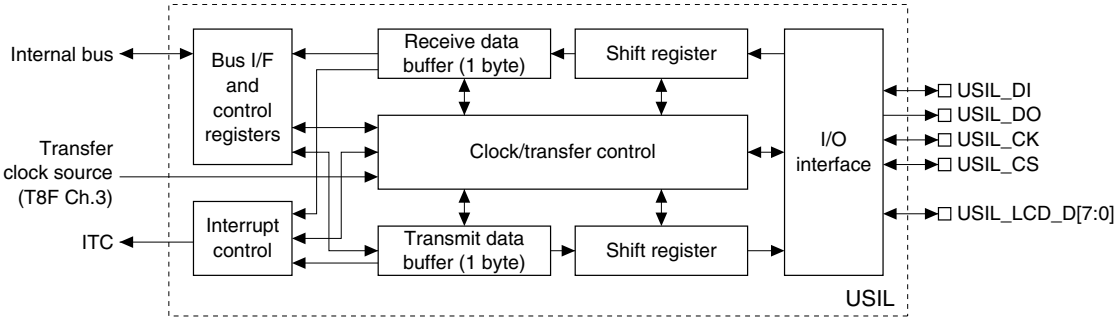


Figure 21.1.1 USIL Configuration

21.2 USIL Pins

Table 21.2.1 lists the USIL input/output pins.

Table 21.2.1 List of USIL Pins

Pin name	USIL mode	Signal name	I/O	Function
USIL_DI	Software reset	–	I	–
	UART	uart_rx	I	Data input pin Inputs serial data sent from an external serial device.
	SPI master	spi_di	I	
	SPI slave	spi_di	I	
	I²C master	i2c_sda	I/O	Data input/output pin Inputs/outputs serial data from/to the I²C bus. (∗1)
	I²C slave	i2c_sda	I/O	
	LCD SPI	lcds_a0	O	A0 signal output pin Outputs the A0 signal to the LCD driver/panel.
	LCD parallel	lcdp_a0	O	
USIL_DO	Software reset	–	I	–
	UART	uart_tx	O	Data output pin Outputs serial data sent to an external serial device.
	SPI master	spi_do	O	
	SPI slave	spi_do	O	
	I²C master	–	–	Not used
	I²C slave	–	–	
	LCD SPI	lcds_do	O	Data output pin Outputs serial data sent to the LCD driver/panel.
	LCD parallel	lcdp_wr	O	Write signal output pin Outputs the write signal to the LCD driver/panel.
USIL_CK	Software reset	–	I	–
	UART	–	–	Not used
	SPI master	spi_ck	O	Clock output pin Outputs the SPI clock.
	SPI slave	spi_ck	I	Clock input pin Inputs an external clock.
	I²C master	i2c_sck	I/O	SCL input/output pin Inputs SCL line status from the I²C bus. Also outputs the I²C clock.
	I²C slave	i2c_sck	I/O	SCL input/output pin Inputs SCL line status from the I²C bus. Also outputs a clock stretch condition.
	LCD SPI	lcds_ck	O	Clock output pin Outputs the SPI clock.
	LCD parallel	lcdp_rd	O	Read signal output pin Outputs the read signal to the LCD driver/panel.
USIL_CS	Software reset	–	I	–
	UART	–	–	Not used
	SPI master	–	–	
	SPI slave	#spi_ss	I	SPI slave select signal input pin Low level input to this pin selects USIL (in SPI slave mode) as an SPI slave device.
	I²C master	i2c_sda	I/O	Data input/output pin Inputs/outputs serial data from/to the I²C bus. (∗1)
	I²C slave	i2c_sda	I/O	
	LCD SPI	–	–	Not used
	LCD parallel	lcdp_cs	O	Chip select signal output pin Outputs the chip select signal to the LCD driver/panel.

Pin name	USIL mode	Signal name	I/O	Function
USIL_LCD_ D[7:0]	Software reset	–	I	–
	UART	–	–	Not used
	SPI master	–	–	
	SPI slave	–	–	
	I²C master	–	–	
	I²C slave	–	–	
	LCD SPI	–	–	
	LCD parallel	lcdp_da[7:0]	I/O	8-bit data input/output pins Inputs/outputs 8-bit parallel data from/to the LCD driver/panel.

∗1: When USIL is configured to I²C master or slave mode, either the USIL_DI pin or the USIL_CS pin can be used as the data input/output pin. Note, however, that both the USIL_DI and USIL_CS pins cannot be used as the data input/output pin simultaneously.

∗2: After the pin functions are switched for USIL, the pins are configured for input until the interface mode is set using the USIL_GCFG register. If USIL software reset is performed when the pins are already configured for USIL, all USIL pins are configured for input. Before performing software reset while USIL is operating, switch the pin functions to general-purpose input/output ports as necessary.

Note: Use a GPIO port to output the slave select signal when USIL is configured to SPI master mode.

The USIL input/output pins (USIL_DI, USIL_DO, USIL_CK, USIL_CS, USIL_LCD_D[7:0]) are shared with I/O ports and are initially set as general-purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as USIL input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

21.3 USIL Clock Sources

Operating clock

The USIL uses PCLK as the operating clock. Therefore, PCLK must be supplied from the CMU before starting the USIL including setting the control registers. For more information on the PCLK supply, refer to the "Clock Management Unit (CMU)."

Transfer clock

When the USIL is configured to a UART, SPI master (normal mode), I²C master, LCD SPI, or LCD parallel interface, the source clock for transfer is supplied by the fine mode 8-bit timer (T8F Ch.3). Program T8F Ch.3 according to the transfer rate and enable supplying the source clock to the USIL module. The USIL module divides the source clock to generate the transfer clock (or sampling clock). Be aware that the division ratio in the USIL depends on the interface mode.
When the USIL is configured to an SPI master (fast mode), PCLK is used as the source clock.
When the USIL is configured to an SPI slave or I²C slave device, the transfer clock is supplied from the external master device. However, SPI slave mode uses PCLK and I²C slave mode uses the T8F Ch.3 output clock to generate the sampling signal.

Table 21.3.1 USIL Clocks

Clock	Interface mode	Clock source
Operating clock	UART	PCLK
	SPI master	PCLK
	SPI slave	PCLK
	I²C master	PCLK
	I²C slave	PCLK
	LCD SPI	PCLK
	LCD parallel	PCLK
Transfer/sampling clock source (division ratio in USIL)	UART	T8F Ch.3 (f_{SOURCE}/8)
	SPI master	Normal mode: T8F Ch.3 (f_{SOURCE}/2) Fast mode: PCLK (f_{PCLK})
	SPI slave	PCLK (f_{PCLK}/4) for sampling
	I²C master	T8F Ch.3 (f_{SOURCE}/8)
	I²C slave	T8F Ch.3 (f_{SOURCE}) for sampling
	LCD SPI	T8F Ch.3 (f_{SOURCE}/2)
	LCD parallel	T8F Ch.3 (f_{SOURCE})

$f_{SOURCE} = f_{SYS\_CLK} \times DF / (TR + 1)$
bps = f_{SOURCE} / 8 (UART mode, I²C master mode)
bps = f_{SOURCE} / 2 (SPI master mode, LCD SPI mode)

f_{SOURCE}: T8F Ch.3 output clock frequency [Hz]
f_{SYS_CLK}: System clock frequency [Hz]
bps: Transfer rate [bps]
DF: Division ratio set by DF[3:0]/T8F_CLK3 register (T8F Ch.3)
TR: Reload data to be set to the T8F_TR3 register (T8F Ch.3)

Example: UART mode, transfer rate = 115,200 bps, system clock = 48 MHz, DF[3:0]/T8F_CLK3 register setting (T8F Ch.3) = 1/1.

$TR = (f_{SYS\_CLK} \times DF) / (bps \times 8) - 1 = 48{,}000{,}000 \times 1 / (115{,}200 \times 8) - 1 = 51$ (=0x33)

For more information on controlling the T8F module, refer to the "Fine Mode 8-bit Timers (T8F)" chapter.

Note: When the USIL is set to I²C slave mode, i2c_sck (I²C clock) is supplied from the external I²C master. The T8F output clock frequency (f_{SOURCE}) should be determined according to the i2c_sck frequency.

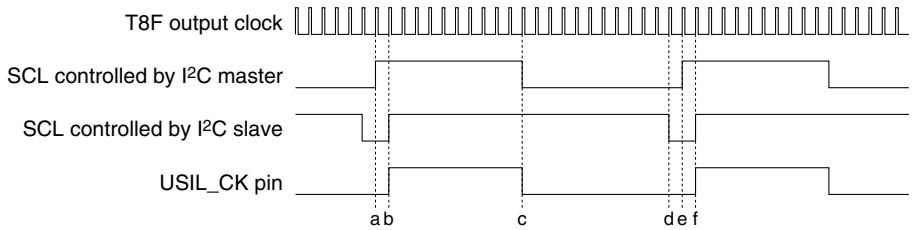


Figure 21.3.1 I²C Clock in I²C Slave Mode

Tbf = Ti2c_baud_rate
Tbc = Ti2c_baud_rate_high
Tcf = Ti2c_baud_rate_low

Tce: The I²C master occupies the SCL line by driving it to low.
Tac: The I²C master releases the SCL line.
Tdf: In order to finish the internal operations, the I²C slave occupies the SCL line for two source clock (T8F output clock) cycles by driving it to low after detecting that the I²C master drives the SCL line to low.

The T8F output clock frequency (f_{SOURCE}) must be set so that the conditions shown below are satisfied.

$f_{SOURCE} > 3/Tbc$
$f_{SOURCE} > 4/Tce$

Be aware that the actual SCL signal will be delayed, as the I²C slave forcibly drives the SCL line to low. The figure below shows an example in which the timing becomes worse.

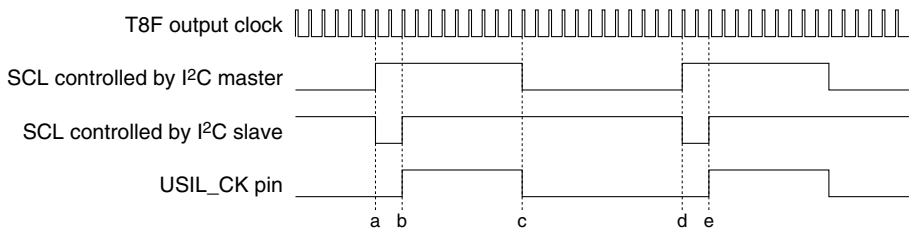


Figure 21.3.2 Example of Delayed I²C Clock

21.4 USIL Module Settings

Make the following settings before starting data transfers using the USIL module.
(1) Program the clock source module to supply the clock required to the USIL module. (See Section 21.3.)
(2) Reset the USIL module.
(3) Set the USIL interface mode and a general condition (MSB first/LSB first) to be applied to all interface modes.
(4) Configure the pins to be used for USIL according to the interface mode. (See Section 21.2.)
(5) Set the data format and operating conditions for the interface mode selected.
(6) Set interrupt and DMA transfer conditions if necessary. (See Section 21.7.)

21.4.1 USIL Module Software Reset

Writing 0x0 to USILMOD[2:0]/USIL_GCFG register resets the USIL module circuits. Be sure to perform software reset and then set the interface mode before changing other USIL configurations. The software reset operation resets all the USIL register contents to the initial values.

21.4.2 Interface Mode

The USIL module provides seven serial interface functions shown in Section 21.1. Each channel can be configured to one of them using the USILMOD[2:0]/USIL_GCFG register.

Table 21.4.2.1 Interface Mode Selection

USILMOD[2:0]	Interface mode
0x7	LCD parallel
0x6	LCD SPI
0x5	I²C slave
0x4	I²C master
0x3	SPI slave
0x2	SPI master
0x1	UART
0x0	Software reset

(Default: 0x0)

Note: Be sure to perform software reset and set the interface mode before changing other USIL configurations.

21.4.3 General Settings for All Interface Modes

MSB first/LSB first selection

Use LSBFST/USIL_GCFG register to select whether the data MSB or LSB is input/output first.
LSB first is selected when LSBFST is set to 0 (default). MSB first is selected when LSBFST is set to 1.

21.4.4 Settings for UART Mode

When the USIL is used in UART mode, configure the data length, stop bit, and parity bit. The start bit length is fixed at 1 bit.

Data length

Use UCHLN/USIL_UCFG register to select the data length. Setting UCHLN to 0 (default) configures the data length to 7 bits. Setting UCHLN to 1 configures it to 8 bits.

Stop bit

Use USTPB/USIL_UCFG register to select the stop bit length. Setting USTPB to 0 (default) configures the stop bit length to 1 bit. Setting USTPB to 1 configures it to 2 bits.

Parity bit

Use UPREN/USIL_UCFG register to select whether the parity function is enabled or not. Setting UPREN to 0 (default) disables the parity function. In this case, no parity bit will be added to transfer data and receive data will not be checked for parity. Setting UPREN to 1 enables the parity function. In this case, a parity bit will be added to transfer data and receive data will be checked for parity.
When the parity function is enabled, the parity mode should be selected using UPMD/USIL_UCFG register. Setting UPMD to 0 (default) adds a parity bit and checks for odd parity. Setting UPMD to 1 adds a parity bit and checks for even parity.

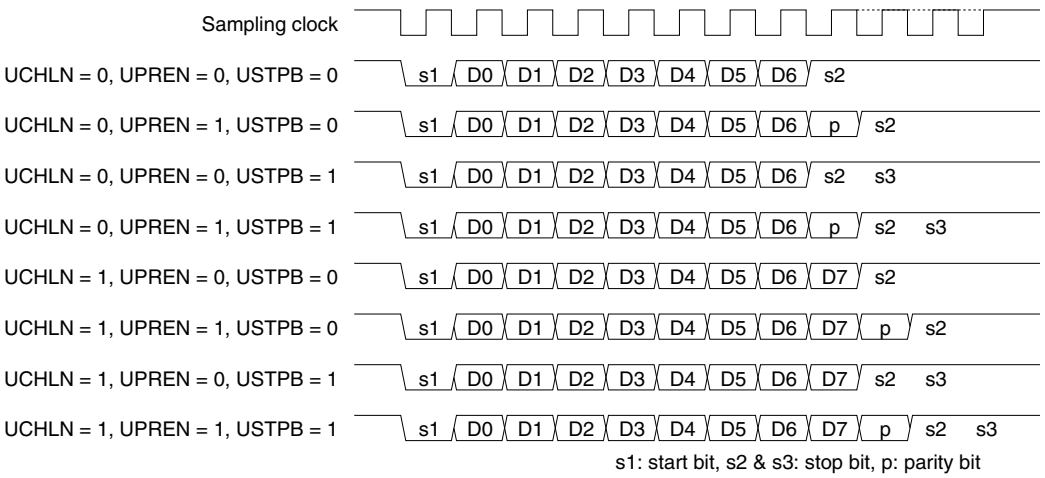


Figure 21.4.4.1 Transfer Data Format in UART Mode (LSB first)

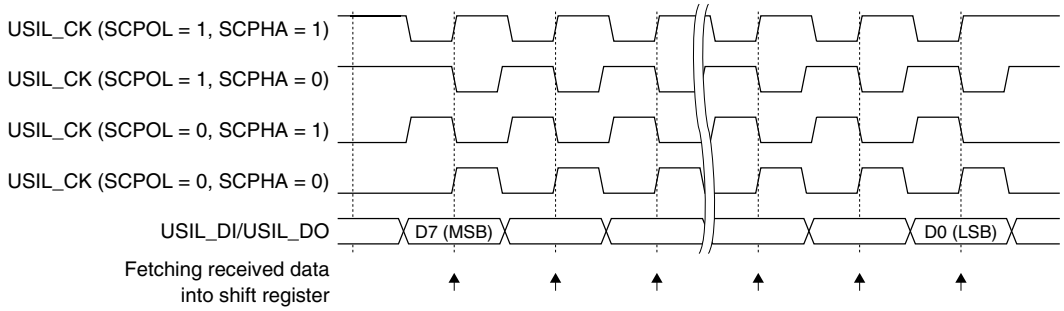
21.4.5 Settings for SPI Mode

When the USIL is used in SPI mode (master or slave), configure the SPI clock polarity/phase. When used in SPI master mode, select the clock mode.
Note that the data length in SPI mode is fixed at 8 bits.

SPI clock polarity and phase settings (master mode and slave mode)

Use SCPOL/USIL_SCFG register to select the SPI clock polarity. Setting SCPOL to 1 treats the SPI clock as active low. Setting it to 0 (default) treats it as active high.
The SPI clock phase can be selected using SCPHA/USIL_SCFG register.
These control bits set transfer timing as shown in Figure 21.4.5.1.

Master mode



Slave mode

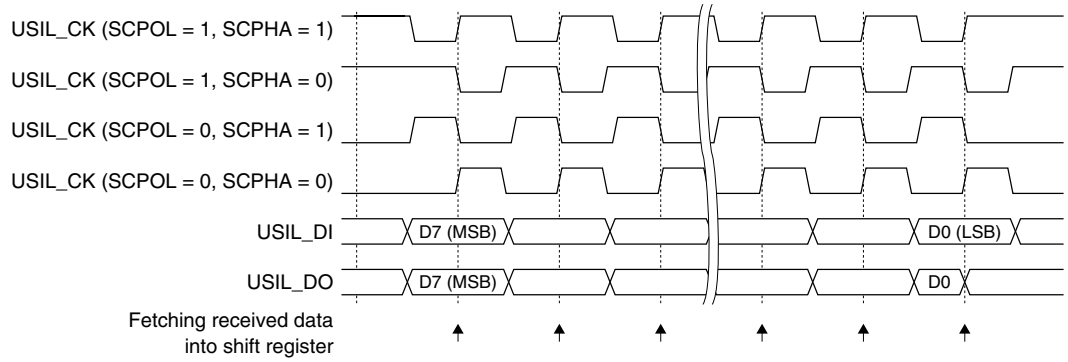


Figure 21.4.5.1 Clock and Data Transfer Timing (MSB first)

Clock mode (master mode only)

In SPI master mode, either normal or fast clock mode can be selected using SFSTMOD/USIL_SCFG register. Setting SFSTMOD to 0 (default) places the USIL into normal mode and the USIL generates the transfer clock by dividing the T8F output by 2. Setting SFSTMOD to 1 places the USIL into fast mode and the USIL uses PCLK supplied from the CMU directly as the transfer clock. The fast mode does not use the T8F.

The SPI slave mode uses the T8F output clock for generating the sampling clock.

21.4.6 Settings for I²C Mode

The I²C mode does not need to set data format and other conditions. The data length in I²C mode is fixed at 8 bits.

21.4.7 Settings for LCD SPI Mode

When the USIL is used in LCD SPI mode, configure the SPI clock polarity/phase and the data format.

SPI clock polarity and phase settings

Use LSCPOL/USIL_LSCFG register to select the SPI clock polarity. Setting LSCPOL to 1 treats the SPI clock as active low. Setting it to 0 (default) treats it as active high.
The SPI clock phase can be selected using LSCPHA/USIL_LSCFG register.
These control bits set transfer timing as shown in Figure 21.4.7.1.

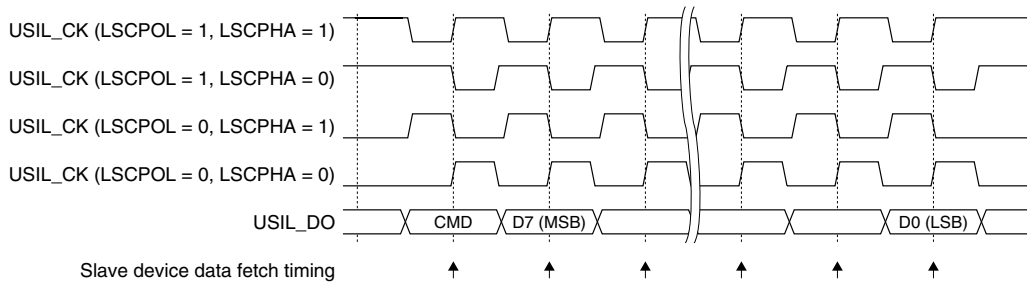


Figure 21.4.7.1 Clock and Data Transfer Timing (Command bit enabled, MSB first)

Data format

In SPI master mode, the display data format can be selected using LSDMOD[1:0]/USIL_LSDCFG register.

Table 21.4.7.1 LCD SPI Data Mode

LSDMOD[1:0]	Data mode
0x3	24-bit mode
0x2	18-bit mode
0x1	16-bit mode
0x0	8-bit mode

(Default: 0x0)

The 18-bit mode supports four data formats and one of them can be selected using LS18DFM[1:0]/USIL_LSDCFG register.

Table 21.4.7.2 LCD SPI 18-bit Data Format

LS18DFM[1:0]	Data format
0x3	Format 3
0x2	Format 2
0x1	Format 1
0x0	Format 0

(Default: 0x0)

To send data prefixed with a CMD (command) bit, set LSCMDEN/USIL_LSCFG register to 1. The command bit is used for controlling the SPI LCD driver/panel connected to the USIL. The command bit value to be sent can be specified using LSCMD/USIL_LSCFG register. Setting LSCMD to 1 configures the command bit to high. Setting LSCMD to 0 configures the command bit to low.
When using an LCD driver/panel that supports the A0 signal (command/data select signal), set LSCMDEN to 0 (default) to disable sending the CMD bit on the data line. In this case, the USIL_DI pin is used as the A0 signal output pin. The USIL_DI pin goes high when LSCMD is set to 1, and it goes low when LSCMD is set to 0.

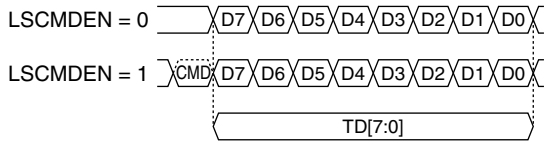


Figure 21.4.7.2 8-bit Data Format

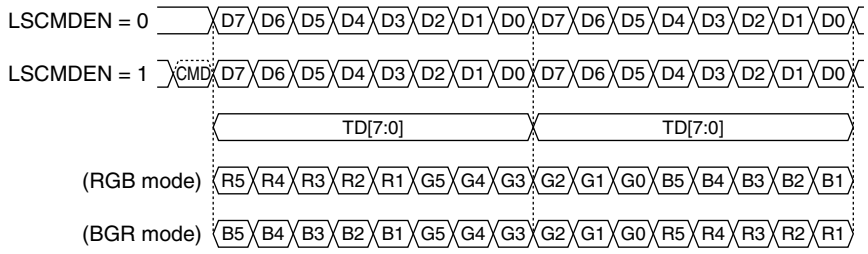
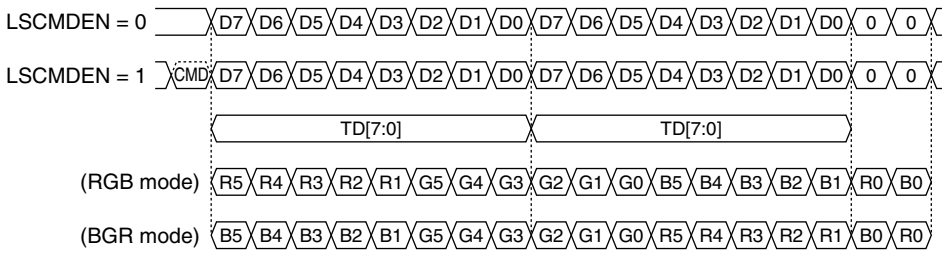
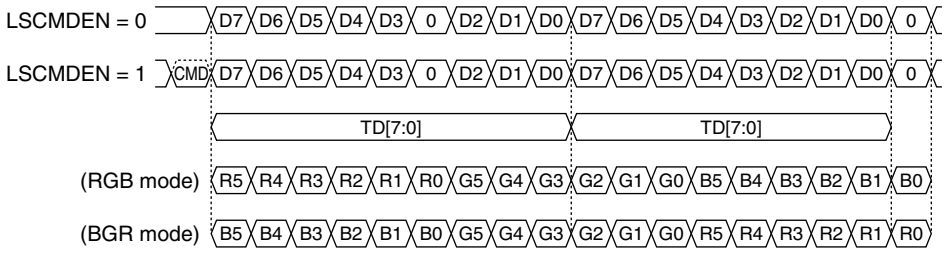


Figure 21.4.7.3 16-bit Data Format



18-bit Format 0



18-bit Format 1

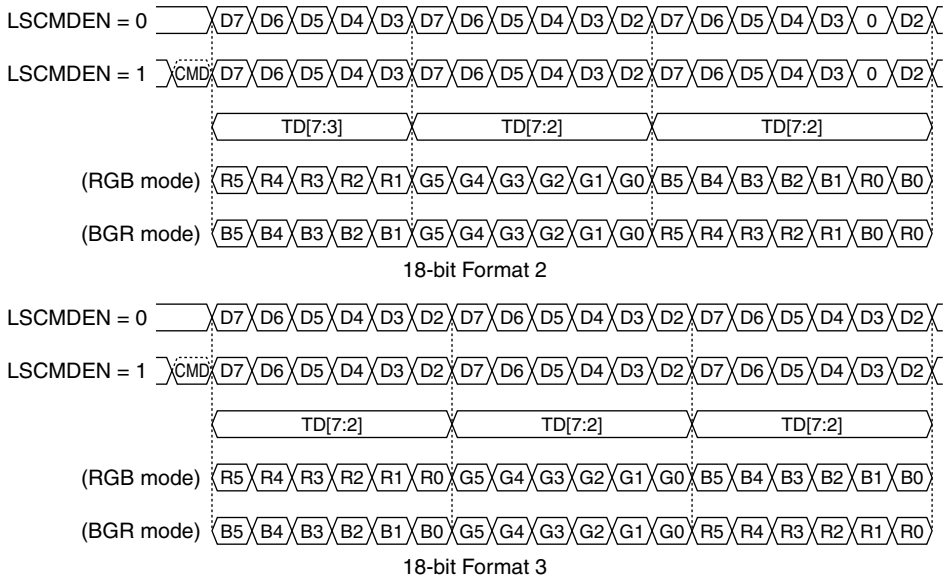


Figure 21.4.7.4 18-bit Data Format

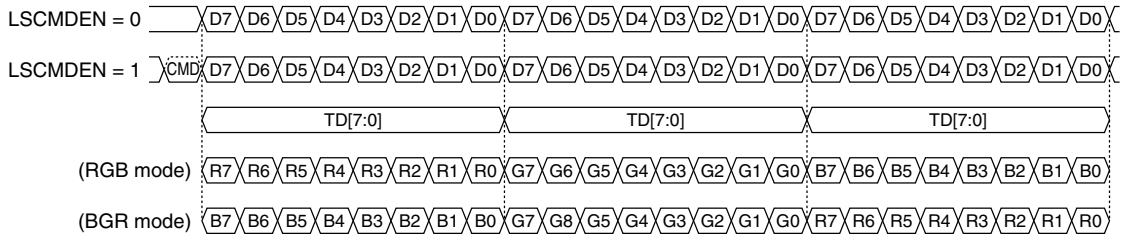


Figure 21.4.7.5 24-bit Data Format

21.4.8 Settings for LCD Parallel Mode

When the USIL is used in LCD parallel mode, configure the access timing parameters.

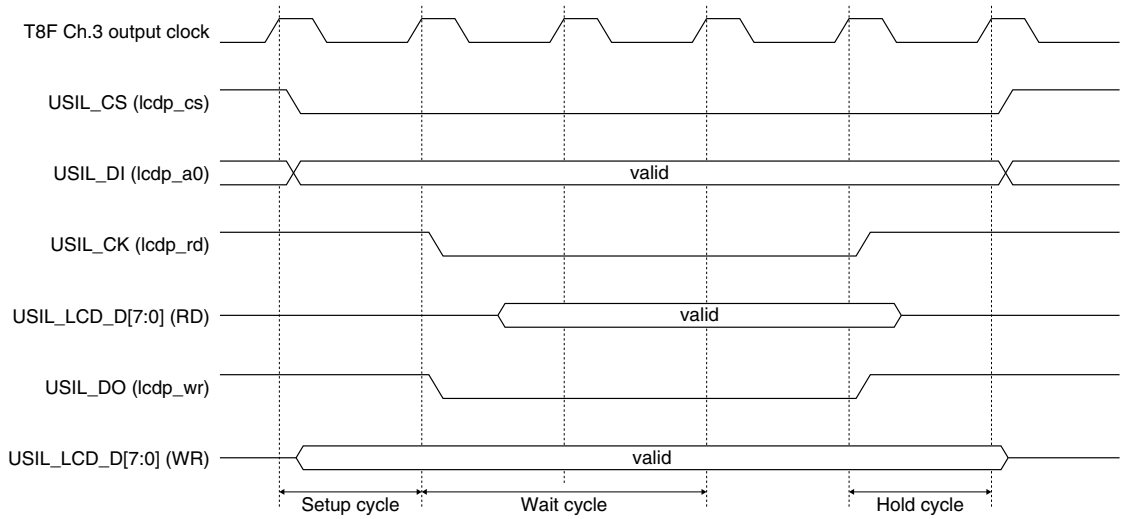


Figure 21.4.8.1 Access Timing Parameters

Setup cycle

The setup cycle can be set to 1–4 cycles using LPST[1:0]/USIL_LPAC register.

Table 21.4.8.1 Setup Cycle Settings

LPST[1:0]	Number of setup cycles
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

Hold cycle

The hold cycle can be set to 1–4 cycles using LPHD[1:0]/USIL_LPAC register.

Table 21.4.8.2 Hold Cycle Settings

LPHD[1:0]	Number of hold cycles
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

Wait cycle

The wait cycle can be set to 0–15 cycles using LPWT[3:0]/USIL_LPAC register.

Table 21.4.8.3 Wait Cycle Settings

LPWT[3:0]	Number of wait cycles
0xf	15 cycles
0xe	14 cycles
:	:
0x2	2 cycles
0x1	1 cycle
0x0	0 cycles

(Default: 0x0)

21.5 Data Transfer Control

This section describes how to control data transfers. The following explanations assume that the configurations described above and interrupt/DMA settings have already been finished.

21.5.1 Data Transfer in UART Mode

Data transmission

To start data transmission in UART mode, write the transmit data to the transmit data buffer (TD[7:0]/USIL_TD register).

The buffer data is sent to the transmit shift register, and the start bit is output from the USIL_DO pin. The data in the shift register is then output in sequence. Following output of the eighth data bit, the parity bit (if parity is enabled) and the stop bit are output.

The transmitter circuit includes two status flags: UTDIF/USIL_UIF register and UTBSY/USIL_UIF register.

The UTDIF flag indicates the transmit data buffer status. This flag is set to 1 indicating that the transmit data buffer becomes empty when data written to the transmit data buffer is sent to the transmit shift register. UTDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Write subsequent data to the transmit data buffer to start the following transmission using this interrupt or DMA. The transmit data buffer size is 1 byte, but a shift register is provided separately to allow data to be written while the previous data is being sent. If an interrupt or DMA is not used for transmission, be sure to confirm that the transmit data buffer is empty before writing transmit data. Writing data before UTDIF has been set will overwrite earlier transmit data inside the transmit data buffer. After UTDIF is set to 1, it can be reset to 0 by writing 1.

The UTBSY flag indicates the USIL status in UART mode. This flag switches to 1 when transmit data is written to the transmit buffer and reverts to 0 after both the shift register and transmit buffer become empty.

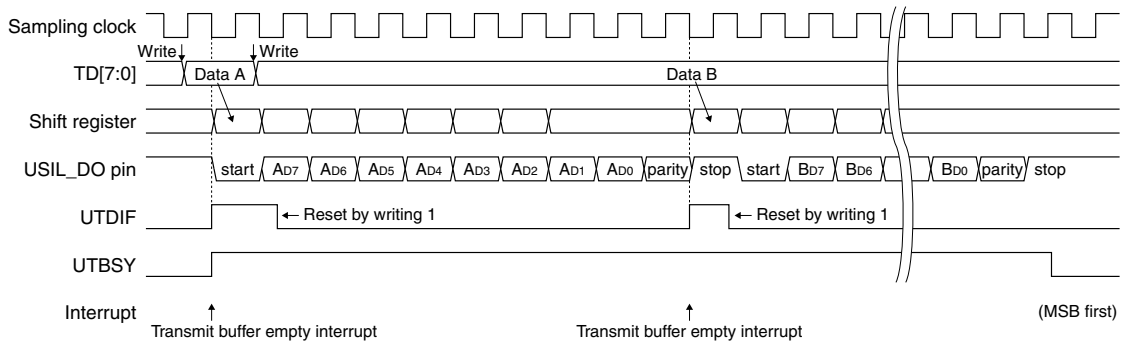


Figure 21.5.1.1 Data Transmission Timing Chart (UART mode)

Data reception

When the external serial device sends a start bit, the receiver circuit detects its low level and starts sampling the following data bits. Once the 8-bit data has been received into the shift register, the received data is loaded into the receive data buffer (RD[7:0]/USIL_RD register). If parity checking is enabled, the receiver circuit checks the received data at the same time by checking the parity bit received immediately after the eighth data bit.

The receiver circuit includes two status flags: URDIF/USIL_UIF register and URBSY/USIL_UIF register.

The URDIF flag indicates the receive data buffer status. This flag is set to 1 indicating that the received data can be read out when data received in the shift register is loaded to the receive data buffer. URDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Read the received data from the receive data buffer using this interrupt or DMA. The receive data buffer size is 1 byte, therefore the received data must be read before the subsequent data reception has completed. Furthermore, URDIF must be reset by writing 1. If the next reception is completed when URDIF is 1 and the receive data buffer (USIL_RD register) is not read, an overrun error occurs (at the time stop bit has been received).

The URBSY flag indicates the shift register status. This flag is set to 1 while data is being received in the shift register and reverts to 0 once the received data is loaded to the receive data buffer. Read this flag to check whether the receiver circuit is operating or at standby.

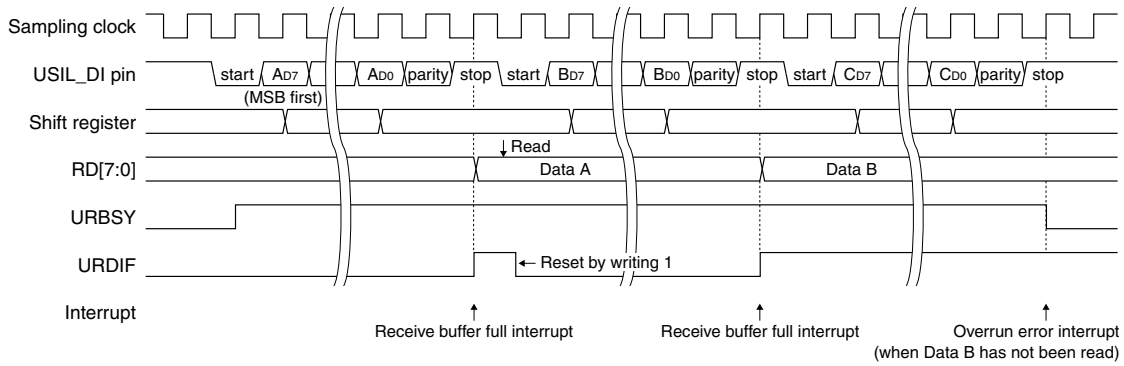


Figure 21.5.1.2 Data Receiving Timing Chart (UART mode)

21.5.2 Data Transfer in SPI Mode

Data transmission

To start data transmission in SPI mode, write the transmit data to the transmit data buffer (TD[7:0]/USIL_TD register).

The buffer data is sent to the transmit shift register. In SPI master mode, the module starts clock output from the USIL_CK pin. In SPI slave mode, the module awaits clock input from the USIL_CK pin. The data in the shift register is shifted in sequence at the clock rising or falling edge (see Figure 21.4.5.1) and sent from the USIL_DO pin.

The SPI controller includes two status flags for transfer control: STDIF/USIL_SIF register and SSIF/USIL_SIF register.
The STDIF flag indicates the transmit data buffer status. STDIF is set to 1 indicating that the transmit data buffer becomes empty when data written to the transmit data buffer is sent to the transmit shift register. STDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Write subsequent data to the transmit data buffer to start the following transmission using this interrupt or DMA. The transmit data buffer size is 1 byte, but a shift register is provided separately to allow data to be written while the previous data is being sent. If an interrupt or DMA is not used for transmission, be sure to confirm that the transmit data buffer is empty before writing transmit data. Writing data before STDIF has been set will overwrite earlier transmit data inside the transmit data buffer.
In SPI master mode, the SSIF flag indicates the USIL status. This flag switches to 1 when transmit data is written to the transmit buffer and reverts to 0 after both the shift register and transmit buffer become empty. Read this flag to check whether the SPI controller is operating or at standby.

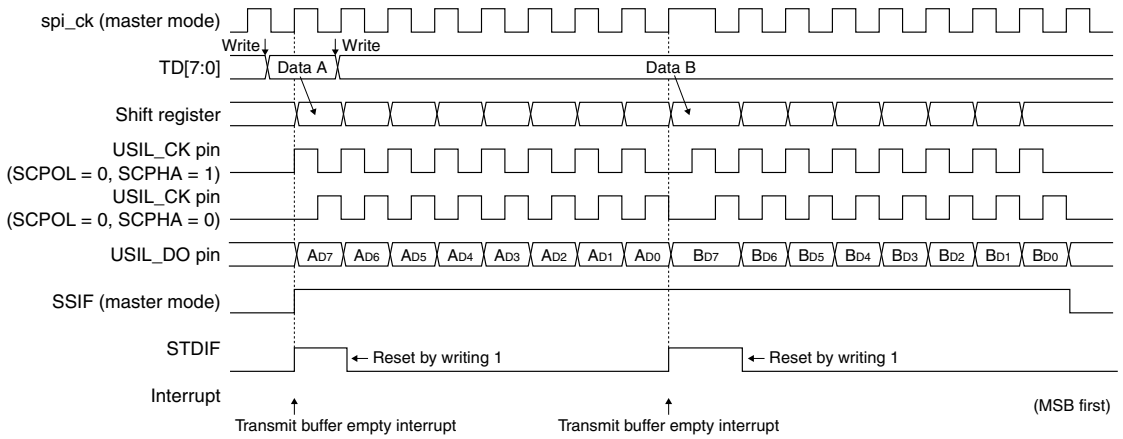


Figure 21.5.2.1 Data Transmission Timing Chart (SPI mode)

Data reception

In SPI master mode, write dummy data to the transmit data buffer. Writing to the transmit data buffer creates the trigger for reception as well as transmission start. Writing actual transmit data enables simultaneous transmission and reception. This starts the SPI clock output from the USIL_CK pin.
In SPI slave mode, the module waits until the clock is input from the USIL_CK pin. There is no need to write to the transmit data buffer if no transmission is required. The receiving operation is started by the clock input from the master device. If data is transmitted simultaneously, write transmit data to the transmit data buffer before the clock is input.
The data is received in sequence in the shift register at the SPI clock edge (see Figure 21.4.5.1). The received data is loaded into the receive data buffer once the 8 bits of data are received in the shift register.
The received data in the buffer can be read from RD[7:0]/USIL_RD register.
The SPI controller includes two status flags for transfer control: SRDIF/USIL_SIF register and SSIF/USIL_SIF register.
The SRDIF flag indicates the receive data buffer status. This flag is set to 1 when the data received in the shift register is loaded into the receive data buffer, indicating that the received data can be read out. SRDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Read the received data from the receive data buffer using this interrupt or DMA. The receive data buffer size is 1 byte, therefore the received data must be read before the subsequent data reception has completed. Furthermore, SRDIF must be reset by writing 1. While SRDIF is set to 1, the next received data will not be transferred from the shift register to the receive data buffer (the first byte data exists in the receive data buffer and the second byte data exists in the shift register). An overrun error occurs if the third byte data is received in this condition, as the second byte data in the shift register is corrupted (an overrun error occurs at the time the first bit of the third byte is fetched).
In SPI master mode, the SSIF flag indicates the shift register status. This flag switches to 1 at the beginning of data reception and reverts to 0 once the data is received. Read this flag to check whether the SPI controller is operating or at standby.

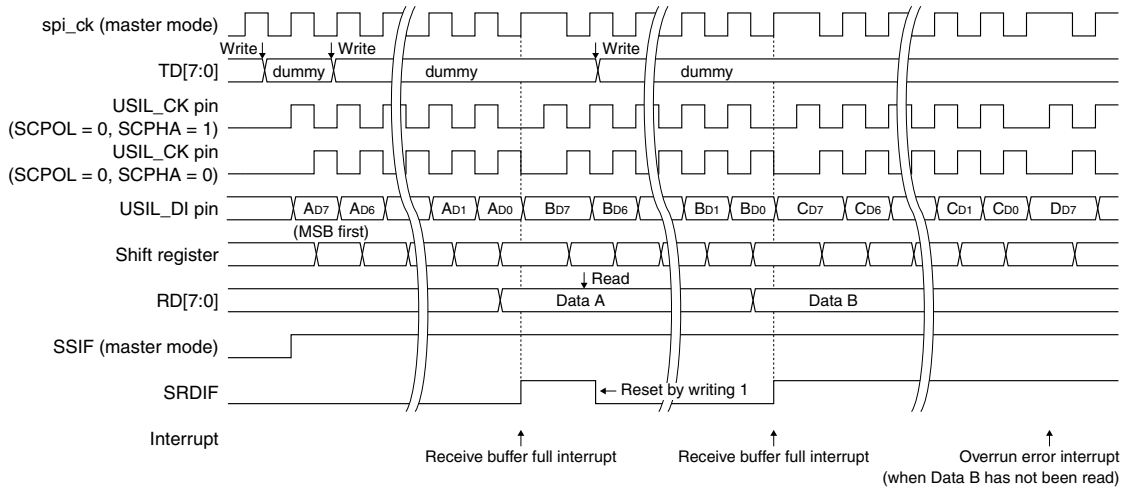


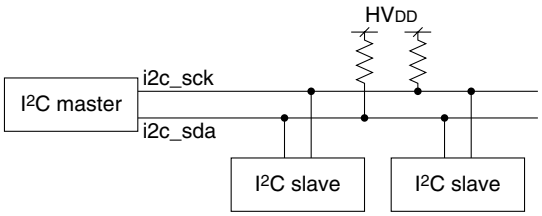
Figure 21.5.2.2 Data Receiving Timing Chart (SPI mode)

Slave select signal

In SPI slave mode, data transmission/receiving operations are enabled when the master device's slave select signal input to the USIL_CS pin is low. When the slave select signal is high, the SPI controller does not start data transfer even if the clock is input to the USIL_CK pin from the master device. The slave select signal status can be checked using SSIF/USIL_SIF register (it functions as the shift register status flag in SPI master mode). SSIF goes 1 when the slave select signal is inactive (high); it goes 0 when the slave select signal is active (low).

If a slave select output is required in SPI master mode, use a general-purpose I/O port and control its output with software.

21.5.3 Data Transfer in I²C Mode



The i2c_sck and i2c_sda pins go to a low level or high impedance status when the USIL module is set to I²C mode. These pins do not output a high level, so the i2c_sck and i2c_sda lines should be pulled up to HV_{DD} with an external pull-up resistor.

Note: Be sure to avoid pulling these pins up to a voltage that exceeds the HV_{DD} level.

Figure 21.5.3.1 I²C Mode Connection Example

Control method in I²C master mode

Data transfer in I²C master mode is controlled using IMTGMOD[2:0]/USIL_IMTG register and IMTG/USIL_IMTG register. Select an I²C master operation using IMTGMOD[2:0] and write 1 to IMTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

Table 21.5.3.1 Trigger List in I²C Master Mode

IMTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception
0x2	Data transmission
0x1	Stop condition
0x0	Start condition

(Default: 0x0)

Writing 1 to IMTG sets IMBSY/USIL_IMIF register to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, IMBSY is reset to 0. At the same time, the interrupt flag (IMIF/USIL_IMIF register) is also set to 1. After an interrupt occurs, read the status bits (IMSTA[2:0]/USIL_IMIF register) to check the operation finished. Then clear IMIF by writing 1. IMSTA[2:0] will be automatically cleared to 0x0.

Table 21.5.3.2 I²C Master Status Bits

IMSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been generated.
0x0	Start condition has been generated.

(Default: 0x0)

Data transmission in I²C master mode

The following describes the data transmission procedure in I²C master mode.

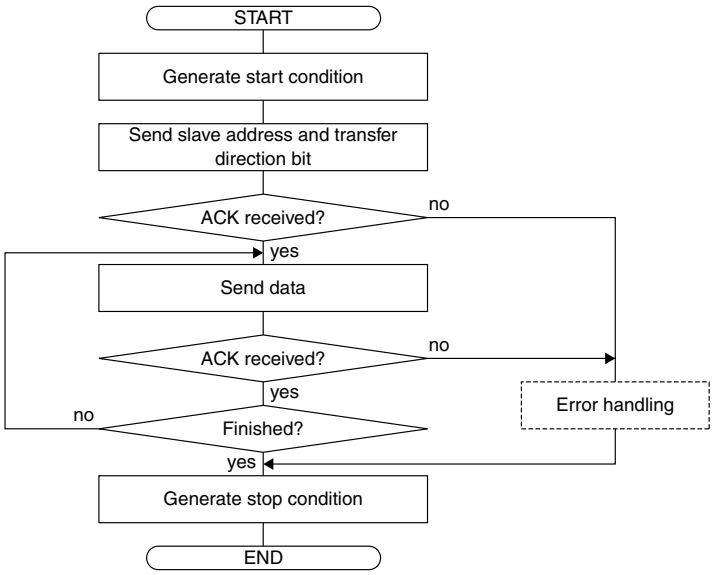
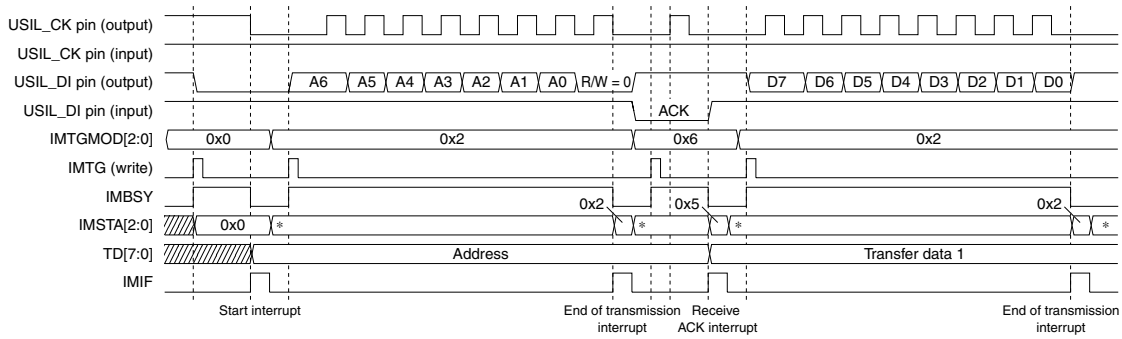
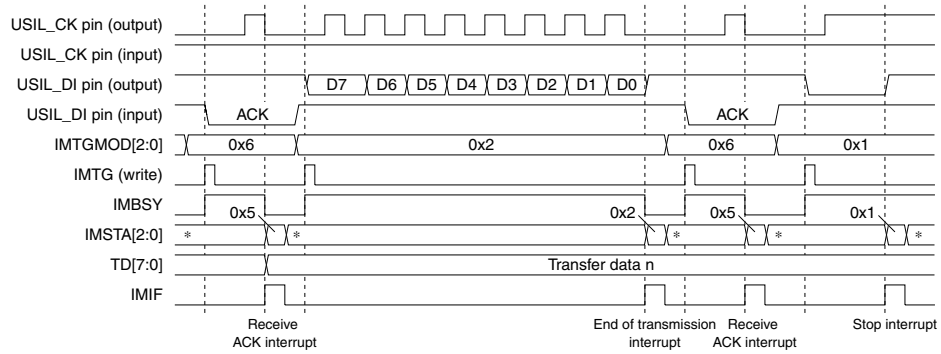


Figure 21.5.3.2 I²C Master Data Transmission Flow Chart



(1) Start condition → Data transmission



(2) Data transmission → Stop condition

∗ When IMIF is cleared via software, IMSTA[2:0] is also cleared to 0x0.

Figure 21.5.3.3 I²C Master Data Transmission Timing Chart

(1) Generating start condition

I²C data transfer starts when the I²C master device generates a start condition. The start condition applies when the SCL line is maintained at high and the SDA line is pulled down to low.

To generate a start condition in this I²C master, set IMTGMOD[2:0] to 0x0 (default) and write 1 to IMTG.

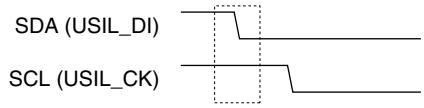


Figure 21.5.3.4 Start Condition

IMBSY is set to 1 while a start condition is being generated. When the start condition is generated, IMBSY is reset to 0 and IMSTA[2:0] is set to 0x0. The I²C bus is busy from this point on.

Note: Other operations cannot be started before a start condition is generated.

(2) Sending slave address and transfer direction bit

After a start condition has been generated, send the address of the slave device to be communicated and a transfer direction bit. I²C slave addresses are either 7-bit or 10-bit. This module uses an 8-bit transfer data buffer to send the slave address and transfer direction bit, enabling single transfers in 7-bit address mode. In 10-bit mode, data is sent twice or three times under software control. Figure 21.5.3.5 shows the configuration of the address data.

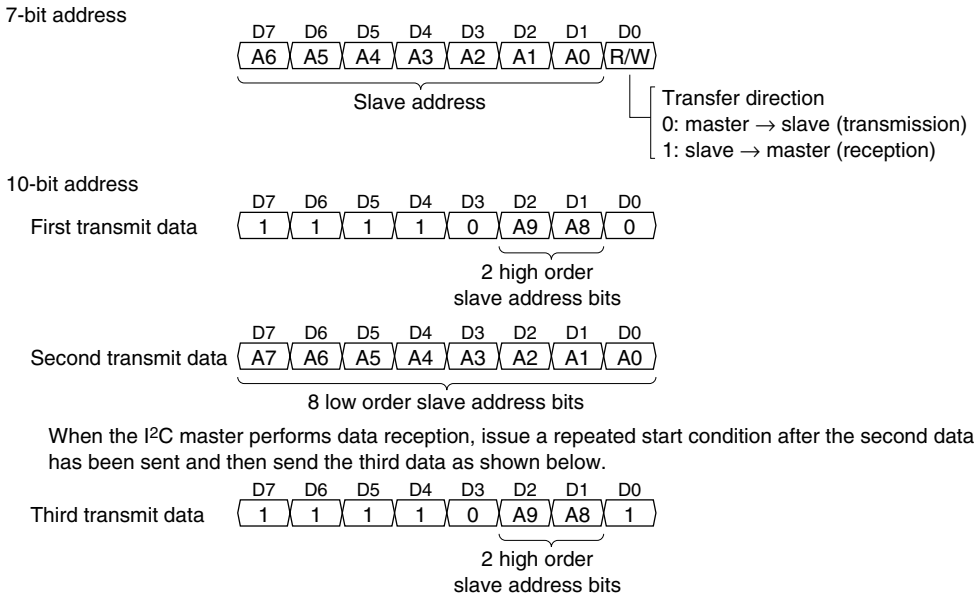


Figure 21.5.3.5 Transmit Data Specifying Slave Address and Transfer Direction

The transfer direction bit indicates the data transfer direction after the slave address has been sent. Set this bit to 0 when sending data from the master to the slave.
To send a slave address, set the address with the transfer direction bit to the transmit data buffer (TD[7:0]/USIL_TD register). Then set IMTGMOD[2:0] to 0x2 and write 1 to IMTG.
To send a 10-bit address, execute this procedure twice or three times as shown in Figure 21.5.3.5.
Writing 1 to IMTG sets IMBSY to 1. When data in the transmit data buffer is sent to the transmit shift register, IMBSY reverts to 0 and IMSTA[2:0] is set to 0x2. Confirm that the slave address (each byte) has been sent by reading IMBSY or using an interrupt.

After a slave address has been sent, the selected slave device sends back an ACK by pulling down the SCL line to low. If the SCL line maintains high, it is regarded as a NAK. In this case, the I²C controller cannot communicate with the slave device specified.

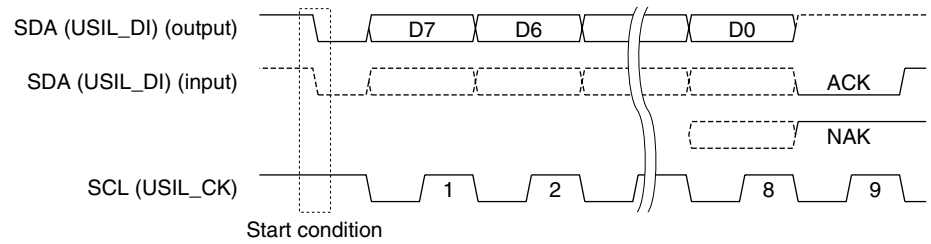


Figure 21.5.3.6 ACK and NAK

It is necessary to check that an ACK has been received before sending data. To do this, set IMTGMOD[2:0] to 0x6 and write 1 to IMTG after the slave address has been sent.
IMBSY is set to 1 while an ACK/NAK is being detected and it reverts to 0 when the detection has completed. Receiving an ACK sets IMSTA[2:0] to 0x5; receiving a NAK sets it to 0x6. Check IMSTA[2:0] after confirming IMBSY or using an interrupt. When an ACK has been received, perform data transmission. When a NAK has been received, perform an error handling.

(3) Data transmission

The data transmission procedure is the same as that of the slave address transmission.

1. Write an 8-bit transmit data to the transmit data buffer (TD[7:0]).
2. Set IMTGMOD[2:0] to 0x2 and IMTG to 1.

This trigger transfers the buffer data to the transmit shift register to start transmission. The module starts clock output from the USIL_CK pin. The data in the shift register is shifted in sequence with the clock and sent from the USIL_DO pin.

Writing 1 to IMTG sets IMBSY to 1. When data in the transmit data buffer is sent to the transmit shift register, IMBSY reverts to 0 and IMSTA[2:0] is set to 0x2 (end of transmit data). An interrupt request can be generated at this point. Write subsequent data to the transmit data buffer to start the following transmission using this interrupt.

However, as in the case of the slave address transmission, check that the slave device has sent back an ACK (by setting IMTGMOD[2:0] to 0x6 and IMTG to 1) before starting the following 8-bit data transmission.

Repeat an 8-bit data transmission and ACK receiving check for the required number of times.

(4) Generating stop condition

To end I²C communication after all data has been sent, the I²C master must generate a stop condition. The stop condition applies when the SCL line is maintained at high and the SDA line is pulled up from low to high. To generate a stop condition in this I²C master, set IMTGMOD[2:0] to 0x1 and write 1 to IMTG.

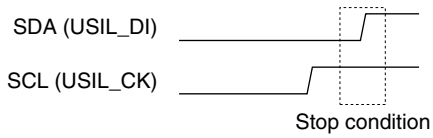


Figure 21.5.3.7 Stop Condition

IMBSY is set to 1 while a stop condition is being generated. When the stop condition is generated, IMBSY is reset to 0 and IMSTA[2:0] is set to 0x1. Read IMBSY or use an interrupt to check that a stop condition has been generated. The I²C bus subsequently switches to free state.

(5) Generating repeated start condition

To make it possible to continue with a different data transfer after a data transmission has completed, the I²C master can omit stop condition generation and generate a repeated start condition. To generate a repeated start condition, perform a start condition generation procedure described in Step (1). Slave address transmission is subsequently possible with the I²C bus remaining in the busy state.

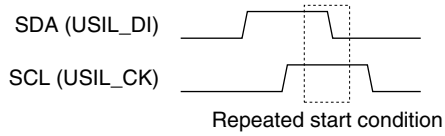


Figure 21.5.3.8 Repeated Start Condition

Data reception in I²C master mode

The following describes the data receiving procedure in I²C master mode.

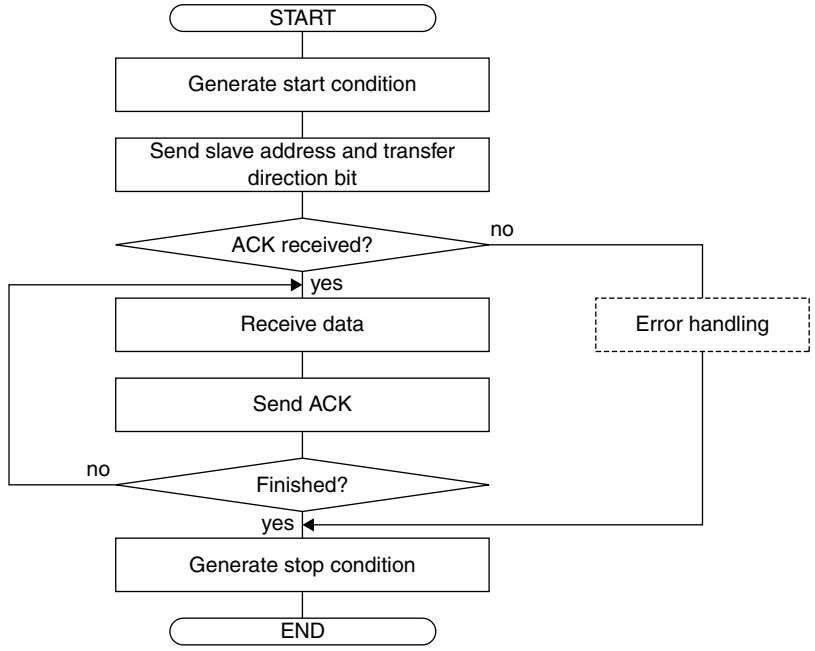
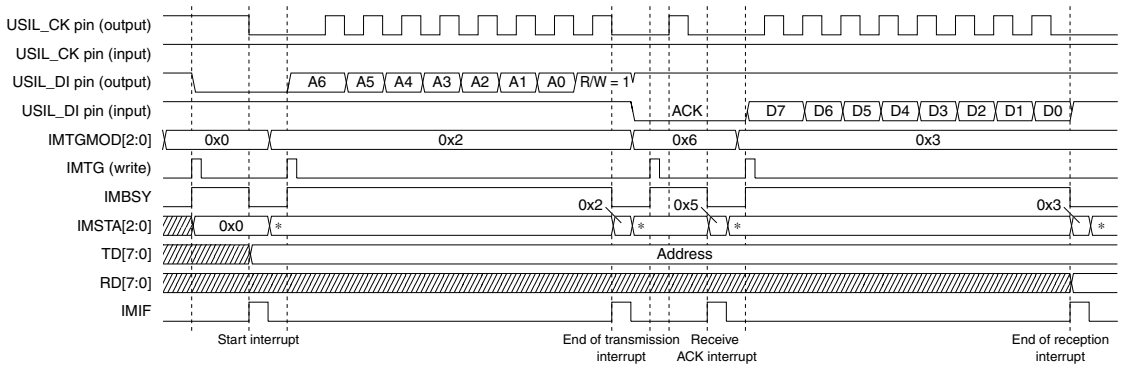
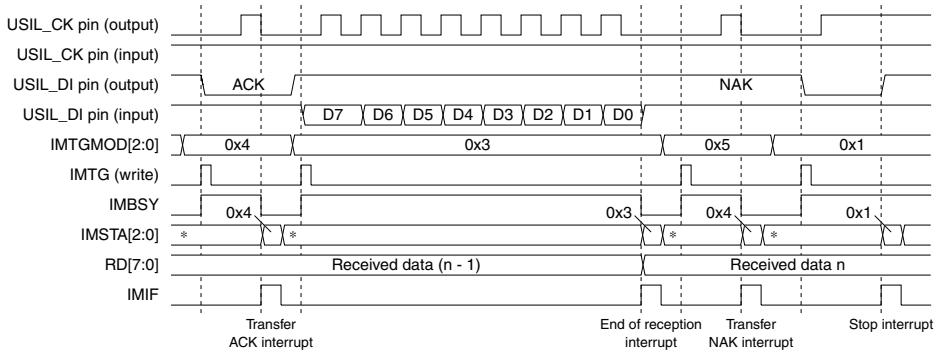


Figure 21.5.3.9 I²C Master Data Receiving Flow Chart



(1) Start condition → Data reception



(2) Data reception → Stop condition

∗ When IMIF is cleared via software, IMSTA[2:0] is also cleared to 0x0.

Figure 21.5.3.10 I²C Master Data Receiving Timing Chart

Note: The timing chart above shows a basic transfer operation that does not include an actual I²C transfer procedure. See "Receiving control byte in I²C slave mode" in "21.9 Precautions."

(1) Generating start condition

The procedure is the same as that of data transmission in I²C master mode.

(2) Sending slave address and transfer direction bit
The procedure is the same as that of data transmission in I²C master mode. However, send the slave address with the transfer direction bit set to 1. Then check that the slave device sends back an ACK.

(3) Data reception
To start data reception, set IMTGMOD[2:0] to 0x3 and write 1 to IMTG.
This trigger starts outputting 8 clocks from the USIL_CK pin. The USIL_DO pin status is sampled in sync with the clock and loaded to the shift register. The received data is loaded to the receive data buffer (RD[7:0]/USIL_RD register) once the 8-bit data has been received in the shift register.
Writing 1 to IMTG sets IMBSY to 1. When the received data is loaded to the receive data buffer, IMBSY reverts to 0 and IMSTA[2:0] is set to 0x3 (end of receive data). An interrupt request can be generated at this point. Read the received data from the receive data buffer using this interrupt.

It is necessary to send back an ACK or NAK to the slave device after an 8-bit data has been received. To send back an ACK, set IMTGMOD[2:0] to 0x4 and write 1 to IMTG. To send back a NAK, set IMTGMOD[2:0] to 0x5 and write 1 to IMTG.
IMBSY is set to 1 while an ACK/NAK is being sent and it reverts to 0 when the transmission has completed. An interrupt or DMA request can be generated at this point. When an ACK or NAK has been sent, IMSTA[2:0] is set to 0x4.
Repeat an 8-bit data reception and ACK (NAK) transmission for the required number of times.

(4) Generating stop condition
The procedure is the same as that of data transmission in I²C master mode.

(5) Generating repeated start condition
The procedure is the same as that of data transmission in I²C master mode.

Clock stretch function

During transmitting/receiving data, the slave device may issue a wait request to the master device by pulling down the SCL line to low until the slave device becomes ready to transmit/receive the subsequent data. The master device enters a standby state until the wait request is canceled (the SCL line goes high).
This I²C controller supports this clock stretch function. When a clock stretch condition is detected after a slave address or data has been sent/received, this module enters a waiting status and it does not start operating even if it accepts a trigger for data transfer until the clock stretch status is canceled. IMBSY is maintained at 1 until the triggered operation has completed including a waiting status.

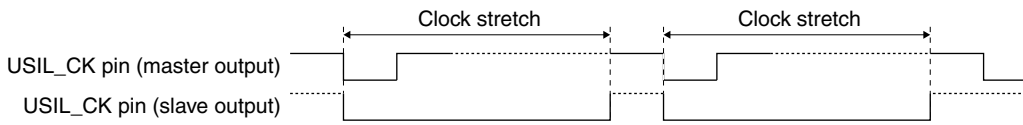


Figure 21.5.3.11 Clock Stretch

Control method in I²C slave mode

Data transfer in I²C slave mode is controlled using ISTGMOD[2:0]/USIL_ISTG register and ISTG/USIL_ISTG register. Select an I²C slave operation using ISTGMOD[2:0] and write 1 to ISTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

Table 21.5.3.3 Trigger List in I²C Slave Mode

ISTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception/stop condition detection
0x2	Data transmission
0x1	Reserved
0x0	Wait for start condition

(Default: 0x0)

Writing 1 to ISTG sets ISBSY/USIL_ISIF register to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, ISBSY is reset to 0. At the same time, the interrupt flag (ISIF/USIL_ISIF register) is also set to 1. After an interrupt occurs, read the status bits (ISSTA[2:0]/USIL_ISIF register) to check the operation finished. Then clear ISIF by writing 1. ISSTA[2:0] will be automatically cleared to 0x0.

Table 21.5.3.4 I²C Slave Status Bits

ISSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been detected.
0x0	Start condition has been detected.

(Default: 0x0)

Data transmission in I²C slave mode

The following describes the data transmission procedure in I²C slave mode.

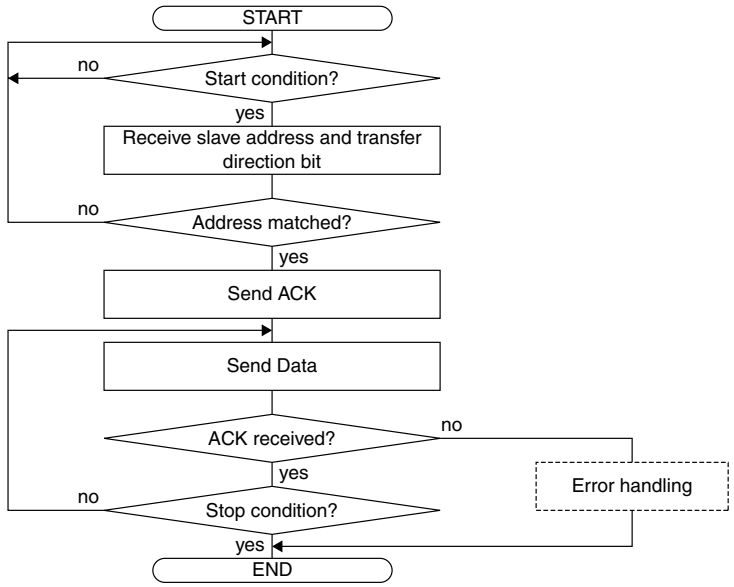
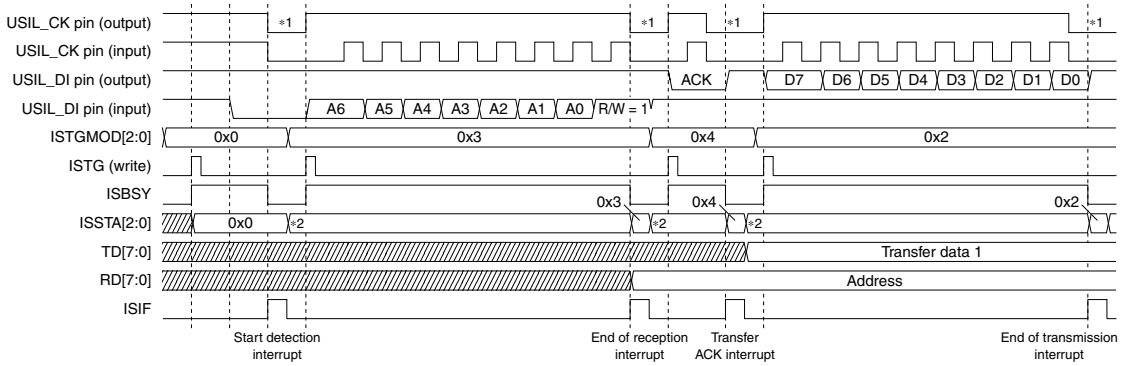
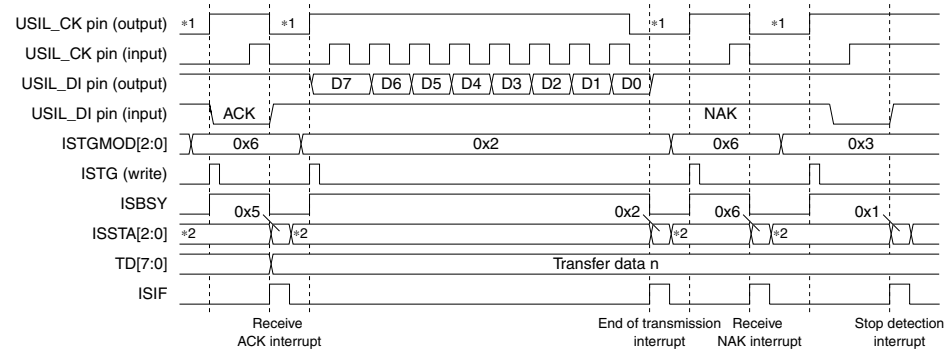


Figure 21.5.3.12 I²C Slave Data Transmission Flow Chart



(1) Start condition → Data transmission



(2) Data transmission → Stop condition

∗1 When the USIL_CK input is detected as low after the operation selected by ISTGMOD[2:0] has finished, the USIL I²C slave controller pulls down the USIL_CK pin to low to places the external I²C master into wait state. This pull-down is canceled to release the I²C master from wait state when the subsequent operation is triggered by ISTG.

∗2 When ISIF is cleared via software, ISSTA[2:0] is also cleared to 0x0.

Figure 21.5.3.13 I²C Slave Data Transmission Timing Chart

Note: The timing chart above shows a basic transfer operation that does not include an actual I²C transfer procedure. See "Receiving control byte in I²C slave mode" in "21.9 Precautions."

(1) Waiting for start condition

I²C data transfer starts when the I²C master device generates a start condition (see Figure 21.5.3.4).

First enable this I²C slave to detect a start condition by setting ISTGMOD[2:0] to 0x0 (default) and writing 1 to ISTG. The I²C controller starts detecting a start condition and sets ISBSY to 1. ISBSY is set to 1 while a start condition is being detected. ISBSY reverts to 0 and ISSTA[2:0] is set to 0x0 when the detection has completed. Check if a start condition is generated by reading ISBSY or using an interrupt.

Note: Other operations cannot be started before a start condition is detected.

(2) Receiving slave address and transfer direction data bit

The I²C master sends the address of the slave device to be communicated and a transfer direction bit (see Figure 21.5.3.5) after it has generated a start condition. Set this I²C slave into receiving status to receive the slave address. To start reception, set ISTGMOD[2:0] to 0x3 and write 1 to ISTG.

This trigger starts sampling clocks input from the USIL_CK pin. When clocks are input, the I²C controller loads the USIL_DO pin status to the shift register in sync with each clock. The received data is loaded to the receive data buffer (RD[7:0]/USIL_RD register) once the 8-bit data has been received in the shift register.

Writing 1 to ISTG sets ISBSY to 1. When the received data is loaded to the receive data buffer, ISBSY reverts to 0 and ISSTA[2:0] is set to 0x3 (end of receive data). An interrupt request can be generated at this point. Read the received data from the receive data buffer using this interrupt.

When a 7-bit address is used, the slave address and transfer direction bit can be obtained in one operation. When a 10-bit address is used, save the first data received in the receive data buffer into the memory and perform data reception again to obtain the remaining address bits.

Check whether the received address is matched to this I²C slave address or not. When they are matched, send back an ACK to the I²C master by setting ISTGMOD[2:0] to 0x4 and write 1 to ISTG. ISBSY is set to 1 while an ACK is being sent and it reverts to 0 when the transmission has completed. An interrupt request can be generated at this point. When an ACK has been sent, ISSTA[2:0] is set to 0x4.

If the received address is not for this I²C slave, abort data reception and return to Step (1) to wait the subsequent start condition.

(3) Data transmission

When the transfer direction bit received with the slave address in Step (2) is 1, start data transmission by the following procedure:

1. Write an 8-bit transmit data to the transmit data buffer (TD[7:0]).
2. Set ISTGMOD[2:0] to 0x2 and ISTG to 1.

This trigger transfers the buffer data to the transmit shift register to start transmission. When clocks are input from the USIL_CK pin, the data in the shift register is shifted in sequence with the clock and sent from the USIL_DO pin.

Writing 1 to ISTG sets ISBSY to 1. When data in the transmit data buffer is sent to the transmit shift register, ISBSY reverts to 0 and ISSTA[2:0] is set to 0x2 (end of transmit data). An interrupt request can be generated at this point. Write subsequent data to the transmit data buffer to start the following transmission using this interrupt.

However, check that the master device has sent back an ACK or NAK (by setting ISTGMOD[2:0] to 0x6 and ISTG to 1) before starting the following 8-bit data transmission.

ISBSY is set to 1 while an ACK/NAK is being detected and it reverts to 0 when the detection has completed. Receiving an ACK sets ISSTA[2:0] to 0x5; receiving a NAK sets it to 0x6. Check ISSTA[2:0] after confirming ISBSY or using an interrupt. When an ACK has been received, perform data transmission. When a NAK has been received, perform the appropriate handling.

(4) When a stop condition is received

If the ISSTA[2:0] value read during data transmission is 0x1, the I²C master device has generated a stop condition (see Figure 21.5.3.7). In this case, abort data transmission.

Data reception in I²C slave mode

The following describes the data receiving procedure in I²C slave mode.

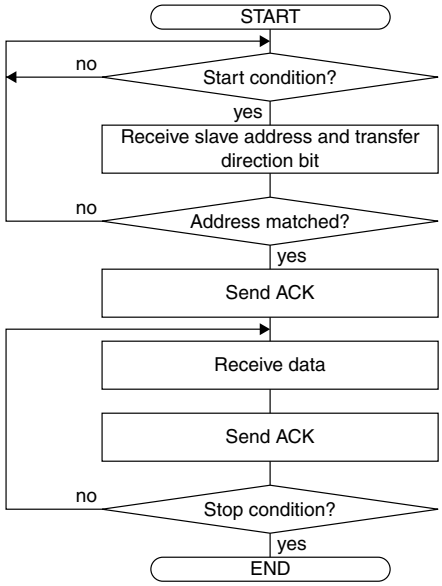
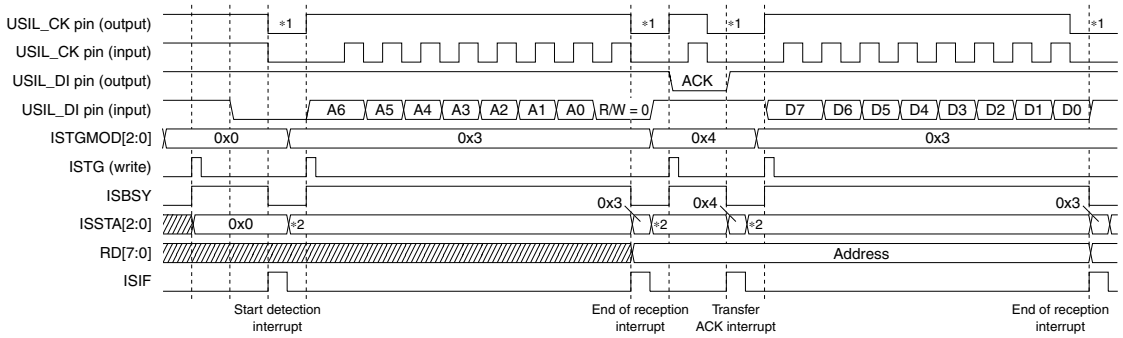
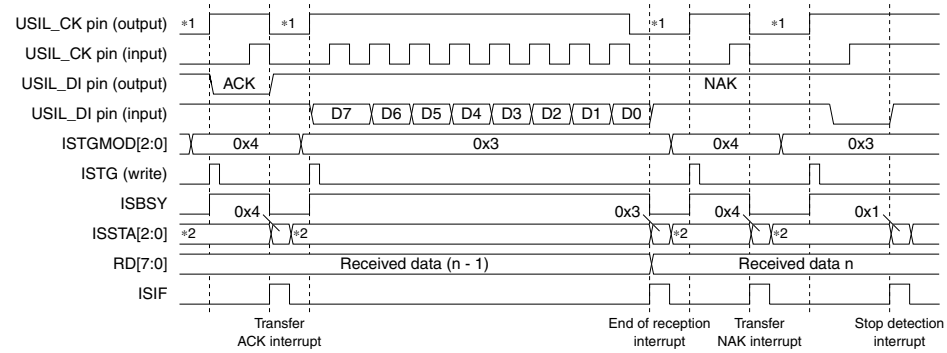


Figure 21.5.3.14 I²C Slave Data Receiving Flow Chart



(1) Start condition → Data reception



(2) Data reception → Stop condition

∗1 When the USIL_CK input is detected as low after the operation selected by ISTGMOD[2:0] has finished, the USIL I²C slave controller pulls down the USIL_CK pin to low to places the external I²C master into wait state. This pull-down is canceled to release the I²C master from wait state when the subsequent operation is triggered by ISTG.

∗2 When ISIF is cleared via software, ISSTA[2:0] is also cleared to 0x0.

Figure 21.5.3.15 I²C Slave Data Receiving Timing Chart

Note: The timing chart above shows a basic transfer operation that does not include an actual I²C transfer procedure. See "Receiving control byte in I²C slave mode" in "21.9 Precautions."

(1) Waiting for start condition
The procedure is the same as that of data transmission in I²C slave mode.

(2) Receiving slave address and transfer direction data bit
The procedure is the same as that of data transmission in I²C slave mode.

(3) Data reception
When the transfer direction bit received with the slave address in Step (2) is 0, start data reception by setting ISTGMOD[2:0] to 0x3 and writing 1 to ISTG.
When clocks are input, the I²C controller loads the USIL_DO pin status to the shift register in sync with each clock. The received data is loaded to the receive data buffer (RD[7:0]/USIL_RD register) once the 8-bit data has been received in the shift register.
Writing 1 to ISTG sets ISBSY to 1. When the received data is loaded to the receive data buffer, ISBSY reverts to 0 and ISSTA[2:0] is set to 0x3 (end of receive data). An interrupt request can be generated at this point. Read the received data from the receive data buffer using this interrupt.

It is necessary to send back an ACK or NAK to the master device after an 8-bit data has been received. To send back an ACK, set ISTGMOD[2:0] to 0x4 and write 1 to ISTG. To send back a NAK, set ISTGMOD[2:0] to 0x5 and write 1 to ISTG.
ISBSY is set to 1 while an ACK/NAK is being sent and it reverts to 0 when the transmission has completed. An interrupt request can be generated at this point. When an ACK or NAK has been sent, ISSTA[2:0] is set to 0x4.
Repeat an 8-bit data reception and ACK (NAK) transmission for the required number of times.

(4) When a stop condition is received
If the ISSTA[2:0] value read during data reception is 0x1, the I²C master device has generated a stop condition (see Figure 21.5.3.7). In this case, abort data reception.

Clock stretch function

While data is being sent/received, this I²C slave generates a clock stretch status by pulling down the SCL line to low to make a wait request to the master device after an ACK is sent/received until the following data transfer is started.

21.5.4 Data Transmission in LCD SPI Mode

The LCD SPI mode supports only data transmission.
To start data transmission in LCD SPI mode, write the transmit data to the transmit data buffer (TD[7:0]/USIL_TD register) after setting the command bit status (LSCMD/USIL_LSCFG register).
The buffer data is sent to the transmit shift register. The module starts clock output from the USIL_CK pin. The data in the shift register is shifted in sequence at the clock rising or falling edge (see Figure 21.4.7.1) and sent from the USIL_DO pin.
The LCD SPI module includes two status flags for transfer control: LSTDIF/USIL_LSIF register and LSBSY/USIL_LSIF register.
The LSTDIF flag indicates the transmit data buffer status. LSTDIF is set to 1 indicating that the transmit data buffer becomes empty when data written to the transmit data buffer is sent to the transmit shift register. LSTDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Write subsequent data to the transmit data buffer to start the following transmission using this interrupt or DMA. The transmit data buffer size is 1 byte, but a shift register is provided separately to allow data to be written while the previous data is being sent. If an interrupt or DMA is not used for transmission, be sure to confirm that the transmit data buffer is empty before writing transmit data. Writing data before LSTDIF has been set will overwrite earlier transmit data inside the transmit data buffer.
The LSBSY flag indicates the USIL status in LCD SPI mode. This flag switches to 1 when transmit data is written to the transmit buffer and reverts to 0 after data transfer for the data size set using LSDMOD[1:0]/USIL_LSDCFG register has completed.

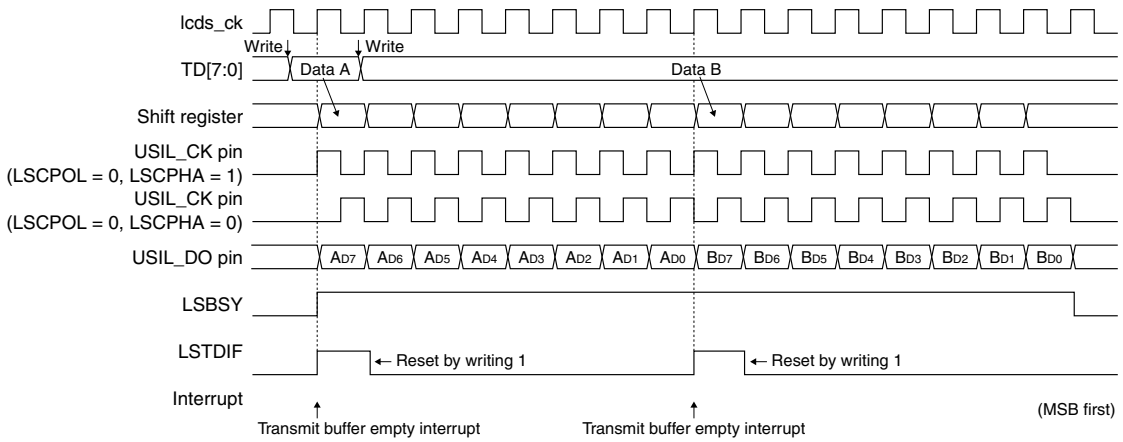


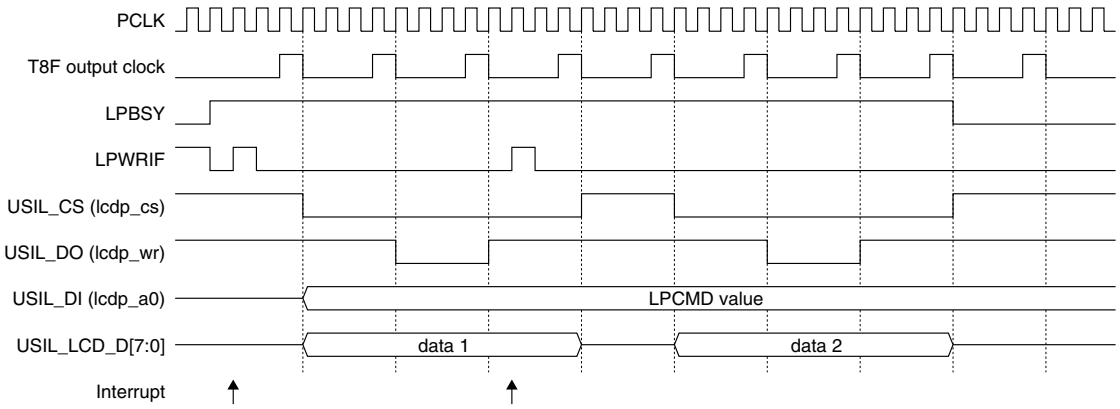
Figure 21.5.4.1 Data Transmission Timing Chart (LCD SPI mode, 16-bit data format)

21.5.5 Data Transfer in LCD Parallel Mode

Data write

To write data to the LCD driver/panel via the LCD parallel interface, write the data to the write (transmit data) buffer (TD[7:0]/USIL_TD register) after setting the command bit status (LPCMD/USIL_LPCFG register).
The command bit must be set before writing data to the write buffer. The command bit value set is output from the USIL_DI pin immediately after it is written to the register.
The LCD parallel interface asserts the chip enable signal and outputs the buffer data via the USIL_LCD_D[7:0] pins. The transmitter circuit includes two status flags: LPWRIF/USIL_LPIF register and LPBSY/USIL_LPIF register.
The LPWRIF flag indicates the write buffer status. This flag is set to 1 indicating that the write buffer becomes empty when data written to the buffer is transferred to the shift register. LPWRIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Write subsequent data to the write buffer to start the following transmission using this interrupt or DMA. If an interrupt or DMA is not used for transmission, be sure to confirm that the transmit data buffer is empty before writing transmit data. Writing data before LPWRIF has been set will overwrite earlier write data inside the write buffer. After LPWRIF is set to 1, it can be reset to 0 by writing 1.

The LPBSY flag indicates the parallel interface status. This flag switches to 1 when data is written the write buffer and reverts to 0 after the write cycle is completed. Read this flag to check whether the parallel interface circuit is operating or at standby.



(LPWT[3:0] = 0x0, LPST[1:0] = 0x0, LPHD[1:0] = 0x0)

Figure 21.5.5.1 Data-Write Timing Chart (LCD parallel mode)

Data read

To read data from the LCD driver/panel via the LCD parallel interface, issue a read trigger by writing 1 to LPRD/USIL_LPCFG register.

Set the command bit (LPCMD/USIL_LPCFG register) value before writing to LPRD. The command bit value set is output from the USIL_DI pin immediately after it is written to the register. Then it loads the USIL_LCD_D[7:0] pin status to the read (receive data) buffer (RD[7:0]/USIL_RD register). LPRD retains 1 until the read data is loaded to the read buffer.

When USIL_LCD_D[7:0] is loaded to the read buffer, the read buffer full flag (LPRDIF/USIL_LPIF register) is set to 1 indicating that the buffered data can be read out. LPRDIF is an interrupt flag. An interrupt or DMA request can be generated when this flag is set to 1 (see Section 21.7). Read data from the read buffer using this interrupt or DMA. The read buffer size is 1 byte, therefore the buffered data must be read before the subsequent data reading has completed. LPRDIF is reset by writing 1.

The LPBSY flag indicates the parallel interface status. This flag switches to 1 when LPRD is set to 1 and reverts to 0 after the read cycle is completed. Read this flag to check whether the parallel interface circuit is operating or at standby.

Note: Once a triggered read cycle is completed, data is stored in the read buffer and LPRDIF switches to high. After that read data from the read buffer before issuing the next read trigger. Otherwise, a subsequent read operation will not be started.

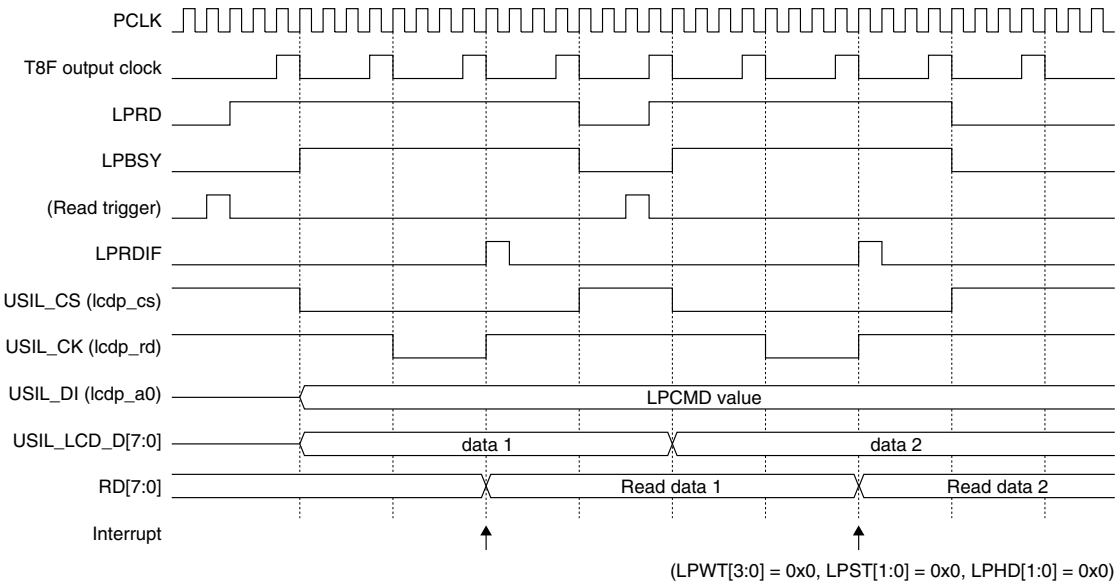


Figure 21.5.5.2 Single Read Timing Chart (LCD parallel mode)

Successive data read

The LCD parallel interface supports a successive data read function. By issuing a read trigger (writing 1 to LPRD) after setting the successive read enable bit (LPSRDEN/USIL_LPCFG register) to 1, the LCD parallel interface repeats data reading from the LCD driver/panel while LPSRDEN is 1 (reading the receive data buffer issues the next read trigger). When LPSRDEN is set to 0, the LCD parallel interface stops data reading after the read cycle being currently executed has finished. An interrupt or DMA can be generated in each read cycle (when data is loaded to the read buffer) similar to the single read operation, use it to read out the read buffer.

Note: After successive data reading has been terminated, there are 2 or 3 data remained in the read buffer. Be sure to read them as shown in the flow chart (Figure 21.5.5.4).

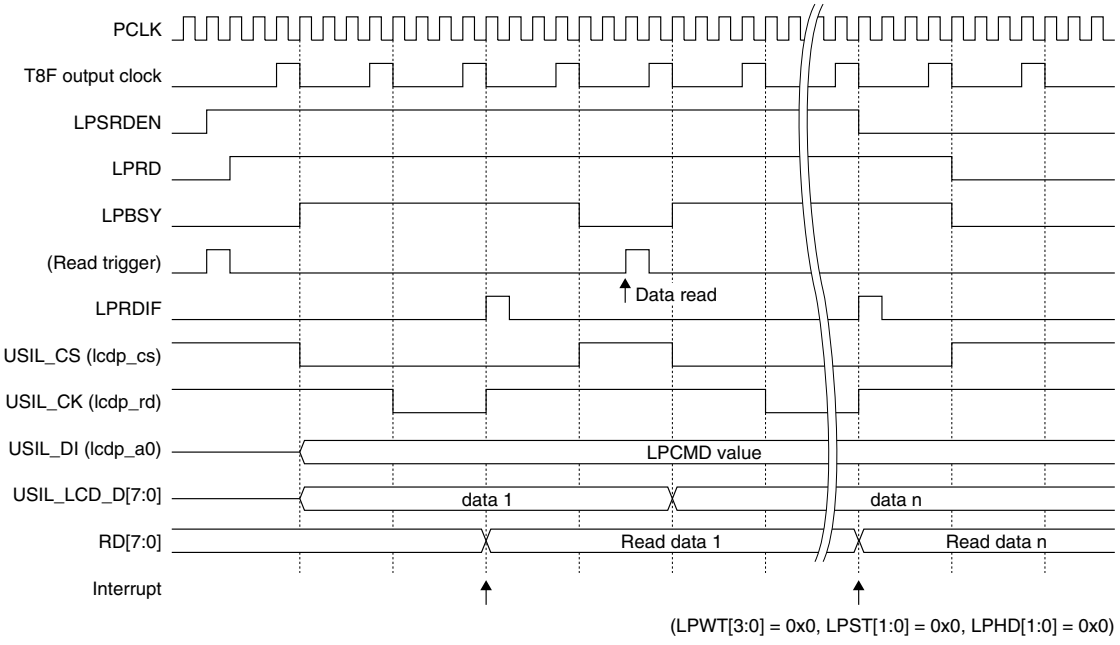


Figure 21.5.5.3 Successive Read Timing Chart (LCD parallel mode)

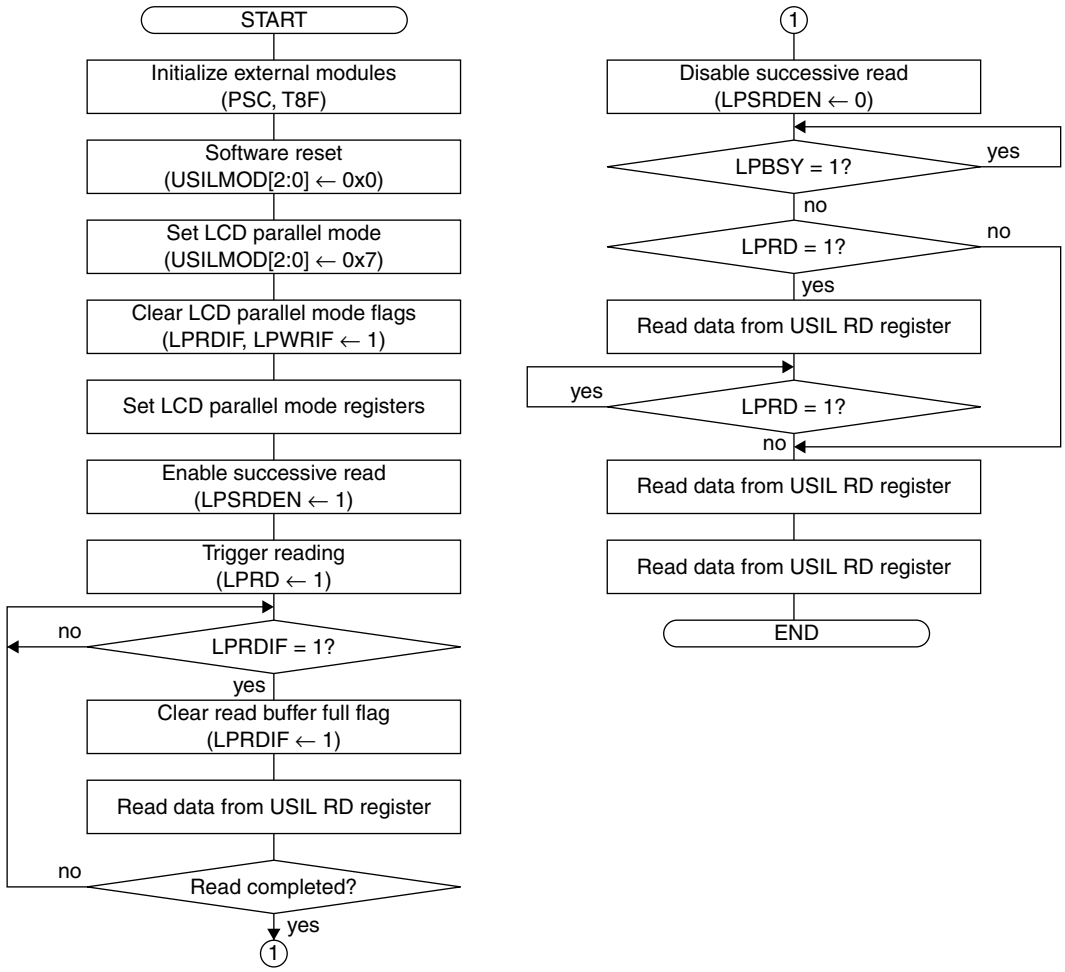


Figure 21.5.5.4 Successive Read Flow Chart (LCD parallel mode)

21.6 Receive Errors

In UART mode, three different receive errors (overrun error, framing error, and parity error) may be detected while receiving data. In SPI and I²C modes, overrun errors may be detected while receiving data.
Since receive errors are interrupt causes, they can be processed by generating interrupts. For more information on interrupt control, see Section 21.7.

Overrun error (UART, SPI, I²C master/slave modes)

UART mode

An overrun error occurs if the next reception is completed when URDIF is 1 and the receive data buffer (USIL_RD register) is not read (an overrun error occurs at the time stop bit has been received).
When an overrun error occurs, the overrun error flag (UOEIF/USIL_UIF register) is set to 1. The receiving operation continues even if this error occurs. To reset UOEIF, perform USIL software reset (write 0x0 to USILMOD[2:0]/USIL_GCFG register) to initialize USIL.

SPI mode

An overrun error occurs if data are received successively when SRDIF is 1. While SRDIF is set to 1, the next received data will not be transferred from the shift register to the receive data buffer (the first byte data exists in the receive data buffer and the second byte data exists in the shift register). An overrun error occurs if the third byte data is received in this condition, as the second byte data in the shift register is corrupted (an overrun error occurs at the time the first bit of the third byte is fetched).

When an overrun error occurs, the overrun error flag (SEIF/USIL_SIF register) is set to 1. The receiving operation continues even if this error occurs. SEIF is reset by writing 1. To reset an overrun error, write 1 to SEIF and then read the receive data buffer (USIL_RD register) twice. The procedure that writes 1 to SEIF and reads USIL_RD register twice can be reversed.

I²C master/slave mode

An overrun error occurs when a transmit or receive trigger is issued after two-byte data has been received (the first byte data exists in the receive data buffer and the second byte data exists in the shift register) without the receive data buffer being read.
When an overrun error occurs, the overrun error flag (IMEIF/USIL_IMIF register for I²C master mode or ISEIF/USIL_ISIF register for I²C slave mode) is set to 1. The receiving operation continues even if this error occurs. IMEIF/ISEIF is reset by writing 1. To reset an overrun error, write 1 to IMEIF/ISEIF and then read the receive data buffer (USIL_RD register) twice.

Framing error (UART mode only)

If the stop bit is received as 0 in UART mode, the UART controller determines loss of sync and a framing error occurs. If the stop bit is configured to two bits, only the first bit is checked.
The framing error flag (USEIF/USIL_UIF register) is set to 1 if this error occurs. The received data is still transferred to the receive data buffer if this error occurs and the receiving operation continues, but the data cannot be guaranteed, even if no framing error occurs for subsequent data receiving. The framing error flag is reset to 0 by writing 1.

Parity error (UART mode only)

If UPREN/USIL_UCFG register has been set to 1 (parity enabled), data received is checked for parity in UART mode. Data received in the shift register is checked for parity when sent to the receive data buffer. The matching is checked against the UPMD/USIL_UCFG register setting (odd or even parity). If the result is a non-match, a parity error is issued, and the parity error flag (UPEIF/USIL_UIF register) is set to 1. Even if this error occurs, the data received is sent to the receive data buffer, and the receiving operation continues. However, the received data cannot be guaranteed if a parity error occurs. The UPEIF flag is reset to 0 by writing 1.

21.7 USIL Interrupts and DMA

This section describes the USIL interrupts generated in each interface mode and invoking DMA.
For more information on interrupt processing and DMA transfer, see the “Interrupt Controller (ITC)” chapter and the “DMA Controller (DMAC)” chapter, respectively.

USIL outputs one interrupt signal shared by the all interrupt causes to the interrupt controller (ITC). Inspect the interrupt flags available in each mode to determine the interrupt cause occurred.

21.7.1 Interrupts in UART Mode

The UART mode includes a function for generating the following three different types of interrupts.
- Transmit buffer empty interrupt
- Receive buffer full interrupt
- Receive error interrupt

Transmit buffer empty interrupt

To use this interrupt, set UTDIE/USIL_UIE register to 1. If UTDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When transmit data written to the transmit data buffer is transferred to the shift register, the USIL module sets UTDIF/USIL_UIF register to 1, indicating that the transmit data buffer is empty. If transmit buffer empty interrupts are enabled (UTDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the UTDIF flag in the interrupt handler routine to determine whether the USIL (UART mode) interrupt is attributable to a transmit buffer empty. If UTDIF is 1, the next transmit data can be written to the transmit data buffer by the interrupt handler routine.

Receive buffer full interrupt

To use this interrupt, set URDIE/USIL_UIE register to 1. If URDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

If a received data is loaded into the receive data buffer, the USIL module sets URDIF/USIL_UIF register to 1. If receive buffer full interrupts are enabled (URDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the URDIF flag in the interrupt handler routine to determine whether the USIL (UART mode) interrupt is attributable to a receive buffer full. If URDIF is 1, the received data can be read from the receive data buffer by the interrupt handler routine. However, be sure to check whether a receive error has occurred or not.

Receive error interrupt

To use this interrupt, set UEIE/USIL_UIE register to 1. If UEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

The USIL module sets an error flag (UPEIF/USIL_UIF register, USEIF/USIL_UIF register, or UOEIF/USIL_UIF register) to 1 if a parity error, framing error, or overrun error is detected when receiving data. If receive error interrupts are enabled (UEIE = 1), an interrupt request is sent simultaneously to the ITC. If other interrupt conditions are satisfied, an interrupt occurs. You can inspect the UPEIF, USEIF, and UOEIF flags in the interrupt handler routine to determine whether the USIL (UART mode) interrupt was caused by a receive error. If any of the error flags has the value 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, perform USIL software reset (write 0x0 to USILMOD[2:0]/USIL_GCFG register) to initialize USIL.

21.7.2 Interrupts in SPI Mode

The SPI master/slave modes include a function for generating the following three different types of interrupts.

- Transmit buffer empty interrupt
- Receive buffer full interrupt
- Receive error interrupt

Transmit buffer empty interrupt

To use this interrupt, set STDIE/USIL_SIE register to 1. If STDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When transmit data written to the transmit data buffer is transferred to the shift register, the USIL module sets STDIF/USIL_SIF register to 1, indicating that the transmit data buffer is empty. If transmit buffer empty interrupts are enabled (STDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the STDIF flag in the interrupt handler routine to determine whether the USIL (SPI master/slave mode) interrupt is attributable to a transmit buffer empty. If STDIF is 1, the next transmit data can be written to the transmit data buffer by the interrupt handler routine.

Receive buffer full interrupt

To use this interrupt, set SRDIE/USIL_SIE register to 1. If SRDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

If a received data is loaded into the receive data buffer, the USIL module sets SRDIF/USIL_SIF register to 1. If receive buffer full interrupts are enabled (SRDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the SRDIF flag in the interrupt handler routine to determine whether the USIL (SPI master/slave mode) interrupt is attributable to a receive buffer full. If SRDIF is 1, the received data can be read from the receive data buffer by the interrupt handler routine. However, be sure to check whether a receive error has occurred or not.

Receive error interrupt

To use this interrupt, set SEIE/USIL_SIE register to 1. If SEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

The USIL module sets SEIF/USIL_SIF register to 1 if an overrun error is detected when receiving data. If receive error interrupts are enabled (SEIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the SEIF flags in the interrupt handler routine to determine whether the USIL (SPI master/slave mode) interrupt was caused by a receive error. If SEIF is 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, clear SEIF by writing 1 and then read the receive data buffer (USIL_RD register) twice.

21.7.3 Interrupts in I²C Master Mode

The I²C master mode includes a function for generating the following two different types of interrupts.

- Operation completion interrupt
- Receive error interrupt

Operation completion interrupt

To use this interrupt, set IMIE/USIL_IMIE register to 1. If IMIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When the operation that initiated by a software trigger has completed, the USIL module sets IMIF/USIL_IMIF register to 1. If operation completion interrupts are enabled (IMIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the IMSTA[2:0]/USIL_IMIF register in the interrupt handler routine to determine the I²C operation/status that causes the interrupt.

Table 21.7.3.1 I²C Master Status Bits

IMSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been generated.
0x0	Start condition has been generated.

(Default: 0x0)

Receive error interrupt

To use this interrupt, set IMEIE/USIL_IMIE register to 1. If IMEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

An overrun error occurs at the time a transmit or receive trigger is issued after two-byte data has been received without reading the receive data buffer.

The USIL module sets IMEIF/USIL_IMIF register to 1 if an overrun error is detected when receiving data. If receive error interrupts are enabled (IMEIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the IMEIF flags in the interrupt handler routine to determine whether the USIL (I²C master mode) interrupt was caused by a receive error. If IMEIF is 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, clear IMEIF by writing 1, and then read the receive data buffer (USIL_RD register) twice.

21.7.4 Interrupts in I²C Slave Mode

The I²C slave mode includes a function for generating the following two different types of interrupts.
• Operation completion interrupt
• Receive error interrupt

Operation completion interrupt

To use this interrupt, set ISIE/USIL_ISIE register to 1. If ISIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When the operation that initiated by a software trigger has completed, the USIL module sets ISIF/USIL_ISIF register to 1. If operation completion interrupts are enabled (ISIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the ISSTA[2:0]/USIL_ISIF register in the interrupt handler routine to determine the I²C operation/status that causes the interrupt.

Table 21.7.4.1 I²C Slave Status Bits

ISSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been detected.
0x0	Start condition has been detected.

(Default: 0x0)

Receive error interrupt

To use this interrupt, set ISEIE/USIL_ISIE register to 1. If ISEIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

An overrun error occurs at the time a transmit or receive trigger is issued after two-byte data has been received without reading the receive data buffer.

The USIL module sets ISEIF/USIL_ISIF register to 1 if an overrun error is detected when receiving data. If receive error interrupts are enabled (ISEIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the ISEIF flags in the interrupt handler routine to determine whether the USIL (I²C slave mode) interrupt was caused by a receive error. If ISEIF is 1, the interrupt handler routine will proceed with error recovery.

To reset an overrun error, clear ISEIF by writing 1, and then read the receive data buffer (USIL_RD register) twice.

21.7.5 Interrupts in LCD SPI Mode

The LCD SPI mode includes a function for generating the following interrupt.
• Transmit buffer empty interrupt

Transmit buffer empty interrupt

To use this interrupt, set LSTDIE/USIL_LSIE register to 1. If LSTDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When transmit data written to the transmit data buffer is transferred to the shift register, the USIL module sets LSTDIF/USIL_LSIF register to 1, indicating that the transmit data buffer is empty. If transmit buffer empty interrupts are enabled (LSTDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the LSTDIF flag in the interrupt handler routine to determine whether the USIL (LCD SPI mode) interrupt is attributable to a transmit buffer empty. If LSTDIF is 1, the next transmit data can be written to the transmit data buffer by the interrupt handler routine.

21.7.6 Interrupts in LCD Parallel Mode

The LCD parallel mode includes a function for generating the following two different types of interrupts.
• Write buffer empty interrupt
• Read buffer full interrupt

Write buffer empty interrupt

To use this interrupt, set LPWRIE/USIL_LPIE register to 1. If LPWRIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When data written to the write (transmit data) buffer is output via the USIL_LCD_D[7:0] pins , the USIL module sets LPWRIF/USIL_LPIF register to 1, indicating that the write buffer is empty. If write buffer empty interrupts are enabled (LPWRIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the LPWRIF flag in the interrupt handler routine to determine whether the USIL (LCD parallel mode) interrupt is attributable to a write buffer empty. If LPWRIF is 1, the next data can be written to the write buffer by the interrupt handler routine.

Read buffer full interrupt

To use this interrupt, set LPRDIE/USIL_LPIE register to 1. If LPRDIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
If data sent from the LCD driver/panel is loaded into the read (receive data) buffer, the USIL module sets LPRDIF/USIL_LPIF register to 1. If read buffer full interrupts are enabled (LPRDIE = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. You can inspect the LPRDIF flag in the interrupt handler routine to determine whether the USIL (LCD parallel mode) interrupt is attributable to a read buffer full. If LPRDIF is 1, data can be read from the read buffer by the interrupt handler routine.

21.7.7 DMA Transfer

The causes of receive (read) buffer full and transmit (write) buffer empty interrupts in UART, SPI master/slave, LCD SPI, and LCD parallel modes can invoke a DMA. This allows continuous data transmission/reception through DMA transfer between memory and transmit/receive data buffers. These interrupt signals are output to both the ITC and DMAC. Therefore, DMA transfer can be performed without generating any USIL interrupt.
The following lists the DMAC channels that allow selection of a USIL interrupt cause as the trigger.

USIL receive (read) buffer full: DMAC Ch.6
USIL transmit (write) buffer empty: DMAC Ch.7

For more information on DMA transfer, see the “DMA Controller (DMAC)” chapter.

Note: The USIL module cannot invoke a DMA in I²C master and slave mode.

21.8 Control Register Details

Table 21.8.1 List of USIL Registers

Address	Register name		Function
0x300600	USIL_GCFG	USIL Global Configuration Register	Set interface and MSB/LSB modes
0x300601	USIL_TD	USIL Transmit Data Buffer Register	Transmit data buffer
0x300602	USIL_RD	USIL Receive Data Buffer Register	Receive data buffer
0x300640	USIL_UCFG	USIL UART Mode Configuration Register	Set UART transfer conditions
0x300641	USIL_UIE	USIL UART Mode Interrupt Enable Register	Enable/disable UART interrupts
0x300642	USIL_UIF	USIL UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
0x300650	USIL_SCFG	USIL SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
0x300651	USIL_SIE	USIL SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
0x300652	USIL_SIF	USIL SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
0x300660	USIL_IMTG	USIL I²C Master Mode Trigger Register	Start I²C master operations
0x300661	USIL_IMIE	USIL I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
0x300662	USIL_IMIF	USIL I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
0x300670	USIL_ISTG	USIL I²C Slave Mode Trigger Register	Start I²C slave operations
0x300671	USIL_ISIE	USIL I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
0x300672	USIL_ISIF	USIL I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status

Address	Register name		Function
0x300680	USIL_LSCFG	USIL LCD SPI Mode Configuration Register	Set LCD SPI transfer conditions
0x300681	USIL_LSIE	USIL LCD SPI Mode Interrupt Enable Register	Enable/disable LCD SPI interrupts
0x300682	USIL_LSIF	USIL LCD SPI Mode Interrupt Flag Register	Indicate LCD SPI interrupt cause status
0x30068f	USIL_LSDCFG	USIL LCD SPI Mode Data Configuration Register	Select display data format
0x300690	USIL_LPCFG	USIL LCD Parallel I/F Mode Configuration Register	Set LCD parallel interface conditions
0x300691	USIL_LPIE	USIL LCD Parallel I/F Mode Interrupt Enable Register	Enable/disable LCD parallel interface interrupts
0x300692	USIL_LPIF	USIL LCD Parallel I/F Mode Interrupt Flag Register	Indicate LCD parallel interface interrupt cause status
0x30069f	USIL_LPAC	USIL LCD Parallel I/F Mode Access Timing Register	Set LCD parallel interface access timing parameters

The USIL registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

USIL Global Configuration Register (USIL_GCFG)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
USIL Global Configuration Register (USIL_GCFG)	**0x300600** (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3	**LSBFST**	MSB/LSB first mode select	1 MSB first	0 LSB first	0	R/W	
		D2–0	**USILMOD [2:0]**	Interface mode configuration	USILMOD[2:0]	I/F mode	0x0	R/W	
					0x7	LCD Parallel			
					0x6	LCD SPI			
					0x5	I²C slave			
					0x4	I²C master			
					0x3	SPI slave			
					0x2	SPI master			
					0x1	UART			
					0x0	Software reset			

Note: This register must be configured before setting other USIL registers.

D[7:4] Reserved

D3 LSBFST: MSB/LSB First Mode Select Bit

Selects whether serial data will be transferred from the MSB or LSB.
1 (R/W): MSB first
0 (R/W): LSB first (default)

This setting affects all interface modes.

D[2:0] USILMOD[2:0]: Interface Mode Configuration Bits

Selects an interface mode.

Table 21.8.2 Interface Mode Selection

USILMOD[2:0]	Interface mode
0x7	LCD parallel
0x6	LCD SPI
0x5	I²C slave
0x4	I²C master
0x3	SPI slave
0x2	SPI master
0x1	UART
0x0	Software reset

(Default: 0x0)

Perform software reset (set USILMOD[2:0] to 0x0) and then set the interface mode before changing other USIL configurations.

Notes: 1. Setting USILMOD[2:0] to 0x0 (software reset) resets all the USIL register contents to the initial values.

2. The USIL transmitter and receiver circuits will be able to operate after USILMOD[2:0] is set to a value (0x1 to 0x7) other than 0x0.

3. The USIL registers for an interface mode are available only when USILMOD[2:0] is set to the mode setting value.

Table 21.8.3 Corresponding Between USILMOD[2:0] Settings and Available Registers

USIL register	USILMOD[2:0]						
	0x1	0x2	0x3	0x4	0x5	0x6	0x7
LCD parallel mode registers	–	–	–	–	–	–	Available
LCD SPI mode registers	–	–	–	–	–	Available	–
I²C slave mode registers	–	–	–	–	Available	–	–
I²C master mode registers	–	–	–	Available	–	–	–
SPI master/slave mode registers	–	Available	Available	–	–	–	–
UART mode registers	Available	–	–	–	–	–	–

USIL Transmit Data Buffer Register (USIL_TD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL Transmit Data Buffer Register (USIL_TD)	**0x300601** (8 bits)	D7–0	**TD[7:0]**	USIL transmit data buffer TD7 = MSB TD0 = LSB	0x0 to 0xff	0x0	R/W	

D[7:0] TD[7:0]: USIL Transmit Data Buffer Bits

Sets transmit data to be written to the transmit data buffer. (Default: 0x0)

In UART, SPI master, LCD SPI, and LCD parallel modes, transmission begins immediately after writing data to this register. In SPI slave mode, transmission will begin when the clock is input from the SPI master device.

In I²C master/slave mode, transmission begins by the software trigger for data transmission.

The data written to this register is converted into serial data through the shift register and is output from the USIL_DO pin with the bit set to 1 as high level and the bit set to 0 as low level.

In LCD parallel mode, the data written to this register is output via the USIL_LCD_D[7:0] pins.

A transmit buffer empty interrupt can be generated when data written to this register has been transferred to the shift register or output from the USIL_LCD_D[7:0] pins. The subsequent transmit data can then be written, even while data is being sent.

USIL Receive Data Buffer Register (USIL_RD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL Receive Data Buffer Register (USIL_RD)	**0x300602** (8 bits)	D7–0	**RD[7:0]**	USIL receive data buffer RD7 = MSB RD0 = LSB	0x0 to 0xff	0x0	R	

D[7:0] RD[7:0]: USIL Receive Data Buffer Bits

Contains the received data. (Default: 0x0)

Serial data input from the USIL_DI pin is converted to parallel, with the high level bit set to 1 and the low level bit set to 0, and then it is loaded to this register. In LCD parallel mode, data input from the USIL_LCD_D[7:0] pins are loaded to this register.

A receive buffer full interrupt can be generated when the data received has been loaded to this register. Data can then be read until subsequent data is received.

This register is read-only.

USIL UART Mode Configuration Register (USIL_UCFG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL UART Mode Configuration Register (USIL_UCFG)	**0x300640** (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	**UCHLN**	Character length select	1	8 bits	0	7 bits	0	R/W	
		D2	**USTPB**	Stop bit select	1	2 bits	0	1 bit	0	R/W	
		D1	**UPMD**	Parity mode select	1	Even	0	Odd	0	R/W	
		D0	**UPREN**	Parity enable	1	With parity	0	No parity	0	R/W	

Note: This register is effective only in UART mode. Configure USIL to UART mode before setting this register.

D[7:4] Reserved

D3 UCHLN: Character Length Select Bit

Selects the serial transfer data length.
1 (R/W): 8 bits
0 (R/W): 7 bits (default)

When 7-bit data length is selected, D7 in the transmit data buffer is ignored and D7 in the receive data buffer is always set to 0.

D2 USTPB: Stop Bit Select Bit

Selects the stop bit length.
1 (R/W): 2 bits
0 (R/W): 1 bit (default)

Writing 1 to USTPB selects 2 stop bits; writing 0 to it selects 1 bit. The start bit is fixed at 1 bit.

D1 UPMD: Parity Mode Select Bit

Selects the parity mode.
1 (R/W): Even parity
0 (R/W): Odd parity (default)

Parity checking and parity bit addition are enabled only when UPREN is set to 1. The UPMD setting is disabled if UPREN is 0.

D0 UPREN: Parity Enable Bit

Enables the parity function.
1 (R/W): With parity
0 (R/W): No parity (default)

UPREN is used to select whether received data parity checking is performed and whether a parity bit is added to transmit data. Setting UPREN to 1 parity-checks the received data. A parity bit is automatically added to the transmit data. If UPREN is set to 0, no parity bit is checked or added.

USIL UART Mode Interrupt Enable Register (USIL_UIE)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL UART Mode Interrupt Enable Register (USIL_UIE)	0x300641 (8 bits)	D7–3	–	reserved	–				–	–	0 when being read.
		D2	UEIE	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	URDIE	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	UTDIE	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in UART mode. Configure USIL to UART mode before this register can be used.

D[7:3] Reserved

D2 UEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when a receive error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process receive errors using interrupts.

D1 URDIE: Receive Buffer Full Interrupt Enable Bit

Enables interrupt requests to the ITC when received data is loaded to the receive data buffer.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to read received data using interrupts.

D0 UTDIE: Transmit Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC when data written to the transmit data buffer is sent to the shift register (i.e. when data transmission begins).
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the transmit data buffer using interrupts.

USIL UART Mode Interrupt Flag Register (USIL_UIF)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL UART Mode Interrupt Flag Register (USIL_UIF)	0x300642 (8 bits)	D7	–	reserved	–				–	–	0 when being read.
		D6	URBSY	Receive busy flag	1	Busy	0	Idle	0	R	
		D5	UTBSY	Transmit busy flag	1	Busy	0	Idle	0	R	
		D4	UPEIF	Parity error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D3	USEIF	Framing error flag	1	Error	0	Normal	0	R/W	
		D2	UOEIF	Overrun error flag	1	Error	0	Normal	0	R/W	
		D1	URDIF	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	UTDIF	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	

Note: This register is effective only in UART mode. Configure USIL to UART mode before this register can be used.

D7 Reserved

D6 URBSY: Receive Busy Flag Bit

Indicates the receive shift register status.

1 (R): Busy
0 (R): Idle (default)

URBSY is set to 1 when the first start bit is detected (when data reception begins) and is reset to 0 when the data received in the shift register is loaded into the receive data buffer. Inspect URBSY to determine whether the receiving circuit is operating or at standby.

D5 UTBSY: Transmit Busy Flag Bit

Indicates the transmit shift register status.

1 (R): Busy
0 (R): Idle (default)

UTBSY is set to 1 when transmit data is loaded from the transmit data buffer into the shift register and is reset to 0 when the data transfer is completed.

D4 UPEIF: Parity Error Flag Bit

Indicates whether a parity error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

UPEIF is set to 1 when a parity error occurs. At the same time a receive error interrupt request is sent to the ITC if UEIE/USIL_UIE register is 1. Parity checking is enabled only when UPREN/USIL_UCFG register is set to 1 and is performed when received data is transferred from the shift register to the receive data buffer. UPEIF is reset by writing 1.

D3 USEIF: Framing Error Flag Bit

Indicates whether a framing error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

USEIF is set to 1 when a framing error occurs. At the same time a receive error interrupt request is sent to the ITC if UEIE/USIL_UIE register is 1. A framing error occurs when data is received with the stop bit set to 0. USEIF is reset by writing 1.

D2 UOEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

UOEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if UEIE/USIL_UIE register is 1. An overrun error occurs if the next reception is completed when URDIF is 1 and the receive data buffer (USIL_RD register) is not read (an overrun error occurs at the time stop bit has been received). To reset UOEIF, perform USIL software reset (write 0x0 to USILMOD[2:0]/USIL_GCFG register) to initialize USIL.

D1 URDIF: Receive Buffer Full Flag Bit

Indicates the receive data buffer status.

1 (R): Data full
0 (R): No data (default)
1 (W): Reset to 0
0 (W): Ignored

URDIF is set to 1 when data received in the shift register is sent to the receive data buffer (when receiving is completed), indicating that the data can be read. At the same time a receive buffer full interrupt request is sent to the ITC if URDIE/USIL_UIE register is 1. URDIF is reset by writing 1.

D0 UTDIF: Transmit Data Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Empty (default)
0 (R): Data exists
1 (W): Reset to 0
0 (W): Ignored

UTDIF is set to 1 when the transmit data written to the transmit data buffer is transferred to the shift register (when transmission starts), indicating that the next transmit data can be written to. At the same time a transmit buffer empty interrupt request is sent to the ITC if UTDIE/USIL_UIE register is 1. UTDIF is reset by writing 1.

USIL SPI Master/Slave Mode Configuration Register (USIL_SCFG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL SPI Master/Slave Mode Configuration Register (USIL_SCFG)	0x300650 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SCPHA	Clock phase select	1	Phase 1	0	Phase 0	0	R/W	
		D2	SCPOL	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	–	reserved	–				–	–	Do not set to 1.
		D0	SFSTMOD	Fast mode select	1	Fast	0	Normal	0	R/W	

Note: This register is effective only in SPI master and slave modes. Configure USIL to SPI master/slave mode before this register can be used.

D[7:4] Reserved

D3 SCPHA: Clock Phase Select Bit

Selects the SPI clock phase.

1 (R/W): Phase 1
0 (R/W): Phase 0 (default)

Set the data transfer timing together with SCPOL. (See Figure 21.8.1.)

D2 SCPOL: Clock Polarity Select Bit

Selects the SPI clock polarity.

1 (R/W): Active low
0 (R/W): Active high (default)

Set the data transfer timing together with SCPHA. (See Figure 21.8.1.)

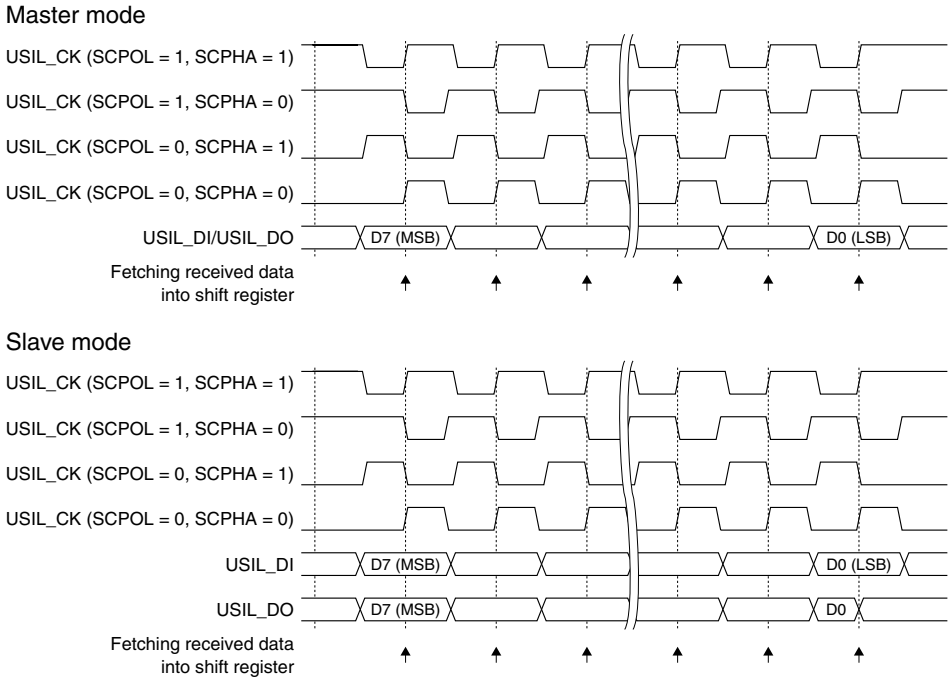


Figure 21.8.1 Clock and Data Transfer Timing

D1 Reserved (Do not set to 1.)

D0 SFSTMOD: Fast Mode Select Bit (for SPI master mode)

Selects Fast mode.
1 (R/W): Fast mode
0 (R/W): Normal mode (default)

In SPI master mode, either normal or fast clock mode can be selected using SFSTMOD. Setting SFSTMOD to 0 (default) places the USIL into normal mode and the USIL generates the transfer clock by dividing the T8F output by 2. Setting SFSTMOD to 1 places the USIL into fast mode and the USIL uses PCLK supplied from the CMU directly as the transfer clock. The fast mode does not use the T8F.
The SPI slave mode uses the T8F output clock for generating the sampling clock.

USIL SPI Master/Slave Mode Interrupt Enable Register (USIL_SIE)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL SPI Master/Slave Mode Interrupt Enable Register (USIL_SIE)	0x300651 (8 bits)	D7–3	–	reserved	–				–	–	0 when being read.
		D2	SEIE	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	SRDIE	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	STDIE	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in SPI master and slave modes. Configure USIL to SPI master/slave mode before this register can be used.

D[7:3] Reserved

D2 SEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when an overrun error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process overrun errors using interrupts.

D1 SRDIE: Receive Buffer Full Interrupt Enable Bit

Enables interrupt requests to the ITC when received data is loaded to the receive data buffer.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to read received data using interrupts.

D0 STDIE: Transmit Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC when data written to the transmit data buffer is sent to the shift register (i.e. when data transmission begins).
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the transmit data buffer using interrupts.

USIL SPI Master/Slave Mode Interrupt Flag Register (USIL_SIF)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL SPI Master/Slave Mode Interrupt Flag Register (USIL_SIF)	0x300652 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SSIF	Transfer busy flag (master)	1	Busy	0	Idle	0	R	
				ss signal low flag (slave)	1	ss = H	0	ss = L			
		D2	SEIF	Overrun error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D1	SRDIF	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	STDIF	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	

Note: This register is effective only in SPI master and slave modes. Configure USIL to SPI master/slave mode before this register can be used.

D[7:4] Reserved

D3 SSIF: Transfer Busy Flag Bit (Master Mode)/ss Signal Low Flag Bit (Slave Mode)

Master mode

Indicates the SPI transfer status.
1 (R): Operating
0 (R): Standby (default)

SSIF is set to 1 when the SPI starts data transfer in master mode and is maintained at 1 while transfer is underway. It is cleared to 0 once the transfer is completed.

Slave mode

Indicates the slave select (USIL_CS) signal status.
1 (R): High level (this SPI is not selected)
0 (R): Low level (this SPI is selected) (default)

SSIF is set to 0 when the master device asserts the slave select (USIL_CS) signal to select this SPI controller (slave device). It is returned to 1 when the master device clears the SPI controller selection by negating the slave select (USIL_CS) signal.

D2 SEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.
1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

SEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if SEIE/USIL_SIE register is 1. An overrun error occurs if data are received successively when SRDIF is 1. While SRDIF is set to 1, the next received data will not be transferred from the shift register to the receive data buffer (the first byte data exists in the receive data buffer and the second byte data exists in the shift register). An overrun error occurs if the third byte data is received in this condition, as the second byte data in the shift register is corrupted (an overrun error occurs at the time the first bit of the third byte is fetched).

SEIF is reset by writing 1. To reset an overrun error, write 1 to SEIF and then read the receive data buffer (USIL_RD register) twice. The procedure that writes 1 to SEIF and reads USIL_RD register twice can be reversed.

D1 SRDIF: Receive Buffer Full Flag Bit

Indicates the receive data buffer status.

1 (R): Data full
0 (R): No data (default)
1 (W): Reset to 0
0 (W): Ignored

SRDIF is set to 1 when data received in the shift register is sent to the receive data buffer (when receiving is completed), indicating that the data can be read. At the same time a receive buffer full interrupt request is sent to the ITC if SRDIE/USIL_SIE register is 1. SRDIF is reset by writing 1.

D0 STDIF: Transmit Buffer Empty Flag Bit

Indicates the transmit data buffer status.

1 (R): Empty (default)
0 (R): Data exists
1 (W): Reset to 0
0 (W): Ignored

STDIF is set to 1 when the transmit data written to the transmit data buffer is transferred to the shift register (when transmission starts), indicating that the next transmit data can be written to. At the same time a transmit buffer empty interrupt request is sent to the ITC if STDIE/USIL_SIE register is 1. STDIF is reset by writing 1.

USIL I²C Master Mode Trigger Register (USIL_IMTG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Master Mode Trigger Register (USIL_IMTG)	0x300660 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	IMTG	I²C master operation trigger	1 Trigger / 0 Ignored	0	W	
					1 Waiting / 0 Finished		R	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	IMTGMOD [2:0]	I²C master trigger mode select	IMTGMOD[2:0] / Trigger mode	0x0	R/W	
					0x7 reserved			
					0x6 Receive ACK/NAK			
					0x5 Transmit NAK			
					0x4 Transmit ACK			
					0x3 Receive data			
					0x2 Transmit data			
					0x1 Stop condition			
					0x0 Start condition			

Note: This register is effective only in I²C master mode. Configure USIL to I²C master mode before this register can be used.

D[7:5] Reserved

D4 IMTG: I²C Master Operation Trigger Bit

Starts an I²C master operation.

1 (W): Trigger
0 (W): Ignored
1 (R): Waiting for starting operation
0 (R): Trigger has finished (default)

Select an I²C master operation using IMTGMOD[2:0] and write 1 to IMTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

D3 Reserved

D[2:0] IMTGMOD[2:0]: I²C Master Trigger Mode Select Bits

Selects an I²C master operation.

Table 21.8.4 Trigger List in I²C Master Mode

IMTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception
0x2	Data transmission
0x1	Stop condition
0x0	Start condition

(Default: 0x0)

USIL I²C Master Mode Interrupt Enable Register (USIL_IMIE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Master Mode Interrupt Enable Register (USIL_IMIE)	0x300661 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	IMEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	IMIE	Operation completion int. enable	1 Enable 0 Disable	0	R/W	

Note: This register is effective only in I²C master mode. Configure USIL to I²C master mode before this register can be used.

D[7:2] Reserved

D1 IMEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when an overrun error occurs.

1 (R/W): Enabled

0 (R/W): Disabled (default)

Set this bit to 1 to process overrun errors using interrupts.

D0 IMIE: Operation Completion Interrupt Enable Bit

Enables interrupt requests to the ITC when the triggered operation has completed.

1 (R/W): Enabled

0 (R/W): Disabled (default)

Set this bit to 1 to confirm whether the triggered operation has completed or not using interrupts.

USIL I²C Master Mode Interrupt Flag Register (USIL_IMIF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Master Mode Interrupt Flag Register (USIL_IMIF)	0x300662 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	IMBSY	I²C master busy flag	1 Busy 0 Standby	0	R	
		D4–2	IMSTA[2:0]	I²C master status	IMSTA[2:0] Status: 0x7 reserved; 0x6 NAK received; 0x5 ACK received; 0x4 ACK/NAK sent; 0x3 End of Rx data; 0x2 End of Tx data; 0x1 Stop generated; 0x0 Start generated	0x0	R	
		D1	IMEIF	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	IMIF	Operation completion flag	1 Completed 0 Not completed	0	R/W	

Note: This register is effective only in I²C master mode. Configure USIL to I²C master mode before this register can be used.

D[7:6] Reserved

D5 IMBSY: I²C Master Busy Flag Bit

Indicates the I²C master operation status.

1 (R): Busy

0 (R): Standby (default)

Writing 1 to IMTG/USIL_IMTG register (starting an I²C master operation) sets IMBSY to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, IMBSY is reset to 0.

D[4:2] IMSTA[2:0]: I²C Master Status Bits

Indicates the I²C master status.

Table 21.8.5 I²C Master Status Bits

IMSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been generated.
0x0	Start condition has been generated.

(Default: 0x0)

When an operation completion interrupt occurs, read IMSTA[2:0] to check the operation that has been finished. IMSTA[2:0] is automatically reset to 0x0 by writing 1 to IMIF.

D1 IMEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred
0 (R): No error (default)
1 (W): Reset to 0
0 (W): Ignored

IMEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if IMEIE/USIL_IMIE register is 1.

An overrun error occurs when a transmit or receive trigger is issued after two-byte data has been received (the first byte data exists in the receive data buffer and the second byte data exists in the shift register) without the receive data buffer being read.

IMEIF is reset by writing 1.

To reset an overrun error, clear IMEIF by writing 1, and then read the receive data buffer (USIL_RD register) twice.

D0 IMIF: Operation Completion Flag Bit

Indicates whether the triggered operation has completed or not.

1 (R): Completed
0 (R): Not completed (default)
1 (W): Reset to 0
0 (W): Ignored

IMIF is set to 1 when the operation that is specified and triggered using the USIL_IMTG register has completed. At the same time an operation completion interrupt request is sent to the ITC if IMIE/USIL_IMIE register is 1. IMIF is reset by writing 1.

USIL I²C Slave Mode Trigger Register (USIL_ISTG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Slave Mode Trigger Register (USIL_ISTG)	0x300670 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	ISTG	I²C slave operation trigger	1 Trigger / 0 Ignored	0	W	
					1 Waiting / 0 Finished		R	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	ISTGMOD [2:0]	I²C slave trigger mode select	ISTGMOD[2:0]: 0x7 reserved; 0x6 Receive ACK/NAK; 0x5 Transmit NAK; 0x4 Transmit ACK; 0x3 Receive data/ Detect stop; 0x2 Transmit data; 0x1 reserved; 0x0 Wait for start	0x0	R/W	

Note: This register is effective only in I²C slave mode. Configure USIL to I²C slave mode before this register can be used.

D[7:5] Reserved

D4 ISTG: I²C Slave Operation Trigger Bit

Starts an I²C slave operation.
1 (W): Trigger
0 (W): Ignored
1 (R): Waiting for starting operation
0 (R): Trigger has finished (default)

Select an I²C slave operation using ISTGMOD[2:0] and write 1 to ISTG as the trigger. The I²C controller controls the I²C bus to generate the specified operating status.

D3 Reserved

D[2:0] ISTGMOD[2:0]: I²C Slave Trigger Mode Select Bits

Selects an I²C slave operation.

Table 21.8.6 Trigger List in I²C Slave Mode

ISTGMOD[2:0]	Trigger
0x7	Reserved
0x6	ACK/NAK reception
0x5	NAK transmission
0x4	ACK transmission
0x3	Data reception/stop condition detection
0x2	Data transmission
0x1	Reserved
0x0	Wait for start condition

(Default: 0x0)

USIL I²C Slave Mode Interrupt Enable Register (USIL_ISIE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Slave Mode Interrupt Enable Register (USIL_ISIE)	0x300671 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	ISEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	ISIE	Operation completion int. enable	1 Enable 0 Disable	0	R/W	

Note: This register is effective only in I²C slave mode. Configure USIL to I²C slave mode before this register can be used.

D[7:2] Reserved

D1 ISEIE: Receive Error Interrupt Enable Bit

Enables interrupt requests to the ITC when an overrun error occurs.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to process overrun errors using interrupts.

D0 ISIE: Operation Completion Interrupt Enable Bit

Enables interrupt requests to the ITC when the triggered operation has completed.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to confirm whether the triggered operation has completed or not using interrupts.

USIL I²C Slave Mode Interrupt Flag Register (USIL_ISIF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Slave Mode Interrupt Flag Register (USIL_ISIF)	0x300672 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	ISBSY	I²C slave busy flag	1 Busy / 0 Standby	0	R	
		D4–2	ISSTA[2:0]	I²C slave status	ISSTA[2:0] / Status: 0x7 reserved; 0x6 NAK received; 0x5 ACK received; 0x4 ACK/NAK sent; 0x3 End of Rx data; 0x2 End of Tx data; 0x1 Stop detected; 0x0 Start detected	0x0	R	
		D1	ISEIF	Overrun error flag	1 Error / 0 Normal	0	R/W	Reset by writing 1.
		D0	ISIF	Operation completion flag	1 Completed / 0 Not completed	0	R/W	

Note: This register is effective only in I²C slave mode. Configure USIL to I²C slave mode before this register can be used.

D[7:6] Reserved

D5 ISBSY: I²C Slave Busy Flag Bit

Indicates the I²C slave operation status.

1 (R): Busy

0 (R): Standby (default)

Writing 1 to ISTG/USIL_ISTG register (starting an I²C slave operation) sets ISBSY to 1 indicating that the I²C controller is busy (operating). When the specified operation has finished, ISBSY is reset to 0.

D[4:2] ISSTA[2:0]: I²C Slave Status Bits

Indicates the I²C slave status.

Table 21.8.7 I²C Slave Status Bits

ISSTA[2:0]	Status
0x7	Reserved
0x6	NAK has been received.
0x5	ACK has been received.
0x4	ACK or NAK has been sent.
0x3	End of receive data.
0x2	End of transmit data.
0x1	Stop condition has been detected.
0x0	Start condition has been detected.

(Default: 0x0)

When an operation completion interrupt occurs, read ISSTA[2:0] to check the operation that has been finished. ISSTA[2:0] is automatically reset to 0x0 by writing 1 to ISIF.

D1 ISEIF: Overrun Error Flag Bit

Indicates whether an overrun error has occurred or not.

1 (R): Error occurred

0 (R): No error (default)

1 (W): Reset to 0

0 (W): Ignored

ISEIF is set to 1 when an overrun error occurs. At the same time a receive error interrupt request is sent to the ITC if ISEIE/USIL_ISIE register is 1. An overrun error occurs when a transmit or receive trigger is issued after two-byte data has been received (the first byte data exists in the receive data buffer and the second byte data exists in the shift register) without the receive data buffer being read.

ISEIF is reset by writing 1.

To reset an overrun error, clear ISEIF by writing 1, and then read the receive data buffer (USIL_RD register) twice.

D0 ISIF: Operation Completion Flag Bit

Indicates whether the triggered operation has completed or not.

1 (R): Completed
0 (R): Not completed (default)
1 (W): Reset to 0
0 (W): Ignored

ISIF is set to 1 when the operation that is specified and triggered using the USIL_ISTG register has completed. At the same time an operation completion interrupt request is sent to the ITC if ISIE/USIL_ ISIE register is 1. ISIF is reset by writing 1.

USIL LCD SPI Mode Configuration Register (USIL_LSCFG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL LCD SPI Mode Configuration Register (USIL_LSCFG)	0x300680 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	LSCPHA	Clock phase select	1	Phase 1	0	Phase 0	0	R/W	
		D2	LSCPOL	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	LSCMD	Command bit	1	High	0	Low	0	R/W	
		D0	LSCMDEN	Command bit enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in LCD SPI mode. Configure USIL to LCD SPI mode before setting this register.

D[7:4] Reserved

D3 LSCPHA: Clock Phase Select Bit

Selects the LCD SPI clock phase.

1 (R/W): Phase 1
0 (R/W): Phase 0 (default)

Set the data transfer timing together with LSCPOL. (See Figure 21.8.2.)

D2 LSCPOL: Clock Polarity Select Bit

Selects the LCD SPI clock polarity.

1 (R/W): Active low
0 (R/W): Active high (default)

Set the data transfer timing together with LSCPHA. (See Figure 21.8.2.)

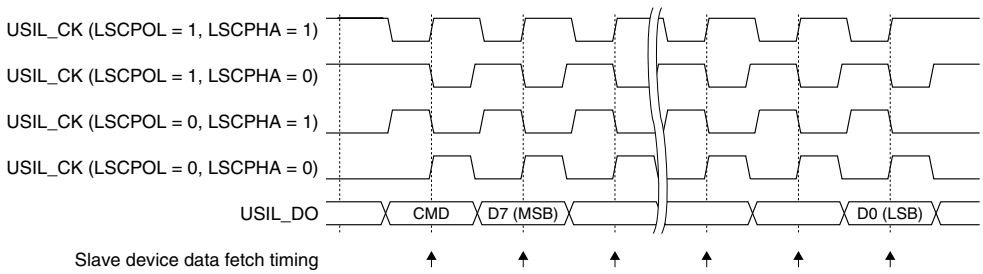


Figure 21.8.2 Clock and Data Transfer Timing

D1 LSCMD: Command Bit

Sets the command bit value (see LSCMDEN below).

1 (R/W): High
0 (R/W): Low (default)

D0 LSCMDEN: Command Bit Enable Bit

Enables sending the command bit on the data line (USIL_DO).

1 (R/W): Enabled
0 (R/W): Disabled (default)

When LSCMDEN is set to 1, data is prefixed with a command bit (1 bit). The command bit is used for controlling the SPI LCD driver/panel connected to the USIL. The command bit value to be transmitted can be specified using LSCMD.

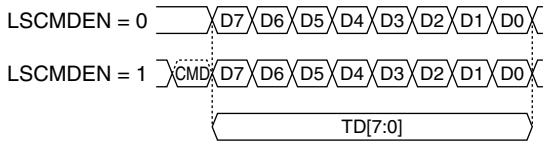


Figure 21.8.3 Data Configuration (8-bit data mode)

When LSCMDEN is set to 0, the command bit selected using LSCMD is output from the USIL_DI (lcds_a0) pin.

USIL LCD SPI Mode Interrupt Enable Register (USIL_LSIE)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL LCD SPI Mode Interrupt Enable Register (USIL_LSIE)	**0x300681** (8 bits)	D7–1	**–**	reserved	–				–	–	0 when being read.
		D0	**LSTDIE**	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in LCD SPI mode. Configure USIL to LCD SPI mode before setting this register.

D[7:1] Reserved

D0 LSTDIE: Transmit Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC when data written to the transmit data buffer is sent to the shift register (i.e. when data transmission begins).
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the transmit data buffer using interrupts.

USIL LCD SPI Mode Interrupt Flag Register (USIL_LSIF)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL LCD SPI Mode Interrupt Flag Register (USIL_LSIF)	**0x300682** (8 bits)	D7–2	**–**	reserved	–				–	–	X when being read.
		D1	**LSBSY**	Transfer busy flag	1	Busy	0	Idle	0	R	
		D0	**LSTDIF**	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	Reset by writing 1.

Note: This register is effective only in LCD SPI mode. Configure USIL to LCD SPI mode before setting this register.

D[7:2] Reserved

D1 LSBSY: Transfer Busy Flag Bit

Indicates the LCD SPI transfer status.
1 (R): Busy
0 (R): Idle (default)

LSBSY is set to 1 when the LCD SPI starts data transfer and is maintained at 1 while transfer is underway. It is cleared to 0 after data transfer for the data size set using LSDMOD[1:0]/USIL_LSDCFG register has completed.

D0 LSTDIF: Transmit Buffer Empty Flag Bit

Indicates the transmit data buffer status.
1 (R): Empty (default)
0 (R): Data exists
1 (W): Reset to 0
0 (W): Ignored

LSTDIF is set to 1 when the transmit data written to the transmit data buffer is transferred to the shift register (when transmission starts), indicating that the next transmit data can be written to. At the same time a transmit buffer empty interrupt request is sent to the ITC if LSTDIE/USIL_LSIE register is 1. LSTDIF is reset by writing 1.

USIL LCD SPI Mode Data Configuration Register (USIL_LSDCFG)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
USIL LCD SPI Mode Data Configuration Register (USIL_ LSDCFG)	0x30068f (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3–2	LS18DFM [1:0]	LCD SPI 18-bit data format select	LS18DFM[1:0]	Data format	0x0	R/W	
					0x3	Format 3			
					0x2	Format 2			
					0x1	Format 1			
					0x0	Format 0			
		D1–0	LSDMOD [1:0]	LCD SPI data mode select	LSDMOD[1:0]	Data mode	0x0	R/W	
					0x3	24-bit mode			
					0x2	18-bit mode			
					0x1	16-bit mode			
					0x0	8-bit mode			

Note: This register is effective only in LCD SPI mode. Configure USIL to LCD SPI mode before setting this register.

D[7:4] Reserved

D[3:2] LS18DFM[1:0]: LCD SPI 18-bit Data Format Select bits

Selects a data format in 18-bit mode. (See Figure 21.8.6.)

Table 21.8.8 LCD SPI 18-bit Data Format

LS18DFM[1:0]	Data format
0x3	Format 3
0x2	Format 2
0x1	Format 1
0x0	Format 0

(Default: 0x0)

D[1:0] LSDMOD[1:0]: LCD SPI Data Mode Select Bit

Selects the LCD SPI data mode.

Table 21.8.9 LCD SPI Data Mode

LSDMOD[1:0]	Data mode
0x3	24-bit mode
0x2	18-bit mode
0x1	16-bit mode
0x0	8-bit mode

(Default: 0x0)

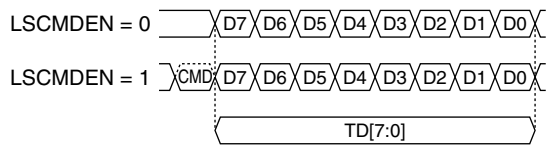


Figure 21.8.4 8-bit Data Format

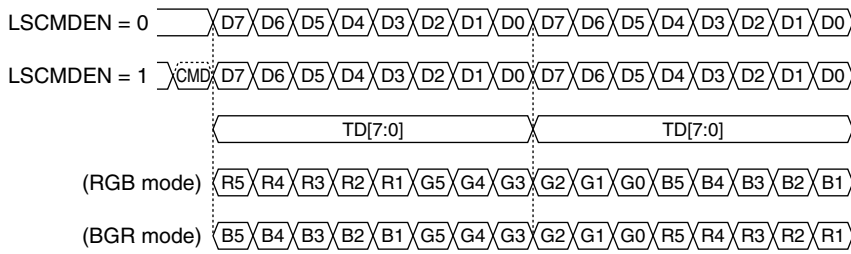
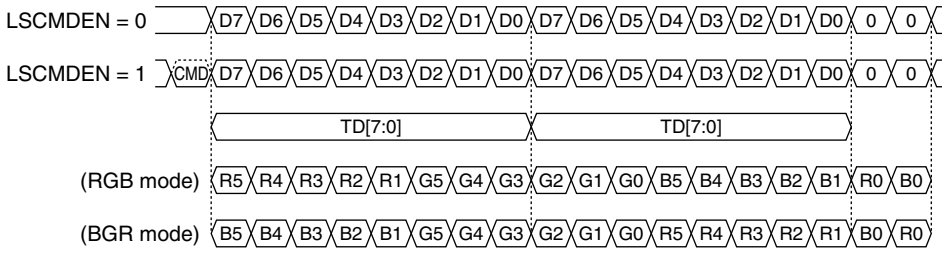
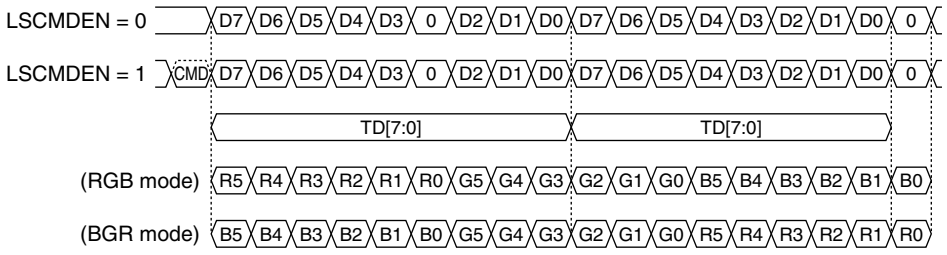


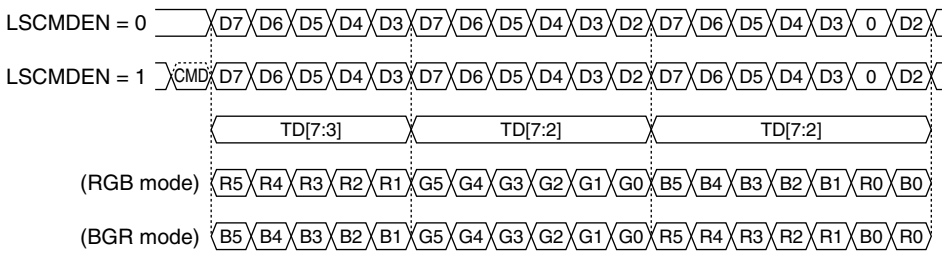
Figure 21.8.5 16-bit Data Format



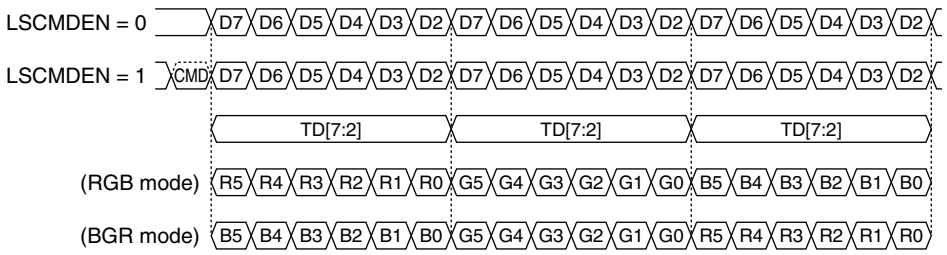
18-bit Format 0



18-bit Format 1



18-bit Format 2



18-bit Format 3

Figure 21.8.6 18-bit Data Format

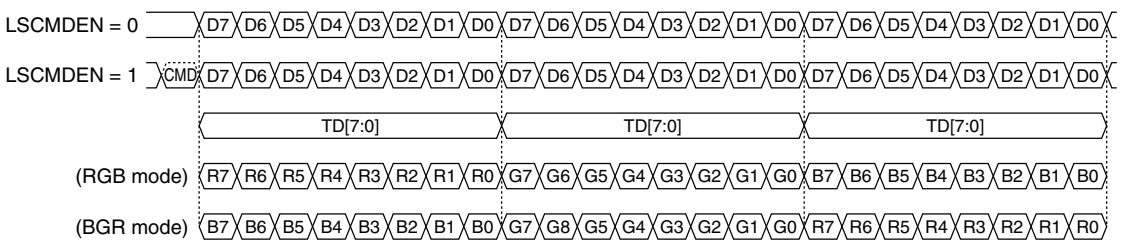


Figure 21.8.7 24-bit Data Format

USIL LCD Parallel I/F Mode Configuration Register (USIL_LPCFG)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL LCD Parallel I/F Mode Configuration Register (USIL_LPCFG)	0x300690 (8 bits)	D7–3	–	reserved	–				–	–	0 when being read.
		D2	LPSRDEN	Successive read enable	1	Enable	0	Disable	0	R/W	
		D1	LPCMD	Command bit	1	High	0	Low	0	R/W	
		D0	LPRD	Read trigger	1	Trigger	0	Ignored	0	W	
					1	Read cycle	0	Read finished		R	

Note: This register is effective only in LCD parallel mode. Configure USIL to LCD parallel mode before setting this register.

D[7:3] Reserved

D2 LPSRDEN: Successive Read Enable Bit

Enables the successive read function.
1 (R/W): Enabled
0 (R/W): Disabled (default)

By issuing a read trigger (writing 1 to LPRD) after setting LPSRDEN to 1, the LCD parallel interface repeats data reading from the LCD driver/panel while LPSRDEN is 1. When LPSRDEN is set to 0, the LCD parallel interface stops data reading after the read cycle being currently executed has finished.

D1 LPCMD: Command bit

Sets the command bit value.
1 (R/W): High
0 (R/W): Low (default)

The command bit selected using LPCMD is output from the USIL_DI (lcdp_a0) pin.

D0 LPRD: Read Trigger Bit

Starts a read cycle of the LCD parallel interface.
1 (W): Trigger (start reading)
0 (W): Ignored
1 (R): During reading
0 (R): Read cycle has finished

To read data from the LCD driver/panel via the LCD parallel interface, issue a read trigger by writing 1 to LPRD. Set the command bit (LPCMD) value before writing to LPRD. The command bit value set is output from the USIL_DI pin immediately after it is written to the register. Then it loads the USIL_LCD_D[7:0] pin status to the read (receive data) buffer (RD[7:0]/USIL_RD register). LPRD retains 1 until the read data is loaded to the read buffer.

USIL LCD Parallel I/F Mode Interrupt Enable Register (USIL_LPIE)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL LCD Parallel I/F Mode Interrupt Enable Register (USIL_LPIE)	0x300691 (8 bits)	D7–2	–	reserved	–				–	–	0 when being read.
		D1	LPRDIE	Read buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	LPWRIE	Write buffer empty interrupt enable	1	Enable	0	Disable	0	R/W	

Note: This register is effective only in LCD parallel mode. Configure USIL to LCD parallel mode before setting this register.

D[7:2] Reserved

D1 LPRDIE: Read Buffer Full Interrupt Enable Bit

Enables interrupt requests to the ITC when data is loaded to the read (receive data) buffer.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to read data using interrupts.

D0 LPWRIE: Write Buffer Empty Interrupt Enable Bit

Enables interrupt requests to the ITC when data written to the write (transmit data) buffer is output via the USIL_LCD_D[7:0] pins.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Set this bit to 1 to write data to the write buffer using interrupts.

USIL LCD Parallel I/F Mode Interrupt Flag Register (USIL_LPIF)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL LCD Parallel I/F Mode Interrupt Flag Register (USIL_LPIF)	0x300692 (8 bits)	D7–3	–	reserved	–				–	–	X when being read.
		D2	LPBSY	Transfer busy flag	1	Busy	0	Idle	0	R	
		D1	LPRDIF	Read buffer full flag	1	Full	0	Not full	0	R/W	Reset by writing 1.
		D0	LPWRIF	Write buffer empty flag	1	Empty	0	Not empty	0	R/W	

Note: This register is effective only in LCD parallel mode. Configure USIL to LCD parallel mode before setting this register.

D[7:3] Reserved

D2 LPBSY: Transfer Busy Flag Bit

Indicates the LCD parallel interface status.

1 (R): Busy
0 (R): Idle (default)

LPBSY is set to 1 when the LCD parallel interface starts data transfer and is maintained at 1 while transfer is underway. It is cleared to 0 once the transfer is completed.

D1 LPRDIF: Read Buffer Full Flag Bit

Indicates the read (receive data) buffer status.

1 (R): Data full
0 (R): No data (default)
1 (W): Reset to 0
0 (W): Ignored

LPRDIF is set to 1 when data received is loaded to the read buffer (when receiving is completed), indicating that the data can be read. At the same time a read buffer full interrupt request is sent to the ITC if LPRDIE/USIL_LPIE register is 1. LPRDIF is reset by writing 1.

D0 LPWRIF: Write Buffer Empty Flag Bit

Indicates the write (transmit data) buffer status.

1 (R): Empty (default)
0 (R): Data exists
1 (W): Reset to 0
0 (W): Ignored

LPWRIF is set to 1 when the data written to the write buffer is output via the USIL_LCD_D[7:0] pins, indicating that the next data can be written to. At the same time a write buffer empty interrupt request is sent to the ITC if LPWRIE/USIL_LPIE register is 1. LPWRIF is reset by writing 1.

USIL LCD Parallel I/F Mode Access Timing Register (USIL_LPAC)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
USIL LCD Parallel I/F Mode Access Timing Register (USIL_LPAC)	0x30069f (8 bits)	D7–6	LPHD[1:0]	Hold cycle	LPHD[1:0]	Hold cycle	0x0	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D5–4	LPST[1:0]	Setup cycle	LPST[1:0]	Setup cycle	0x0	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D3–0	LPWT[3:0]	Wait cycle	LPWT[3:0]	Wait cycle	0x0	R/W	
					0xf	15 cycles			
					0xe	14 cycles			
					:	:			
					0x1	1 cycle			
					0x0	0 cycles			

Note: This register is effective only in LCD parallel mode. Configure USIL to LCD parallel mode before setting this register.

D[7:6] LPHD[1:0]: Hold Cycle Bits

Configures the hold cycle of the LCD parallel interface.

Table 21.8.10 Hold Cycle Settings

LPHD[1:0]	Number of hold cycles
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

D[5:4] LPST[1:0]: Setup Cycle Bits

Configures the setup cycle of the LCD parallel interface.

Table 21.8.11 Setup Cycle Settings

LPST[1:0]	Number of setup cycles
0x3	4 cycles
0x2	3 cycles
0x1	2 cycles
0x0	1 cycle

(Default: 0x0)

D[3:0] LPWT[3:0]: Wait Cycle Bits

Configures the wait cycle of the LCD parallel interface.

Table 21.8.12 Wait Cycle Settings

LPWT[3:0]	Number of wait cycles
0xf	15 cycles
0xe	14 cycles
:	:
0x1	1 cycle
0x0	0 cycles

(Default: 0x0)

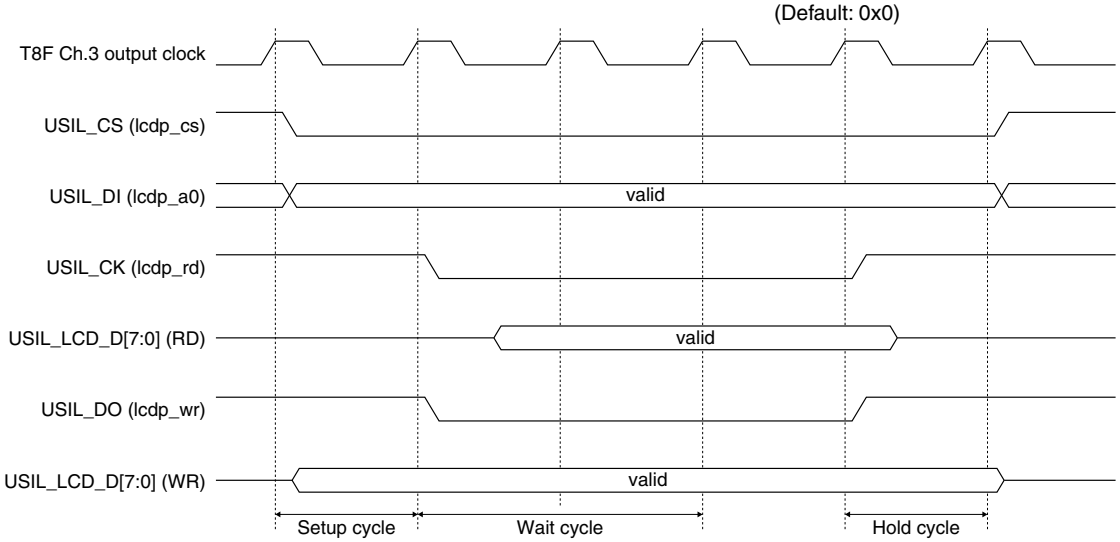


Figure 21.4.8.8 Access Timing Parameters

21.9 Precautions

Interface mode setting

Be sure to perform software reset (USIMOD[2:0]/USIL_GCFG register = 0x0) and set the interface mode (USIMOD[2:0]/USIL_GCFG register = 0x1 to 0x7) before changing other USIL configurations.

Receiving control byte in I²C slave mode

The external I²C master device sends a control byte to the I²C slave device when an ACK has been received after sending a slave address. The subsequent operations of the slave device are determined by the control byte.

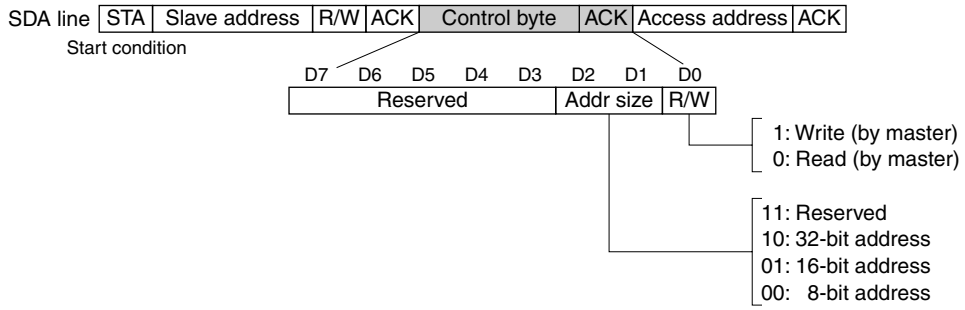


Figure 21.9.1 Control Byte Sent from I²C Master

I²C master write (data receiving from master)

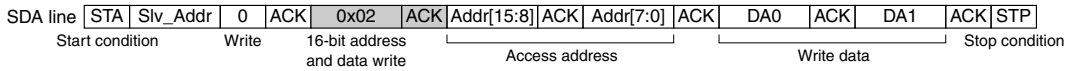


Figure 21.9.2 I²C Master Write (Data Receiving from Master)

The control byte specifies the access address size and writing operations. The received data that follow the control byte should be used as the address and the data to be written according to the access address size.

I²C master read (data transmission to master)

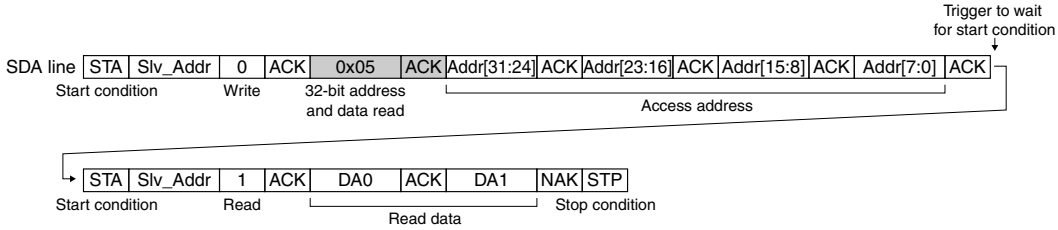


Figure 21.9.3 I²C Master Read (Data Transmission to Master)

The master sends the access address following the control byte. Perform data reception for the control byte and address data to determine the address from which transmit data is read. After sending an ACK for Addr 0, set ISTGMOD[2:0]/USIL_ISTG register to 0x0 and ISTG/USIL_ISTG register to 1 to wait for a start condition that will be sent from the master for reading data (for the slave to sent the read data).

22 I²S

22.1 I²S Module Overview

The S1C33L27 includes a bidirectional 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.
- Either 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 and 24-bit resolution for PCM data to be input.
- A 24-byte receive FIFO (24 bits × 2 channels × 4) is included.
- FIFO data 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 22.1.1 shows the I²S module configuration.

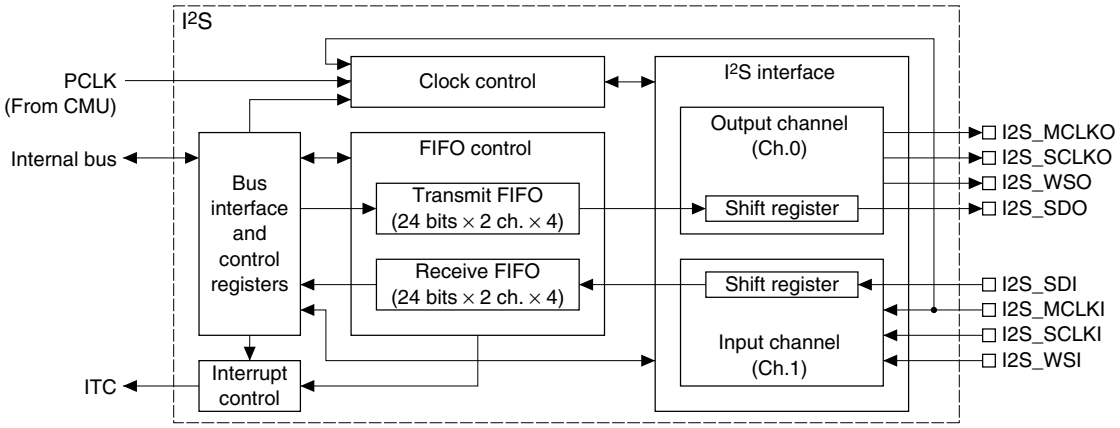


Figure 22.1.1 I²S Module Configuration

22.2 I²S Input/Output Pins

Table 22.2.1 lists the I²S input/output pins.

Table 22.2.1 List of I²S Pins

Pin name	I/O	Qty	Function
I2S_SDO (Ch.0)	O	1	I²S data output pin Outputs serial PCM data.
I2S_WSO (Ch.0)	O	1	I²S word-select signal (LRCLK) output pin Outputs the word-select signal that indicates the channel (L or R) of the data being output.
I2S_SCLKO (Ch.0)	O	1	I²S synchronous clock (bit clock) output pin Outputs the synchronous clock (bit clock) for serial data transfer.
I2S_MCLKO (Ch.0)	O	1	I²S master clock output pin Outputs the I²S master clock.
I2S_SDI (Ch.1)	I	1	I²S data input pin Inputs serial PCM data.
I2S_WSI (Ch.1)	I	1	I²S word-select signal (LRCLK) input pin Inputs the word-select signal that indicates the channel (L or R) of the data being input.
I2S_SCLKI (Ch.1)	I	1	I²S synchronous clock (bit clock) input pin Inputs the synchronous clock (bit clock) for serial data transfer.
I2S_MCLKI (Ch.1)	I	1	I²S master clock input pin Inputs the I²S master clock.

The I²S input/output pins (I2S_SDO/I, I2S_WSO/I, I2S_SCLKO/I, I2S_MCLKO/I) are shared with I/O ports and are initially set as general-purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as I²S input/output pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

22.3 I²S Operating Clock

The I²S module operates with PCLK (= system clock) supplied from the CMU. A CMU register is used for clock supply control (default: enabled). For more information on the clock control, see the "Clock Management Unit (CMU)" chapter. In SLEEP mode, the CMU stops supplying PCLK to the I²S module.

22.4 I²S Module Settings

Before starting data transfer via the I²S bus, the following settings are required.
1. Setting the I/O pins and pin mode
2. Setting the I²S interface clocks
3. Setting the output data format and timing
4. Setting interrupts and DMA (See Section 22.7.)

The following explains the content of each setting.

Note: Before making these settings, be sure to set the I²S module to inactive status (by setting I2SSTART0/I2S_START register and I2SSTART1/I2S_START register to 0). Changing a setting while the I²S module is active may cause a malfunction.

Setting the I/O pins and pin mode

Select the I/O pins used by the I²S module using the port function select bits. For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

In order to input/output from/to external I²S slave devices (5-pin DAC) using five I²S pins, the I²S module provides two different pin modes (5-pin mode and 8-pin mode). I2SPIN/I2S_MODE register is used to select this mode. When I2SPIN is set to 0 (default), the I²S module is set to 8-pin mode; when I2SPIN is set to 1, the I²S module is set to 5-pin mode.

8-pin mode

All the I2S_MCLKO, I2S_SCLKO, I2S_WSO, I2S_SDO, I2S_MCLKI, I2S_SCLKI, I2S_WSI, and I2S_SDI pins are available in this mode. Ch.0 (master output device) and Ch.1 (slave input device) can be used independently. The clock pins of Ch.0 and Ch.1 are not connected inside the IC.

5-pin mode

In this mode, only the I2S_MCLKO, I2S_SCLKO, I2S_WSO, I2S_SDO, and I2S_SDI pins are used for data transfer from/to external I²S slave devices. The Ch.0 clock pins are connected to those of Ch.1 inside the IC and the Ch.0 output clocks are directly fed to Ch.1. This enables the Ch.1 clock pins to be used for other modules when a 5-pin device (e.g., 5-pin DAC) is connected.

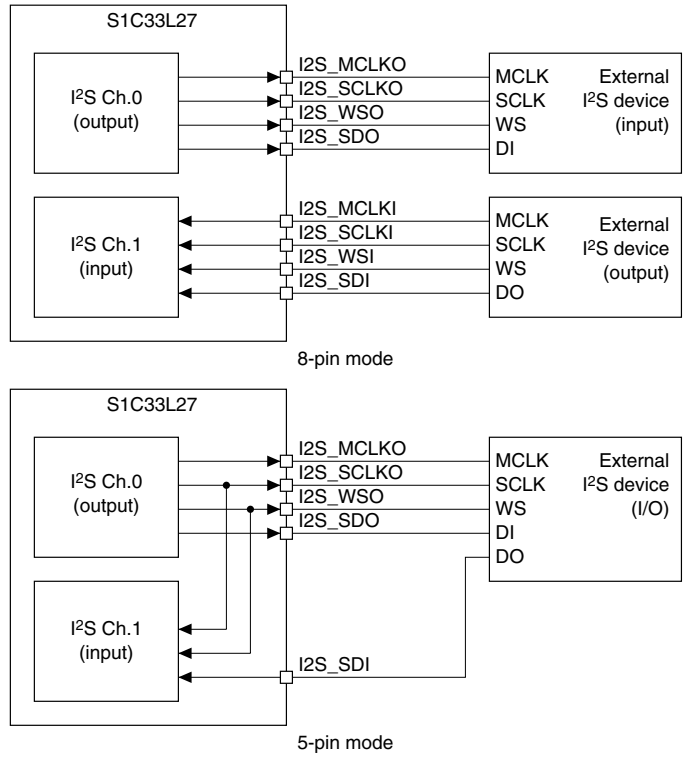


Figure 22.4.1 Pin Mode

Setting the I²S interface clocks

The I²S module inputs/outputs the following three clocks.

1. I2S_MCLK (master clock)
2. I2S_SCLK (bit clock)
3. I2S_WS (word-select clock)

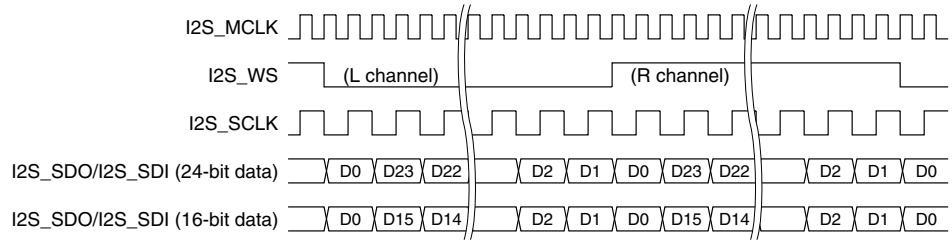


Figure 22.4.2 I²S Interface Clocks

The clock conditions that can be set and their control bits are explained below. For details on the clock settings, see Section 22.9, " I²S Clock Settings."

Source clock for I2S_MCLK (master clock)

Either the internal clock (PCLK) or an external clock input from the I2S_MCLKI pin for I²S Ch.1 can be selected as the source clock for the master clock (I2S_MCLK) using MCLKSEL/I2S_DV_MCLK register.
When MCLKSEL is set to 0 (default), the I²S module generates the master clock (I2S_MCLK) by dividing PCLK and outputs it from the I2S_MCLKO pin for I²S Ch.0. When MCLKSEL is set to 1, the clock input from the I2S_MCLKI pin is sent to the clock generator in the I²S module to generate the bit clock and word-select clock.

I2S_MCLK (master clock) division ratio

When the internal clock is selected as the I2S_MCLK source clock, the I²S module generates the I2S_MCLK to be output from the I2S_MCLKO pin by dividing the PCLK (system clock). Specify this division ratio using MCLKDIV[5:0]/I2S_DV_MCLK register.

Table 22.4.1 I2S_MCLK (PCLK Division Ratio) Settings

MCLKDIV[5:0]	I2S_MCLK
0x3f	1/64
0x3e	1/63
0x3d	1/62
:	:
0x2	1/3
0x1	1/2
0x0	Reserved

(Default: 0x0)

I2S_SCLK (bit clock) division ratio

The I²S module generates the bit clock to be output from the I2S_SCLKO pin for I²S Ch.0 by dividing the source clock selected for generating I2S_MCLK. Specify this division ratio using BCLKDIV0[7:0]/I2S_DV_BCLK register.

Table 22.4.2 Bit Clock (SRC_CLK Division Ratio) Settings

BCLKDIV0[7:0]	Bit clock (I2S_SCLKO)
0xff	1/512
0xfe	1/510
0xfd	1/508
:	:
0x2	1/6
0x1	1/4
0x0	1/2

(SRC_CLK = PCLK or I2S_MCLKI input clock, default: 0x0)

The following equation can be used to calculate the I²S Ch.0 bit clock frequency.

$$f_{I2S\_SCLKO} = \frac{f_{SRC\_CLK}}{(BCLKDIV + 1) \times 2} \text{ [Hz]}$$

f_{I2S_SCLKO}: I²S Ch.0 bit clock frequency [Hz]
f_{SRC_CLK}: PCLK or I2S_MCLKI input clock frequency [Hz]
BCLKDIV: BCLKDIV0[7:0] set value (0x0 to 0xff)

Since I²S Ch.1 uses the bit clock input from the I2S_SCLKI pin, the setting above will not be applied to Ch.1.

Sampling clock (I2S_WS) cycle

I²S Ch.0 generates the sampling clock (word-select clock) to be output from the I2S_WSO pin by counting the source clock set using BCLKDIV0[7:0]. Specify the number of bit clock cycles corresponding to half the I2S_WS clock cycle (high or low level period) using WSCLKCYC0[4:0]/I2S_DV_BCLK register.

I²S Ch.1 inputs the sampling clock (word-select clock) from the I2S_WSI pin. In the same way as for Ch.0, set the number of bit clock cycles using WSCLKCYC1[4:0]/I2S_DV_BCLK register to specify the clock cycle.

Table 22.4.3 Sampling Clock Cycle Settings

WSCLKCYC*x*[4:0]	Sampling clock period (number of bit clock cycles)
0x1f–0x11	Reserved
0x10	32 clocks
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 following equation can be used to calculate the sampling clock frequency.

$$fs = \frac{f_{I2S\_SCLK}}{n \times 2} \text{ [Hz]}$$

fs: Sampling clock frequency [Hz]
f_{I2S_SCLK}: Bit clock frequency [Hz] (Ch.0: See Table 22.4.2. Ch.1: I2S_SCLKI input clock frequency)
n: Number of bit clock cycles selected using WSCLKCYC*x*[4:0] (See Table 22.4.3.)

Note: The value set at WSCLKCYC*x*[4:0] is the number of bit clock cycles for adjusting the sampling clock cycle, not the number of audio data bits. The set value must be equal to or larger than the number of audio data bits (24 or 16 bits).

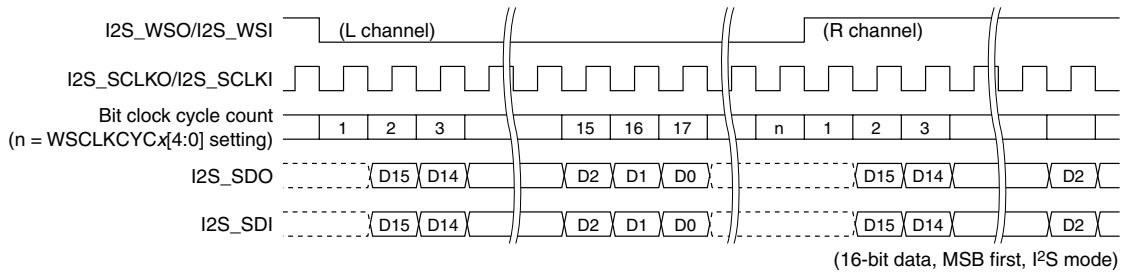


Figure 22.4.3 Sampling Clock Cycle

Selecting the word clock mode

The I2S_WS signal indicates by its level (high or low) whether the current output is left-channel or right-channel data. Set this correspondence between the levels and channels using WCLKMD0/I2S_CTL0 register for Ch.0, and WCLKMD1/I2S_CTL1 register for Ch.1.

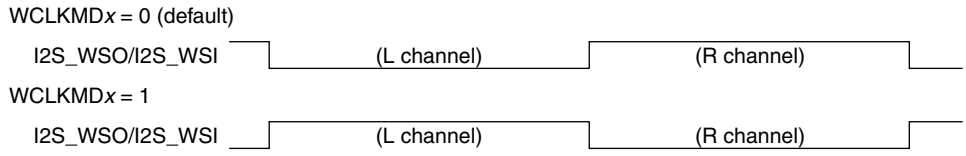


Figure 22.4.4 Word Clock Mode Selections

I2S_SCLK (bit clock) polarity

Select the bit clock polarity for Ch.0 using BCLKPOL0/I2S_CTL0 register and using BCLKPOL1/I2S_CTL1 register for Ch.1.

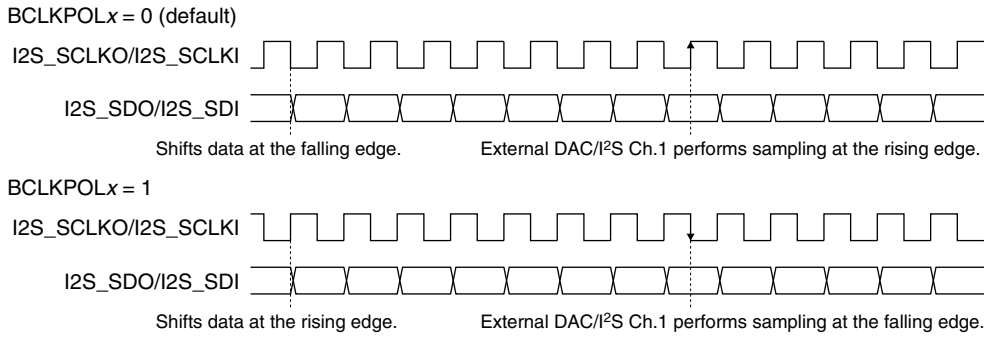


Figure 22.4.5 Bit Clock Polarity Selections

Setting the data format and timing

Input/output data resolution

Set the output data resolution to either 16 bits or 24 bits using DATRES0/I2S_CTL0 register, and the input data resolution using DATRES1/I2S_CTL1 register. When DATRES*x* is set to 0 (default), 16-bit resolution is selected; when it is set to 1, 24-bit resolution is selected.

Output data format (MSB first/LSB first)

Select whether to output the MSB or LSB first using DTFORM0/I2S_CTL0 register. When DTFORM0 is set to 0 (default), the MSB is output first; when it is set to 1, the LSB is output first.

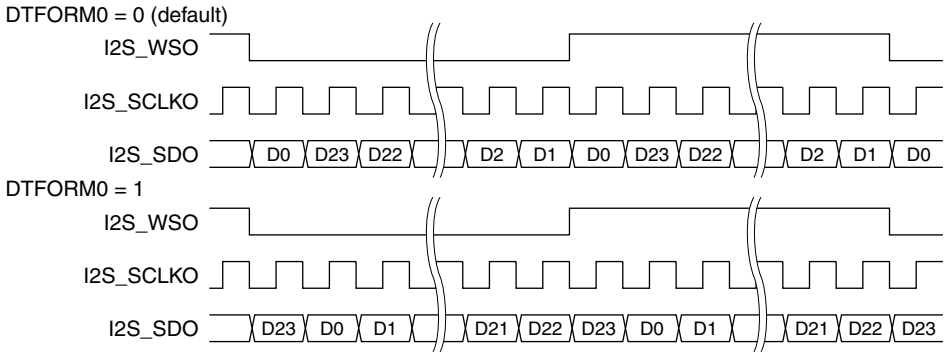


Figure 22.4.6 Output Data Format Selections

Signed/unsigned output data formats

When the right-justified mode has been selected as a data output timing condition, select whether to output the data as signed data or as unsigned data using DTSIGN0/I2S_CTL0 register.

Setting DTSIGN0 to 0 (default) selects the unsigned format. The higher-order bits other than the significant bits are set to 0. Setting DTSIGN0 to 1 selects the signed format. The higher-order bits other than the significant bits are set to signed bits (D23 or D15).

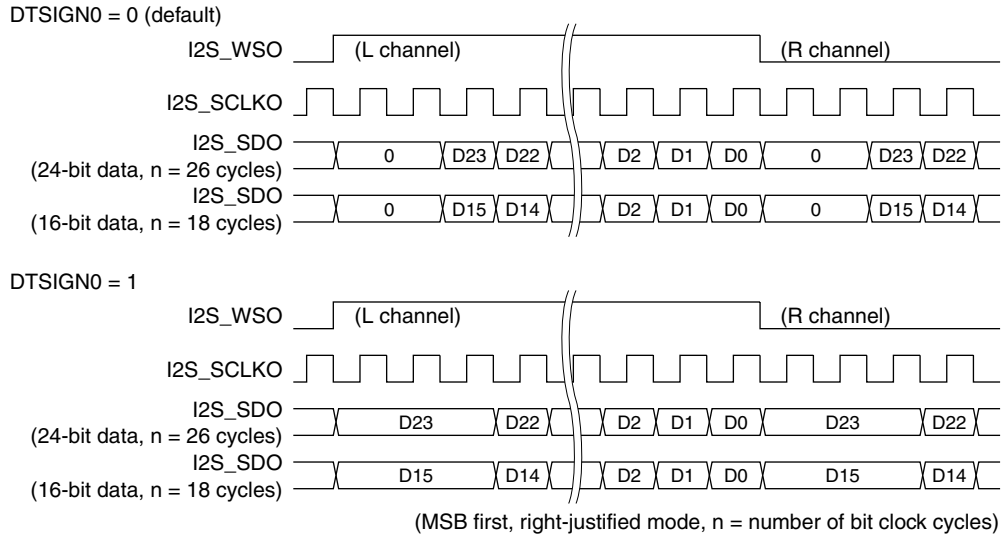


Figure 22.4.7 Unsigned and Signed Formats

This setting is only enabled in the right-justified mode. Regardless of the DTSIGN0 setting, unsigned data is output in all the other modes.

Data input/output timing

Select the data output timing for Ch.0 using DTTMG0[1:0]/I2S_CTL0 register and the data input timing for Ch.1 using DTTMG1[1:0]/I2S_CTL1 register.

Table 22.4.4 Data Input/Output Timing

DTTMG*x*[1:0]	Data input/output timing mode
0x3	Reserved
0x2	Right-justified mode
0x1	Left-justified mode
0x0	I²S mode

(Default: 0x0)

Setting DTTMG*x*[1:0] to 0x0 (default) selects I²S mode. In this mode, the leading bit of each data is input/output after a delay of one I2S_SCLK clock from the I2S_WS signal edge.

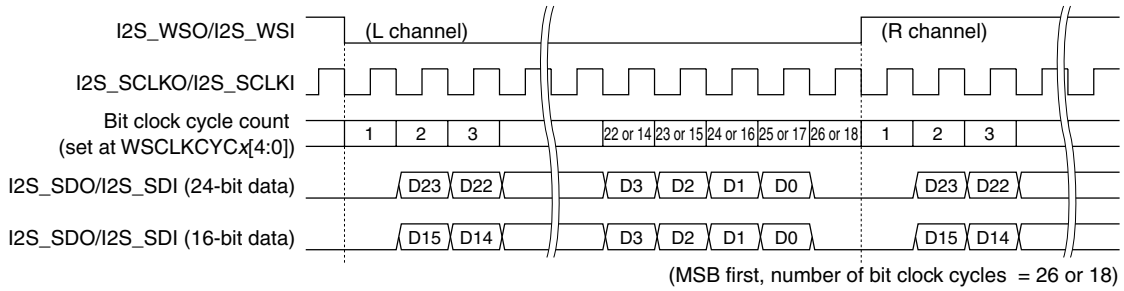


Figure 22.4.8 Data Input/Output Timing 1 (I²S Mode)

Setting DTTMG*x*[1:0] to 0x1 selects left-justified mode. In this mode, input/output of each data starts from the I2S_WS signal edge.

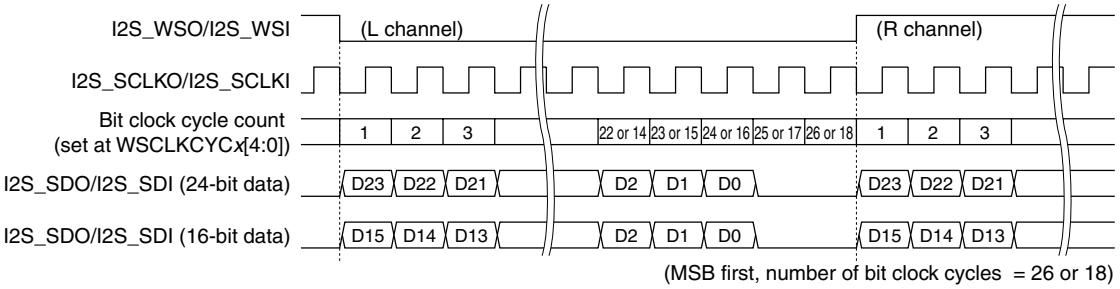


Figure 22.4.9 Data Input/Output Timing 2 (Left-justified Mode)

Setting DTTMG*x*[1:0] to 0x2 selects right-justified mode. In this mode, each data is input/output so that it is right-justified with the I2S_WS signal edge.

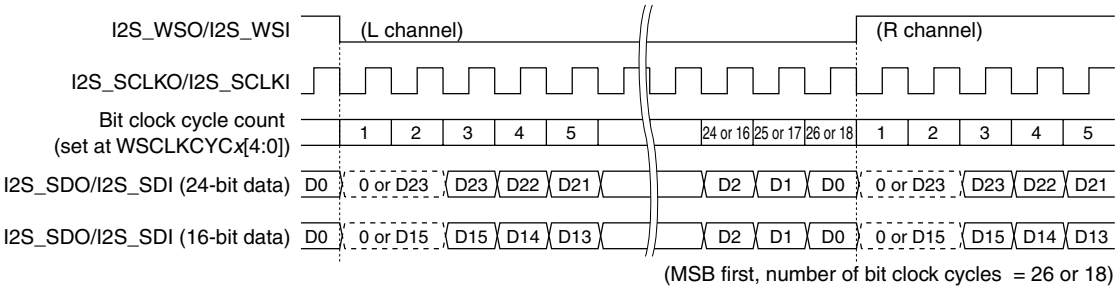


Figure 22.4.10 Data Input/Output Timing 3 (Right-justified Mode)

Note: When using I²S in right-justified mode, the number of bit clock cycles (sampling clock cycle) must be set to a value that is equal to or higher than [data bit size + 2].

22.5 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 or DMA conditions as described in Section 22.7.
3. Write 1 to I2SEN0/I2S_CTL0 register to activate the I²S Ch.0 circuit. The I²S Ch.0 initiates division of the source clock.
4. Set the output channel mode using CHMD0[1:0]/I2S_CTL0 register.

Table 22.5.1 Output Channel Mode Selection

CHMD0[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 1 to I2SOUTEN0/I2S_CTL0 register to enable I²S output. When I2SOUTEN0 is set to 1, all the output pins enter standby status.
   Write 0 to I2SOUTEN0 to disable I²S output. When I2SOUTEN0 is set to 0, the I2S_MCLKO pin still keeps standby status, the I2S_WSO pin is fixed at 0. The I2S_SDO pin is left unchanged. The I2S_SCLKO pin is fixed at 0 (when BCLKPOL0/I2S_CTL0 register = 0) or 1 (when BCLKPOL0 = 1).
   I2SOUTEN0 can be altered even if data is being output. This enables the I²S module to pause or resume shifting data (data output).

6. Write 1 to I2SSTART0/I2S_START register to start output.
   When I2SSTART0 is set to 1, the I²S module issues an interrupt or DMA request. However, the I²S module idles until the FIFO becomes full (24 or 16 bits × 2 channels (L & R) × 4), then loads data (L & R) in the FIFO to the shift register and starts serial output in sync with the I2S_WSO signal.
   The data in the shift register is shifted at the I2S_SCLKO 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 conditions has been read out from the FIFO, an interrupt or DMA request is generated.
   Write data using this interrupt or DMA as described in Step 7.

7. Write audio data to the FIFO.
   There are two ways to write data.

   7.1) Using interrupts
   I²S Ch.0 supports three different types of interrupts.

   • I²S Ch.0 FIFO half empty interrupt
   When the half empty interrupt is enabled, the I²S module generates an interrupt after two stereo data have been read out from the FIFO. However, the I²S module continues shifting out the remaining buffer data until the FIFO becomes completely empty.
   Write two stereo data (24 or 16 bits × 2 channels (L & R) × 2) to the FIFO at once in the half empty interrupt handler.

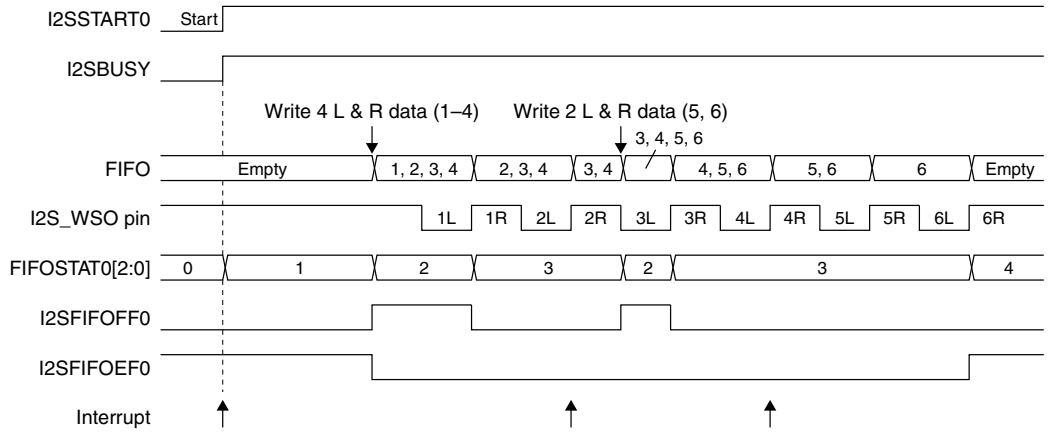
Note: If the handler fills the FIFO with more than two groups of data, it may overwrite the remaining data in the FIFO. If the handler fills the FIFO with one group of data, a half empty interrupt will be issued again after one remaining data is sent to the shift register. Therefore, the FIFO should be filled with two groups of data in the half empty interrupt handler.

   • I²S Ch.0 FIFO whole empty interrupt
   When the whole empty interrupt is enabled, the I²S module generates an interrupt after all data (four stereo data) have been read out from the FIFO. In this case, the FIFO becomes completely empty. If the I²S module sends the shift register data completely and the FIFO is still empty, it stops shifting out FIFO data until the FIFO becomes full again.
   Be sure to write four stereo data (24 or 16 bits × 2 channels (L & R) × 4) to the FIFO at once in the whole empty interrupt handler, otherwise the I²S module continues idle status.

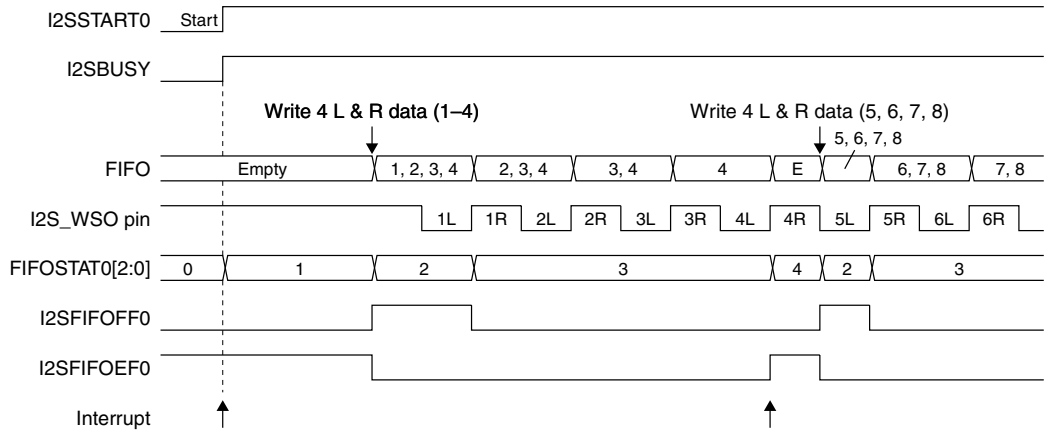
   • I²S Ch.0 FIFO one empty interrupt
   When the one empty interrupt is enabled, the I²S module generates an interrupt after one stereo data has been read out from the FIFO. However, the I²S module continues shifting out the remaining buffer data until the FIFO becomes completely empty.
   Write one group of stereo data (24 or 16 bits × 2 channels (L & R) × 1) to the FIFO at once in the one empty interrupt handler.

Note: If the handler fills the FIFO with more than one group of data, it may overwrite the remaining data in the FIFO. Therefore, the FIFO should always be filled with one group of data in the one empty interrupt handler.

Half empty interrupt



Whole empty interrupt



One empty interrupt

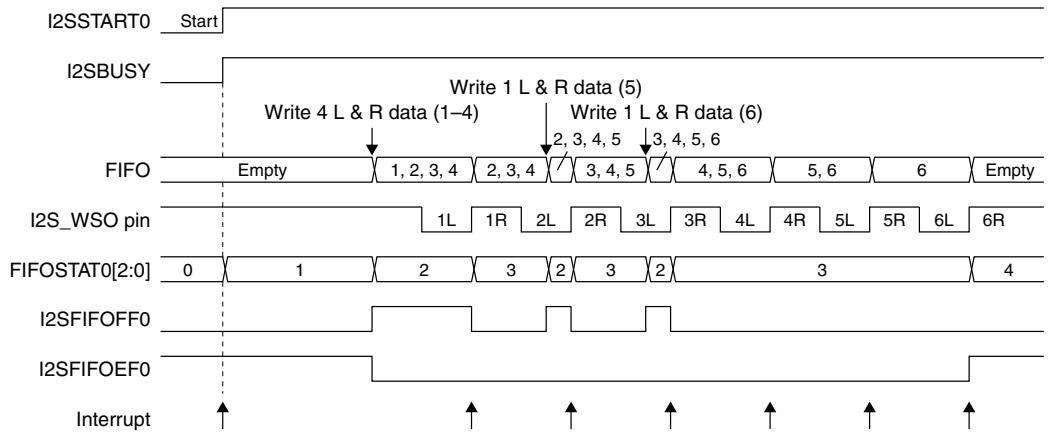


Figure 22.5.1 Transmit FIFO Data and Interrupts

For more information on the I²S interrupt, see Section 22.7.1.

The 32-bit register I2S_FIFO0 (address 0x302420) 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 when it is completely empty.

When writing 16-bit data, use a 16-bit or 32-bit memory write instruction. Note that 8-bit memory write instructions cannot be used. When a 16-bit memory write instruction is used, first write 16-bit L-channel data to address 0x302420, and then 16-bit R-channel data to address 0x302422. When a 32-bit memory write instruction is used, write both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) to address 0x302420.
When writing 24-bit data, use a 32-bit memory write instruction. Note that 8-bit and 16-bit memory write instructions cannot be used. First write 24-bit L-channel data to address 0x302420, and then 24-bit R-channel data to address 0x302420.
Both channel data must be written as a pair even if "mono" is selected as the output channel mode.
The sequence for writing four groups of stereo data (24 or 16 bits × 2 channels (L & R) × 4) to the FIFO is shown in Tables 22.5.2 to 22.5.4 (the instructions to write the data to the CPU register and the procedures have been omitted).

Table 22.5.2 Writing 16-bit Data to the FIFO using 16-bit Memory Write Instructions

No.	Memory write instruction	Write details
1	ld.h [0x302420], %rs1	Data1-L[15:0] → 0x302420
2	ld.h [0x302422], %rs2	Data1-R[15:0] → 0x302422
3	ld.h [0x302420], %rs3	Data2-L[15:0] → 0x302420
4	ld.h [0x302422], %rs4	Data2-R[15:0] → 0x302422
5	ld.h [0x302420], %rs5	Data3-L[15:0] → 0x302420
6	ld.h [0x302422], %rs6	Data3-R[15:0] → 0x302422
7	ld.h [0x302420], %rs7	Data4-L[15:0] → 0x302420
8	ld.h [0x302422], %rs8	Data4-R[15:0] → 0x302422

Table 22.5.3 Writing 16-bit Data to the FIFO using 32-bit Write Instructions

No.	Memory write instruction	Write details
1	ld.w [0x302420], %rs12	{Data1-R[15:0], L[15:0]} → 0x302420
2	ld.w [0x302420], %rs34	{Data2-R[15:0], L[15:0]} → 0x302420
3	ld.w [0x302420], %rs56	{Data3-R[15:0], L[15:0]} → 0x302420
4	ld.w [0x302420], %rs78	{Data4-R[15:0], L[15:0]} → 0x302420

(R = D[31:16], L = D[15:0])

Table 22.5.4 Writing 24-bit Data to the FIFO Using 32-bit Write Instructions

No.	Memory write instruction	Write details
1	ld.w [0x302420], %rs1	Data1-L[23:0] → 0x302420
2	ld.w [0x302420], %rs2	Data1-R[23:0] → 0x302420
3	ld.w [0x302420], %rs3	Data2-L[23:0] → 0x302420
4	ld.w [0x302420], %rs4	Data2-R[23:0] → 0x302420
5	ld.w [0x302420], %rs5	Data3-L[23:0] → 0x302420
6	ld.w [0x302420], %rs6	Data3-R[23:0] → 0x302420
7	ld.w [0x302420], %rs7	Data4-L[23:0] → 0x302420
8	ld.w [0x302420], %rs8	Data4-R[23:0] → 0x302420

7.2) Using DMA transfer

The I²S Ch.0 supports two different types of DMA mode.
DMA transfers are invoked by the cause of one empty interrupt regardless of the DMA mode set. The interrupt signal is output to both the ITC and DMAC. Therefore, DMA transfer can be performed without generating any I²S interrupt.

- Single channel DMA mode

If L-channel and R-channel audio data are sequentially stored in one memory area, use single channel DMA mode. I²S Ch.0 issues two DMA requests simultaneously to DMAC Ch.0 and Ch.1, so either DMAC Ch.0 or Ch.1 can be used for data transfer.
Perform 32-bit data transfer to write both L and R data to the FIFO (address 0x302420) for each DMA request regardless of the data size. Note that 16-bit and 8-bit data transfers cannot be specified in single channel DMA mode.
When writing 16-bit data, transfer both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) to address 0x302420 (fixed address).

When writing 24-bit data, perform data transfer twice. First write 24-bit L-channel data to address 0x302420, and then 24-bit R-channel data to address 0x302420.

- Dual channel DMA mode
  If L-channel and R-channel audio data are stored in different locations, use dual channel DMA mode to transfer L and R data separately to the FIFO. I²S Ch.0 issues two DMA requests simultaneously to DMAC Ch.0 and Ch.1. Since DMAC Ch.0 has higher priority than Ch.1, use DMAC Ch.0 to transfer L-channel data first and DMAC Ch.1 to transfer the following R-channel data.
  When writing 16-bit data to the FIFO, perform 16-bit transfer for each DMA request. Note that 8-bit and 32-bit transfer cannot be specified. First transfer 16-bit L-channel data to address 0x302420 using DMAC Ch.0, and then 16-bit R-channel data to address 0x302422 using DMAC Ch.1.
  When writing 24-bit data to the FIFO, perform 32-bit transfer for each DMA request. Note that 8-bit and 16-bit transfer cannot be specified. First transfer 24-bit L-channel data to address 0x302420 using DMAC Ch.0, and then 24-bit R-channel data to address 0x302420 using DMAC Ch.1.

  For more information on the DMA transfer, see Section 22.7.2.

The I²S module provides two status flags I2SFIFOFF0/I2S_STATUS register and I2SFIFOEF0/I2S_STATUS register to show the FIFO full or empty status.
When four stereo data is written to the FIFO, the FIFO becomes full and I2SFIFOFF0 is set to 1. Note that the newest data of the FIFO is overwritten if data is written to I2S_FIFO0 in this status.
When the FIFO becomes empty, I2SFIFOEF0 is set to 1. When data is written to the FIFO, I2SFIFOEF0 is reset to 0. Note, however, that the I²S module continues idle status until the FIFO becomes full again.
Furthermore, the I²S Ch.0 provides the status bits FIFOSTAT0[2:0]/I2S_STATUS register that indicate the FIFO state machine.

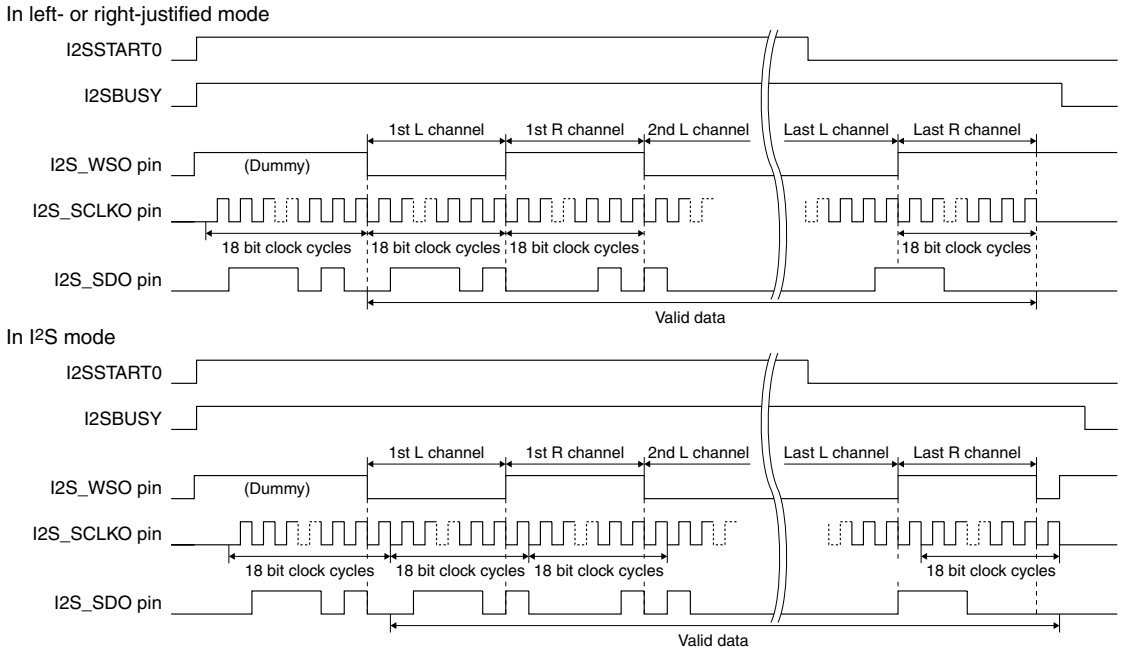
Table 22.5.5 Monitoring the FIFO State Machine

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)

For more information on FIFOSTAT0[2:0], refer to the description of I2S_STATUS register in Section 22.8.

8. To stop output, write 0 to I2SSTART0/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. The bit clock is stopped with pulled down to low. The word select clock is also stopped with pulled down to low if WCLKMD0 = 0 or pulled up to high if WCLKMD0 = 1. When the I²S module stops, I2SBUSY is reset to 0.



Conditions: DATRES0 = 0 (16-bit data), CHMD0[1:0] = 0x0 (stereo), WCLKMD0 = 0 (L ch = low),
BCLKPOL0 = 0 (falling edge), WSCLKCYC0[4:0] = 0x2 (18 clocks)

Figure 22.5.2 Data Output Timing Chart

∗ Output when mute or mono mode is selected

When mute mode is selected using CHMD0[1:0]/I2S_CTL0 register, the I2S_SDO 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_SDO 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 CHMD0[1:0] is changed when data is being output, the mode changes after the current L & R data output has finished.

22.6 Data Input Control (Ch.1)

The following shows audio data input procedure:

1. Set up the I²S conditions as described in Section 22.4.
2. Set up the interrupt or DMA conditions as described in Section 22.7.
3. Write 1 to I2SEN1/I2S_CTL1 register to activate the I²S Ch.1 circuit.
4. Write 1 to I2SSTART1/I2S_START register to start data reception. This setting enables clock input from the I2S_WSI and I2S_SCLKI pins.
   The I²S Ch.1 starts data reception at the falling or rising edge of the I2S_WSI input clock from which the signal goes the first left channel level set using WCLKMD1/I2S_CTL1 register.
   The data bits are loaded to the receive shift register at the rising or falling edge of the I2S_SCLKI input clock specified using BCLKPOL1/I2S_CTL1 register.
   When the receive data is loaded to the shift register, it is sent to the receive FIFO. The first 24-bit or 16-bit data received is loaded as left-channel data, and then the next 24-bit or 16-bit data is loaded as right channel data. The FIFO can store up to four stereo data (24 or 16 bits × 2 channels (L & R) × 4).
   When the number of data according to the interrupt conditions has been loaded to the FIFO, an interrupt or DMA request is generated.
   Read data using this interrupt or DMA as described in Step 5.
5. Read audio data from the FIFO.
   There are two ways to read data.

5.1) Using interrupts

I²S Ch.1 supports two different types of interrupts.

- I²S Ch.1 FIFO whole full interrupt

  When the whole full interrupt is enabled, the I²S module generates an interrupt when the FIFO becomes completely full with four stereo data received.

  Read four stereo data (24 or 16 bits × 2 channels (L & R) × 4) from the FIFO at once in the whole full interrupt handler.

- I²S Ch.1 FIFO one data interrupt

  When the one data interrupt is enabled, the I²S module generates an interrupt every time one stereo data has been loaded to the FIFO. Read one stereo data (24 or 16 bits × 2 channels (L & R) × 1) from the FIFO at once in the one data interrupt handler. The I²S module continues data reception even if this interrupt cause occurs.

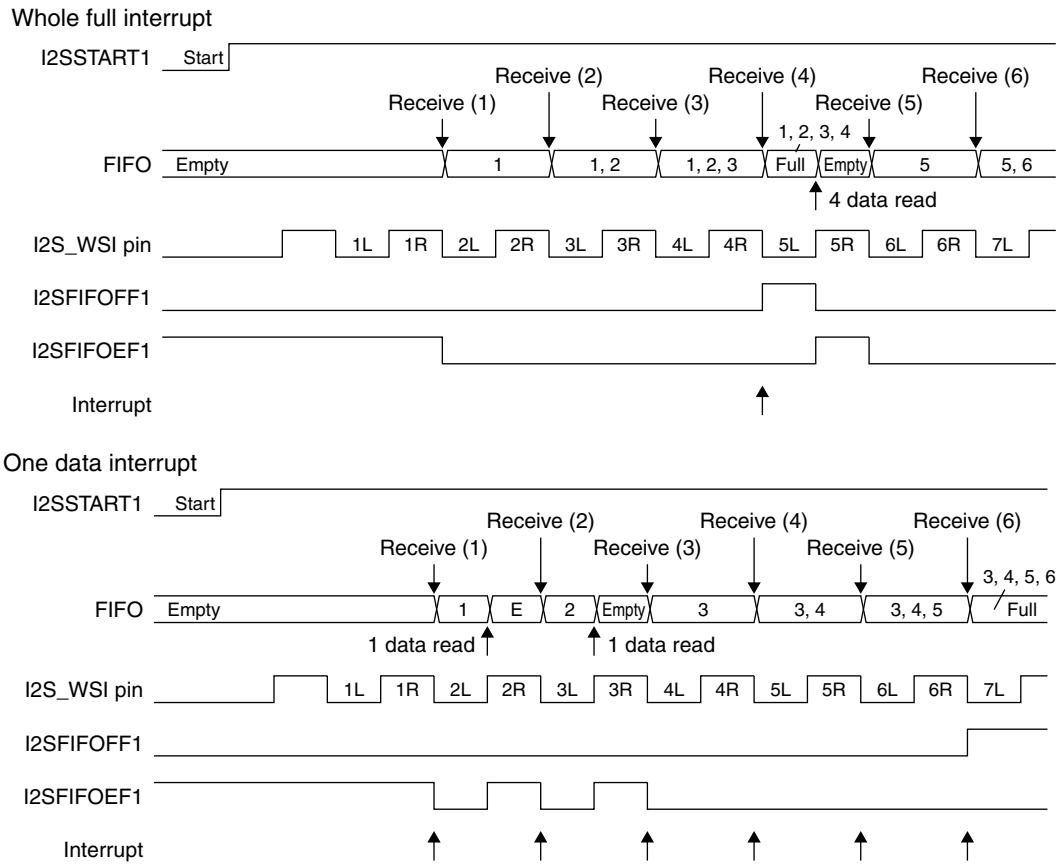


Figure 22.6.1 Receive FIFO Data and Interrupts

For more information on the I²S interrupt, see Section 22.7.1.

The 32-bit register I2S_FIFO1 (address 0x302430) is used to read the received data from the FIFO.

When reading 16-bit data, use a 16-bit or 32-bit memory read instruction. Note that 8-bit memory read instructions cannot be used. When a 16-bit memory read instruction is used, first read 16-bit L-channel data from address 0x302430, and then 16-bit R-channel data from address 0x302432. When a 32-bit memory read instruction is used, read both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) from address 0x302430.

When reading 24-bit data, use a 32-bit memory read instruction. Note that 8-bit and 16-bit memory read instructions cannot be used. First read 24-bit L-channel data from address 0x302430, and then 24-bit R-channel data from address 0x302430.

Both channel data must be read as a pair.

The sequence for reading four groups of stereo data (24 or 16 bits × 2 channels (L & R) × 4) from the FIFO is shown in Tables 22.6.1 to 22.6.3 (the instructions to store the read data to memory and the procedures have been omitted).

Table 22.6.1 Reading 16-bit Data from the FIFO using 16-bit Memory Read Instructions

No.	Memory read instruction	Read details
1	ld.h %rd1, [0x302430]	0x302430 → Data1-L[15:0]
2	ld.h %rd2, [0x302432]	0x302432 → Data1-R[15:0]
3	ld.h %rd3, [0x302430]	0x302430 → Data2-L[15:0]
4	ld.h %rd4, [0x302432]	0x302432 → Data2-R[15:0]
5	ld.h %rd5, [0x302430]	0x302430 → Data3-L[15:0]
6	ld.h %rd6, [0x302432]	0x302432 → Data3-R[15:0]
7	ld.h %rd7, [0x302430]	0x302430 → Data4-L[15:0]
8	ld.h %rd8, [0x302432]	0x302432 → Data4-R[15:0]

Table 22.6.2 Reading 16-bit Data from the FIFO using 32-bit Read Instructions

No.	Memory read instruction	Read details
1	ld.w %rd12, [0x302430]	0x302430 → {Data1-R[15:0], L[15:0]}
2	ld.w %rd34, [0x302430]	0x302430 → {Data2-R[15:0], L[15:0]}
3	ld.w %rd56, [0x302430]	0x302430 → {Data3-R[15:0], L[15:0]}
4	ld.w %rd78, [0x302430]	0x302430 → {Data4-R[15:0], L[15:0]}

(R = D[31:16], L = D[15:0])

Table 22.6.3 Reading 24-bit Data from the FIFO Using 32-bit Read Instructions

No.	Memory read instruction	Read details
1	ld.w %rd1, [0x302430]	0x302430 → Data1-L[23:0]
2	ld.w %rd2, [0x302430]	0x302430 → Data1-R[23:0]
3	ld.w %rd3, [0x302430]	0x302430 → Data2-L[23:0]
4	ld.w %rd4, [0x302430]	0x302430 → Data2-R[23:0]
5	ld.w %rd5, [0x302430]	0x302430 → Data3-L[23:0]
6	ld.w %rd6, [0x302430]	0x302430 → Data3-R[23:0]
7	ld.w %rd7, [0x302430]	0x302430 → Data4-L[23:0]
8	ld.w %rd8, [0x302430]	0x302430 → Data4-R[23:0]

5.2) Using DMA transfer

The I²S Ch.1 supports two different types of DMA mode.

DMA transfers are invoked by the cause of one data interrupt regardless of the DMA mode set. The interrupt signal is output to both the ITC and DMAC. Therefore, DMA transfer can be performed without generating any I²S interrupt.

- Single channel DMA mode

When storing L-channel and R-channel audio data to one memory area sequentially, use single channel DMA mode. I²S Ch.1 issues two DMA requests simultaneously to DMAC Ch.2 and Ch.3, so either DMAC Ch.2 or Ch.3 can be used for data transfer.

Perform 32-bit data transfer to read both L and R data from the FIFO (address 0x302430) for each DMA request regardless of the data size. Note that 16-bit and 8-bit data transfers cannot be specified in single channel DMA mode.

When reading 16-bit data, transfer both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) from address 0x302430 (fixed address).

When reading 24-bit data, perform data transfer twice. First read 24-bit L-channel data from address 0x302430, and then 24-bit R-channel data from address 0x302430.

- Dual channel DMA mode

When storing L-channel and R-channel audio data to different locations, use dual channel DMA mode to transfer L and R data separately from the FIFO. I²S Ch.1 issues two DMA requests simultaneously to DMAC Ch.2 and Ch.3. Since DMAC Ch.2 has higher priority than Ch.3, use DMAC Ch.2 to transfer L-channel data first and DMAC Ch.3 to transfer the following R-channel data.

When reading 16-bit data from the FIFO, perform 16-bit transfer for each DMA request. Note that 8-bit and 32-bit transfer cannot be specified. First transfer 16-bit L-channel data from address 0x302430 using DMAC Ch.2, and then 16-bit R-channel data from address 0x302432 using DMAC Ch.3.
When reading 24-bit data from the FIFO, perform 32-bit transfer for each DMA request. Note that 8-bit and 16-bit transfer cannot be specified. First transfer 24-bit L-channel data from address 0x302430 using DMAC Ch.2, and then 24-bit R-channel data from address 0x302430 using DMAC Ch.3.

For more information on the DMA transfer, see Section 22.7.2.

The I²S module provides two status flags I2SFIFOFF1/I2S_STATUS register and I2SFIFOEF1/I2S_STATUS register to show the FIFO full or empty status.
When the FIFO becomes full with four stereo data received, the I2SFIFOFF1 is set to 1.

Note: If the I²S Ch.1 FIFO is not read within a one-word clock cycle after I2SFIFOFF1 has been set to 1, the latest stereo data in the FIFO is overwritten by the next data received.

When the FIFO becomes empty after all stored data have been read, I2SFIFOEF1 is set to 1.

6. To stop reception, write 0 to I2SSTART1/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.

I²S bypass mode

The I²S module supports bypass mode. In this mode, the signals input from the I2S_MCLKI, I2S_SCLKI, I2S_WSI, and I2S_SDI pins are directly output from the I2S_MCLKO, I2S_SCLKO, I2S_WSO, and I2S_SDO pins, respectively. Set I2SBYPASS/I2S_CTL1 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.

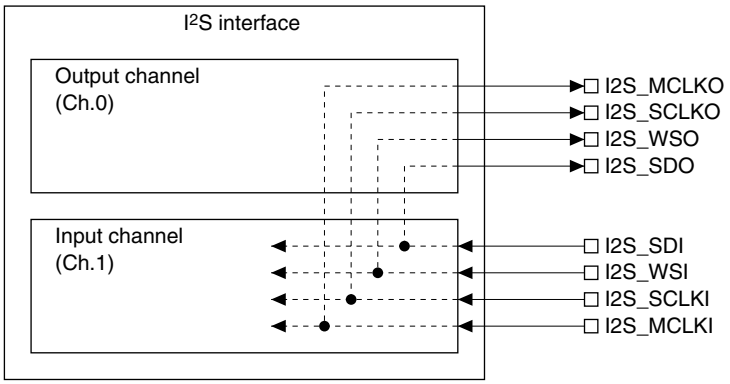


Figure 22.6.2 Bypass Mode

22.7 I²S Interrupt and DMA

This section describes the I²S interrupts and invoking DMA.
For more information on interrupt processing and DMA transfer, see the “Interrupt Controller (ITC)” chapter and the “DMA Controller (DMAC)” chapter, respectively.

22.7.1 Interrupts

The I²S module includes a function for generating the following five different types of interrupts.

I²S Ch.0
- I²S Ch.0 FIFO whole empty interrupt
- I²S Ch.0 FIFO half empty interrupt
- I²S Ch.0 FIFO one empty interrupt

I²S Ch.1
- I²S Ch.1 FIFO whole full interrupt
- I²S Ch.1 FIFO one data interrupt

Each I²S channel outputs one interrupt signal shared by the three or two interrupt causes to the interrupt controller (ITC). Set the interrupt enable bit to 1 using the I2S_INT register to generate the corresponding interrupt.

I²S Ch.0 FIFO whole empty interrupt

To use this interrupt, set I2SWEIE0/I2S_INT register to 1. If I2SWEIE0 is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When all data (four stereo data) have been read from the I²S Ch.0 FIFO to output, the I²S module sets I2SWEIF0/I2S_INT register to 1, indicating that the FIFO is empty. If whole empty interrupts are enabled (I2SWEIE0 = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. If I2SWEIF0 is 1, the application program should fill the FIFO with four stereo data (24 or 16 bits × 2 channels (L & R) × 4).

I²S Ch.0 FIFO half empty interrupt

To use this interrupt, set I2SHEIE0/I2S_INT register to 1. If I2SHEIE0 is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When a free space for two stereo data becomes available in the I²S Ch.0 FIFO, the I²S module sets I2SHEIF0/I2S_INT register to 1. If half empty interrupts are enabled (I2SHEIE0 = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. If I2SHEIF0 is 1, the application program should fill the FIFO with two stereo data (24 or 16 bits × 2 channels (L & R) × 2).

I²S Ch.0 FIFO one empty interrupt

To use this interrupt, set I2SOEIE0/I2S_INT register to 1. If I2SOEIE0 is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When a free space for one stereo data becomes available in the I²S Ch.0 FIFO, the I²S module sets I2SOEIF0/I2S_INT register to 1. If one empty interrupts are enabled (I2SOEIE0 = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. If I2SOEIF0 is 1, the application program should fill the FIFO with one stereo data (24 or 16 bits × 2 channels (L & R) × 1).

Note: At the beginning of I²S output (by setting I2SSTART0 to 1), the FIFO is completely empty. So the I²S module continues the initial status until the FIFO becomes full with four stereo data (24 or 16 bits × 2 channels (L & R) × 4).
The interrupt handler should write one, two or four stereo data to the FIFO according to the interrupt mode selected.

- When filling the FIFO before enabling the interrupt
  The application program should write four stereo data to the FIFO first, and then enable the interrupt.
- When filling the FIFO after enabling the interrupt
  The application program can enable the interrupt directly. The FIFO becomes full when the first whole-empty interrupt, second half-empty interrupt, or fourth one-empty interrupt occurs after the interrupt is enabled.

I²S Ch.1 FIFO whole full interrupt

To use this interrupt, set I2SWFIE1/I2S_INT register to 1. If I2SWFIE1 is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When the I²S Ch.1 FIFO becomes full with four received data (four stereo data), the I²S module sets I2SWFIF1/I2S_INT register to 1, indicating that the FIFO is completely full. If whole full interrupts are enabled (I2SWFIE1 = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. If I2SWFIF1 is 1, the application program should read four stereo data (24 or 16 bits × 2 channels (L & R) × 4) from the FIFO.

I²S Ch.1 FIFO one data interrupt

To use this interrupt, set I2SOFIE1/I2S_INT register to 1. If I2SOFIE1 is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When data (one stereo data) is loaded to the I²S Ch.1 FIFO, the I²S module sets I2SOFIF1/I2S_INT register to 1, indicating that the FIFO contains a received data. If one data interrupts are enabled (I2SOFIE1 = 1), an interrupt request is sent simultaneously to the ITC. An interrupt occurs if other interrupt conditions are met. If I2SOFIF1 is 1, the application program should read one stereo data (24 or 16 bits × 2 channels (L & R) × 1) from the FIFO.

For more information on interrupt processing, see the "Interrupt Controller (ITC)" chapter.

22.7.2 DMA Transfer

The causes of I²S Ch.0 FIFO one empty interrupt and I²S Ch.1 FIFO one data interrupt can invoke a DMA. This allows continuous data input/output through DMA transfer between memory and the FIFO. The DMA request signals are provided independently of the interrupt request signals. Therefore, DMA transfer can be performed without generating any I²S interrupt.
The following lists the DMAC channels that allow selection of the I²S interrupt cause as the trigger:

For I²S Ch.0 (data output)
DMAC Ch.0: Used for L and R data transfer with single DMA, or L data transfer with dual DMA.
DMAC Ch.1: Used for L and R data transfer with single DMA, or R data transfer with dual DMA.

For I²S Ch.1 (data input)
DMAC Ch.2: Used for L and R data transfer with single DMA, or L data transfer with dual DMA.
DMAC Ch.3: Used for L and R data transfer with single DMA, or R data transfer with dual DMA.

Use single or dual DMAC channels according to the audio data storing method. Also the I²S channels must be set to either single channel DMA mode or dual channel DMA mode using I2SDMA0/I2S_MODE register for Ch.0 and I2SDMA1/I2S_MODE register for Ch.1.
When I2SDMA*x* is 0 (default), I²S Ch.*x* is placed into single channel DMA mode, and when I2SDMA*x* is set to 1, I²S Ch.*x* enters dual channel DMA mode.

Single channel DMA mode

I²S Ch.0

If L-channel and R-channel audio data are sequentially stored in one memory area, use single channel DMA mode. I²S Ch.0 issues two DMA requests simultaneously to DMAC Ch.0 and Ch.1, so either DMAC Ch.0 or Ch.1 can be used for data transfer.
Perform 32-bit data transfer to write both L and R data to the FIFO (address 0x302420) for each DMA request regardless of the data size. Note that 16-bit and 8-bit data transfers cannot be specified in single channel DMA mode.
When writing 16-bit data, transfer both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) to address 0x302420 (fixed address).
When writing 24-bit data, perform data transfer twice. First write 24-bit L-channel data to address 0x302420, and then 24-bit R-channel data to address 0x302420.

I²S Ch.1

When storing L-channel and R-channel audio data to one memory area sequentially, use single channel DMA mode. I²S Ch.1 issues two DMA requests simultaneously to DMAC Ch.2 and Ch.3, so either DMAC Ch.2 or Ch.3 can be used for data transfer.
Perform 32-bit data transfer to read both L and R data from the FIFO (address 0x302430) for each DMA request regardless of the data size. Note that 16-bit and 8-bit data transfers cannot be specified in single channel DMA mode.
When reading 16-bit data, transfer both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) from address 0x302430 (fixed address).
When reading 24-bit data, perform data transfer twice. First read 24-bit L-channel data from address 0x302430, and then 24-bit R-channel data from address 0x302430.

Dual channel DMA mode

I²S Ch.0

If L-channel and R-channel audio data are stored in different locations, use dual channel DMA mode to transfer L and R data separately to the FIFO. I²S Ch.0 issues two DMA requests simultaneously to DMAC Ch.0 and Ch.1. Since DMAC Ch.0 has higher priority than Ch.1, use DMAC Ch.0 to transfer L-channel data first and DMAC Ch.1 to transfer the following R-channel data.

When writing 16-bit data to the FIFO, perform 16-bit transfer for each DMA request. Note that 8-bit and 32-bit transfer cannot be specified. First transfer 16-bit L-channel data to address 0x302420 using DMAC Ch.0, and then 16-bit R-channel data to address 0x302422 using DMAC Ch.1.

When writing 24-bit data to the FIFO, perform 32-bit transfer for each DMA request. Note that 8-bit and 16-bit transfer cannot be specified. First transfer 24-bit L-channel data to address 0x302420 using DMAC Ch.0, and then 24-bit R-channel data to address 0x302420 using DMAC Ch.1.

I²S Ch.1

When storing L-channel and R-channel audio data to different locations, use dual channel DMA mode to transfer L and R data separately from the FIFO. I²S Ch.1 issues two DMA requests simultaneously to DMAC Ch.2 and Ch.3. Since DMAC Ch.2 has higher priority than Ch.3, use DMAC Ch.2 to transfer L-channel data first and DMAC Ch.3 to transfer the following R-channel data.

When reading 16-bit data from the FIFO, perform 16-bit transfer for each DMA request. Note that 8-bit and 32-bit transfer cannot be specified. First transfer 16-bit L-channel data from address 0x302430 using DMAC Ch.2, and then 16-bit R-channel data from address 0x302432 using DMAC Ch.3.

When reading 24-bit data from the FIFO, perform 32-bit transfer for each DMA request. Note that 8-bit and 16-bit transfer cannot be specified. First transfer 24-bit L-channel data from address 0x302430 using DMAC Ch.2, and then 24-bit R-channel data from address 0x302430 using DMAC Ch.3.

For more information on DMA transfer, see the “DMA Controller (DMAC)” chapter.

22.8 Control Register Details

Table 22.8.1 List of I²S Registers

Address	Register name		Function
0x302400	I2S_CTL0	I²S Ch.0 Control Register	Control I²S Ch.0 output
0x302404	I2S_CTL1	I²S Ch.1 Control Register	Control I²S Ch.1 input
0x302408	I2S_DV_MCLK	I²S Master Clock Division ratio Register	Configure master clock
0x30240c	I2S_DV_BCLK	I²S Audio Clock Division ratio Register	Configure audio clock
0x302410	I2S_START	I²S Start/Stop Register	Control/indicate I²S start/stop status
0x302414	I2S_STATUS	I²S FIFO Status Register	Indicate FIFO status
0x302418	I2S_INT	I²S Interrupt Control Register	Control I²S interrupts
0x30241c	I2S_MODE	I²S Mode Select Register	Set DMA and pin modes
0x302420	I2S_FIFO0	I²S Ch.0 FIFO Register	I²S Ch.0 output data
0x302422			
0x302430	I2S_FIFO1	I²S Ch.1 FIFO Register	I²S Ch.1 input data
0x302432			

The following describes each I²S register.

Note: When data is written to the registers, the “Reserved” bits must always be written as 0 and not 1.

I²S Ch.0 Control Register (I2S_CTL0)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
I²S Ch.0 Control Register (I2S_CTL0)	0x302400 (32 bits)	D31–11	–	reserved	–				–	–	0 when being read.
		D10	**DTSIGN0**	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	**DTFORM0**	I²S Ch.0 output data format select	1	LSB first	0	MSB first	0	R/W	
		D4	**I2SOUTEN0**	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	**CHMD0[1:0]**	I²S Ch.0 output channel mode select	CHMD0[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/I2S_START register to start data output from I²S Ch.0.

D[31:11] Reserved

D10 DTSIGN0: I²S Ch.0 Signed/Unsigned Data Format Select Bit

Selects the output data format when I²S Ch.0 is set in right-justified mode.
1 (R/W): Signed
0 (R/W): Unsigned (default)

Setting DTSIGN0 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. Set DTSIGN0 to 0 when another data output timing mode is selected.

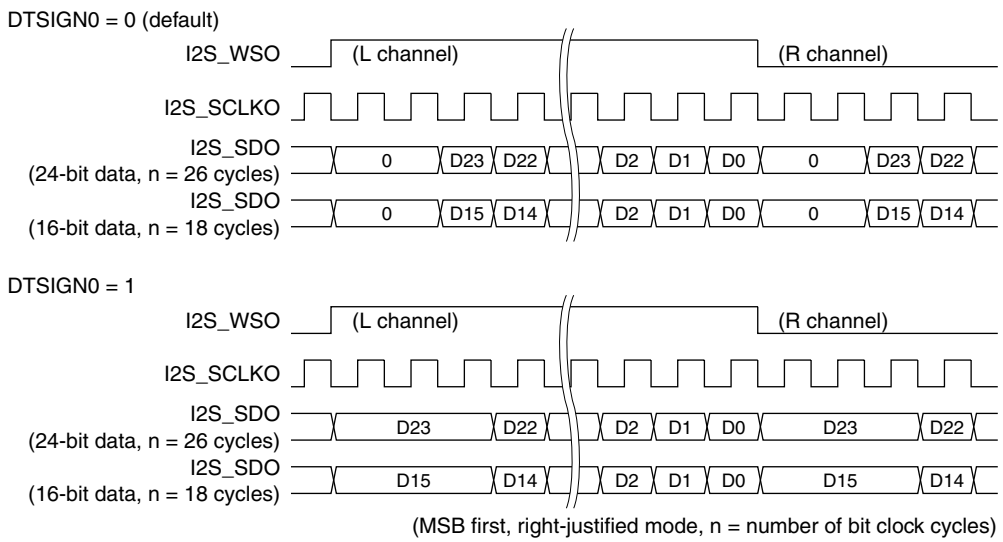


Figure 22.8.1 Unsigned and Signed Format

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.

D8 I2SEN0: I²S Ch.0 Enable Bit

Enables/disables operations of the I²S Ch.0.
1 (R/W): Enabled (On)
0 (R/W): Disabled (Off) (default)

Setting I2SEN0 to 1 activates I²S Ch.0 to enable data transfer. Setting I2SEN0 to 0 stops I²S Ch.0.

D7 WCLKMD0: I²S Ch.0 Output Word Clock Mode Select Bit

Selects the I2S_WSO 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)

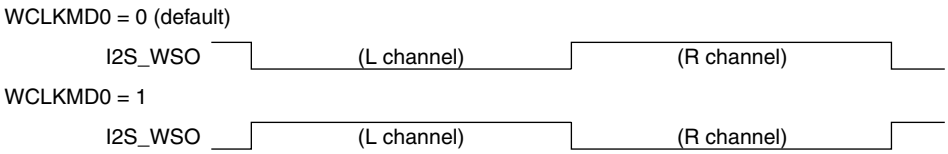


Figure 22.8.2 Word Clock Mode

D6 BCLKPOL0: I²S Ch.0 Output Bit Clock Polarity Select Bit

Selects the bit clock (I2S_SCLKO) polarity.
1 (R/W): Negative
0 (R/W): Positive (default)

When BCLKPOL0 is 0, the I2S_SDO output changes at the falling edge of the I2S_SCLKO clock (bit clock) and the external DAC samples the data bit at the rising edge of I2S_SCLKO.
When BCLKPOL0 is set to 1, the I2S_SDO output changes at the rising edge of I2S_SCLKO and the external DAC samples the data bit at the falling edge of I2S_SCLKO.

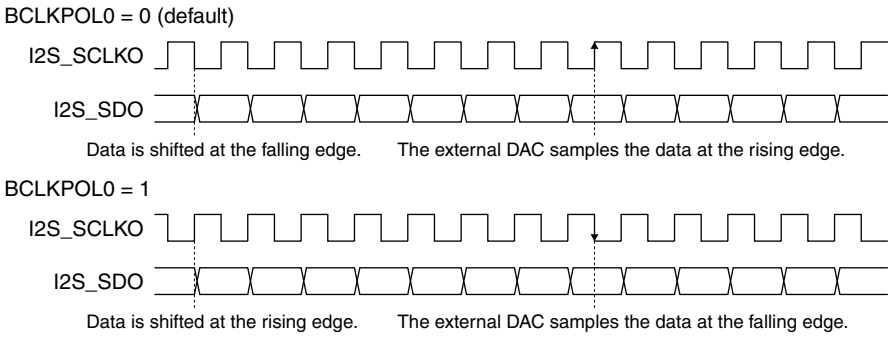


Figure 22.8.3 Bit Clock Polarity

D5 DTFORM0: I²S Ch.0 Output Data Format Select Bit

Selects either MSB first or LSB first as the I²S Ch.0 data output direction.
1 (R/W): LSB first
0 (R/W): MSB first (default)

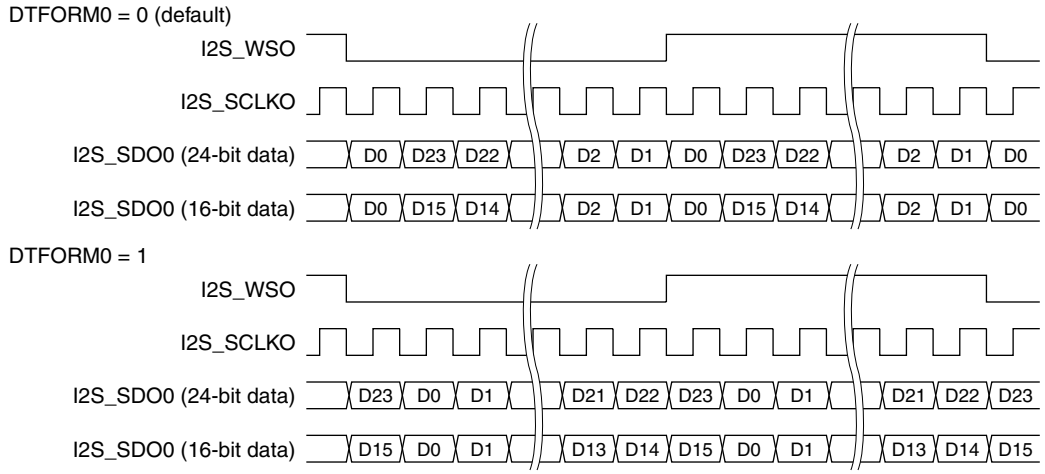


Figure 22.8.4 Output Data Format (Example in I²S Mode)

D4 I2SOUTEN0: I²S Ch.0 Output Enable Bit

Enables/disables output of the I²S Ch.0 signals.
1 (R/W): Enabled (On)
0 (R/W): Disabled (Off) (default)

When I2SOUTEN0 = 0, the I2S_MCLKO pin is maintained at standby status and the I2S_WSO pin is fixed at 0. The I2S_SDO pin is left unchanged. The I2S_SCLKO pin is fixed at 0 (when BCLKPOL0 = 0) or 1 (when BCLKPOL0 = 1).
When I2SOUTEN0 is set to 1, all output pins enter standby status.
I2SOUTEN0 can be altered even if data is being output. This enables the I²S module to pause or resume shifting data (data output).

D[3:2] DTTMG0[1:0]: I²S Ch.0 Output Data Timing Select Bits

Selects the data bit output timing.

Table 22.8.2 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_SCLKO clock delay from the I2S_WSO signal edge.

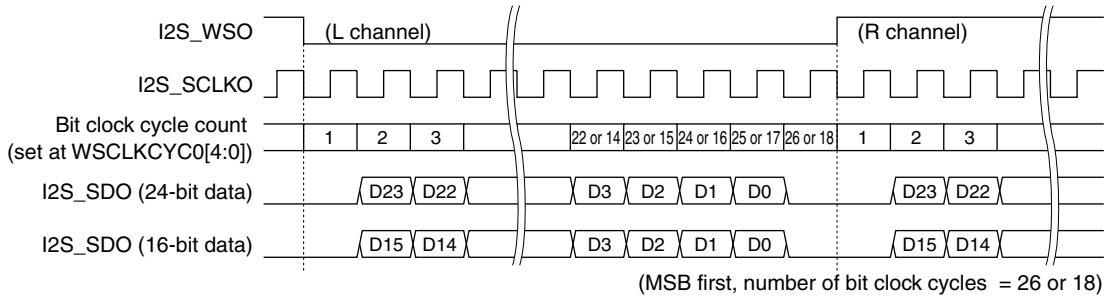


Figure 22.8.5 Data Output Timing 1 (I²S Mode)

When DTTMG0[1:0] is set to 0x1, left-justified mode is selected. In this mode, each data output will start at the I2S_WSO signal edge.

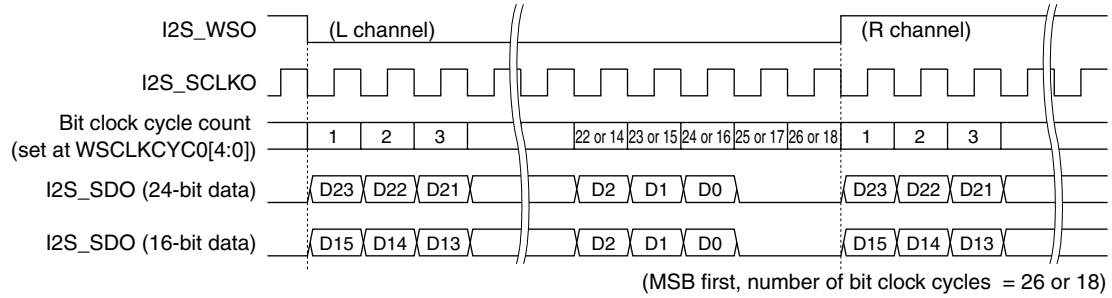


Figure 22.8.6 Data Output Timing 2 (Left-Justified Mode)

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_WSO signal edge.

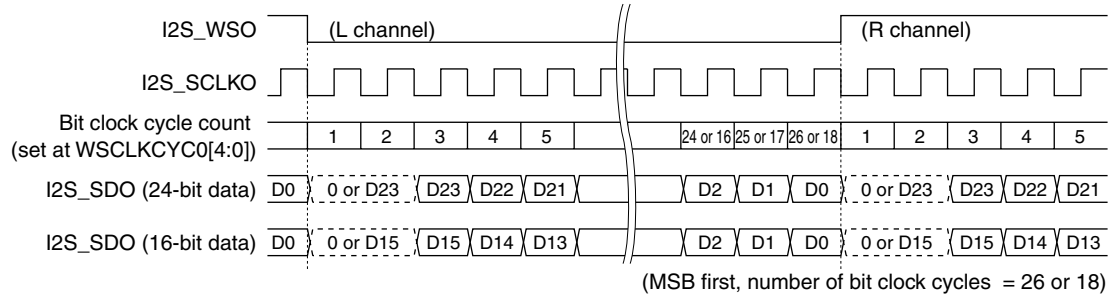


Figure 22.8.7 Data Output Timing 3 (Right-Justified Mode)

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] CHMD0[1:0]: I²S Ch.0 Output Channel Mode Select Bits

Selects the I²S Ch.0 output channel mode.

Table 22.8.3 Output Channel Mode

CHMD0[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_SDO 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_SDO pin is fixed at 0 during the output period for the unselected channel.

The FIFO data is read out normally, therefore an interrupt occurs.

I²S Ch.1 Control Register (I2S_CTL1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Ch.1 Control Register (I2S_CTL1)	**0x302404** (32 bits)	D31–7	**–**	reserved	–	–	–	0 when being read.
		D6	**I2SBYPASS**	I²S bypass mode select	1 Bypass 0 Normal	0	R/W	
		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] Timing mode 0x3 reserved 0x2 Right justified 0x1 Left justified 0x0 I²S	0x0	R/W	
		D1	**DATRES1**	I²S Ch.1 input data resolution select	1 24 bits 0 16 bits	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/I2S_START register to start data input to I²S Ch.1.

D[31:7] Reserved

D6 I2SBYPASS: I²S Bypass Mode Select Bit

Selects I²S bypass mode.
1 (R/W): Bypass mode
0 (R/W): Normal mode (default)

Setting I2SBYPASS to 1 places the I²S module into bypass mode. In this mode, the signals input from the I2S_MCLKI, I2S_SCLKI, I2S_WSI, and I2S_SDI pins are directly output from the I2S_MCLKO, I2S_SCLKO, I2S_WSO, and I2S_SDO pins, respectively. When I2SBYPASS is 0 (default), I²S Ch.0 and Ch.1 can be used independently.

D5 WCLKMD1: I²S Ch.1 Input Word Clock Mode Select Bit

Selects the I2S_WSI 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)

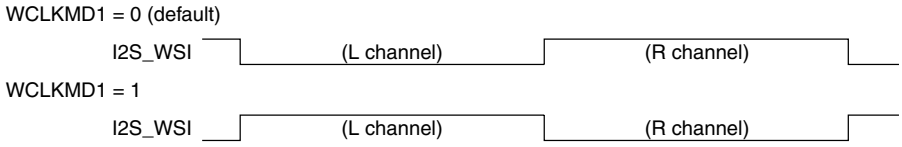


Figure 22.8.8 Word Clock Mode

D4 BCLKPOL1: I²S Ch.1 Input Bit Clock Polarity Select Bit

Selects the bit clock (I2S_SCLKI) polarity.
1 (R/W): Negative
0 (R/W): Positive (default)

When BCLKPOL1 is 0, the I2S_SDI input changes at the falling edge of the I2S_SCLKI clock (bit clock) and the I²S Ch.1 samples the data bit at the rising edge of I2S_SCLKI.
When BCLKPOL1 is set to 1, the I2S_SDI input changes at the rising edge of I2S_SCLKI and the I²S Ch.1 samples the data bit at the falling edge of I2S_SCLKI.

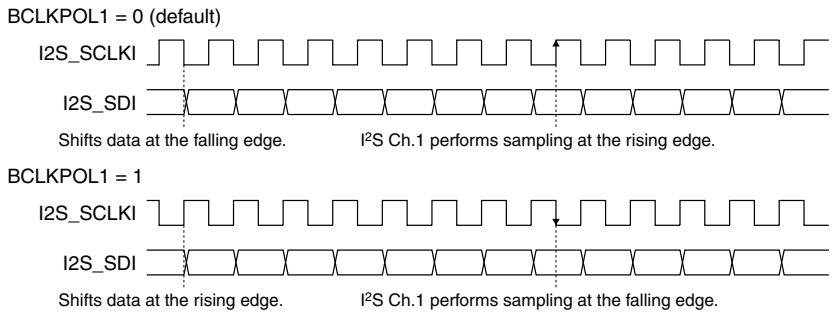


Figure 22.8.9 Bit Clock Polarity

D[3:2] DTTMG1[1:0]: I²S Ch.1 Input Data Timing Select Bits

Selects the data bit input timing.

Table 22.8.4 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_SCLKI clock delay from the I2S_WSI signal edge.

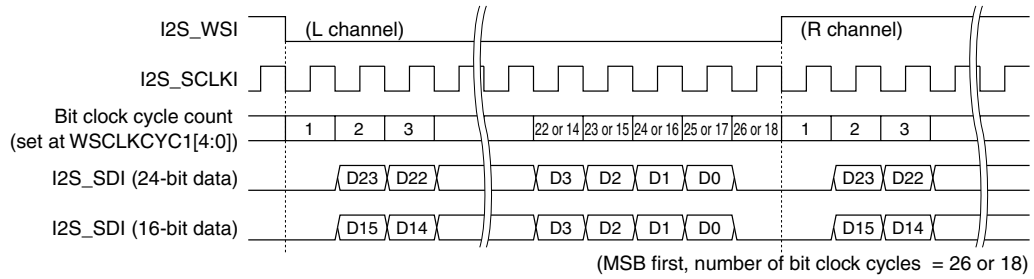


Figure 22.8.10 Data Input Timing 1 (I²S Mode)

When DTTMG1[1:0] is set to 0x1, left-justified mode is selected. In this mode, each data input will start at the I2S_WSI signal edge.

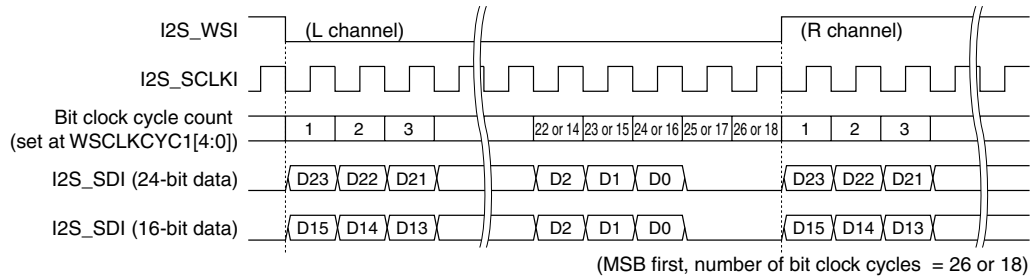


Figure 22.8.11 Data Input Timing 2 (Left-Justified Mode)

When DTTMG1[1:0] is set to 0x2, right-justified mode is selected. In this mode, input data will be right justified to the I2S_WSI signal edge.

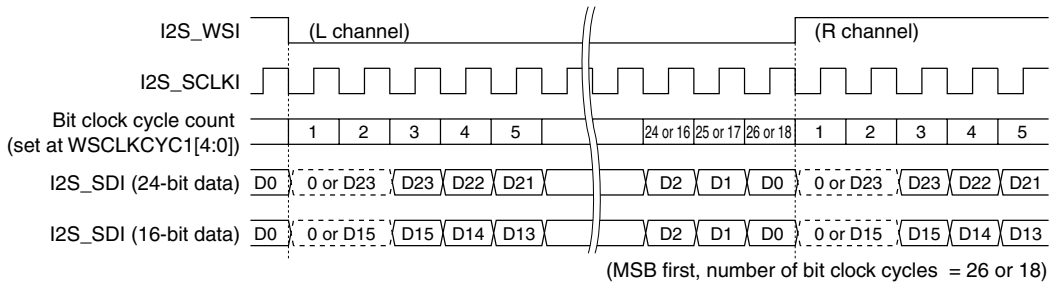


Figure 22.8.12 Data Input Timing 3 (Right-Justified Mode)

D1 DATRES1: I²S Ch.1 Input Data Resolution Select Bit

Selects the input audio data resolution for I²S Ch.1.
1 (R/W): 24 bits
0 (R/W): 16 bits (default)

Setting DATRES1 to 0 (default) selects 16 bits and setting 1 selects 24 bits.

D0 I2SEN1: I²S Ch.1 Enable Bit

Enables/disables operations of the I²S Ch.1.
1 (R/W): Enabled (On)
0 (R/W): Disabled (Off) (default)

Setting I2SEN1 to 1 activates I²S Ch.1 to enable data transfer. Setting I2SEN1 to 0 stops I²S Ch.1.

I²S Master Clock Division Ratio Register (I2S_DV_MCLK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Master Clock Division Ratio Register (I2S_DV_MCLK)	0x302408 (32 bits)	D31–16	–	reserved	–	–	–	0 when being read.
		D15	MCLKSEL	I2S_MCLK source clock select	1 External clk / 0 System clk	0	R/W	
		D14–6	–	reserved	–	–	–	0 when being read.
		D5–0	MCLKDIV [5:0]	I2S_MCLK division ratio select	MCLKDIV[5:0] / Division ratio	0x0	R/W	Source clock = I2S_CLK (PCLK)
					0x3f / 1/64			
					0x3e / 1/63			
					0x3d / 1/62			
					: / :			
					0x2 / 1/3			
					0x1 / 1/2			
					0x0 / reserved			

D[31:16] Reserved

D15 MCLKSEL: I2S_MCLK Source Clock Select Bit

Selects the source clock for generating the I²S Ch.0 master clock (I2S_MCLKO).
1 (R/W): External clock
0 (R/W): System clock (PCLK) (default)

When MCLKSEL is set to 0 (default), the I²S module generates the master clock (I2S_MCLKO) by dividing PCLK. When MCLKSEL is set to 1, the I²S Ch.0 uses the external clock input from the I2S_MCLKI pin as the master clock.

D[14:6] Reserved

D[5:0] MCLKDIV[5:0]: I2S_MCLK Division Ratio Select Bits

Configures the I²S master clock (I2S_MCLKO) to be output from the I2S_MCLKO pin.
The I²S module generates the I2S_MCLKO by dividing the operating clock (PCLK generated by the CMU). Specify the division ratio using MCLKDIV[5:0].

Table 22.8.5 I2S_MCLKO (Master Clock) Settings

MCLKDIV[5:0]	PCLK division ratio
0x3f	1/64
0x3e	1/63
0x3d	1/62
:	:
0x2	1/3
0x1	1/2
0x0	Reserved

(Default: 0x0)

I²S Audio Clock Division Ratio Register (I2S_DV_BCLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
I²S Audio Clock Division Ratio Register (I2S_DV_BCLK)	**0x30240c** (32 bits)	D31–21	**–**	reserved	–		–	–	0 when being read.
		D20–16	**WSCLKCYC1 [4:0]**	I²S Ch.1 WS clock cycle setup	WSCLKCYC1[4:0]	Clock period	0x0	R/W	
					Other	reserved			
					0x10	32 clocks			
					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			
		D15–13	**–**	reserved	–		–	–	0 when being read.
		D12–8	**WSCLKCYC0 [4:0]**	I²S Ch.0 WS clock cycle setup	WSCLKCYC0[4:0]	Clock period	0x0	R/W	
					Other	reserved			
					0x10	32 clocks			
					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			
		D7–0	**BCLKDIV0 [7:0]**	I²S Ch.0 bit clock division ratio select	BCLKDIV0[7:0]	Division ratio	0x0	R/W	Source clock = Clock selected by MCLKSEL
					0xff	1/512			
					0xfe	1/510			
					0xfd	1/508			
					:	:			
					0x2	1/6			
					0x1	1/4			
					0x0	1/2			

D[31:21] Reserved

D[20:16] WSCLKCYC1[4:0]: I²S Ch.1 WS Clock Cycle Setup Bits

I²S Ch.1 inputs the sampling clock from the I2S_WSI pin. The clock period must be specified with the number of bit clock cycles using WSCLKCYC1[4:0]. See the description of WSCLKCYC0[4:0].

D[15:13] Reserved

D[12:8] WSCLKCYC0[4:0]: I²S Ch.0 WS Clock Cycle Setup Bits

Specifies the sampling clock (I2S_WSO signal) period.

I²S Ch.0 generates the sample clock to be output from the I2S_WSO pin by counting the bit clock configured with BCLKDIV0[7:0]. Specify the half cycle (a high or low level period) of the I2S_WSO clock with the number of bit clock cycles using WSCLKCYC0[4:0].

Table 22.8.6 Sampling Clock Settings

WSCLKCYC*x*[4:0]	Sampling clock period (number of bit clock cycles)
0x1f–0x11	Reserved
0x10	32 clocks
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.

$$fs = \frac{f_{I2S\_SCLK}}{n \times 2} \text{ [Hz]}$$

fs: Sampling clock frequency [Hz]
f_{I2S_SCLK}: Bit clock frequency [Hz] (Ch.0: See Table 22.8.7. Ch.1: I2S_SCLKI input clock frequency)
n: Number of bit clocks selected by WSCLKCYC*x*[4:0] (See Table 22.8.6.)

Note: The value to be set to the WSCLKCYC*x*[4:0] is not the number of audio data bits, but the number of bit clock cycles that is used to adjust the sampling clock period. It must be equal to or greater than the number of audio data bits (24 or 16 bits).

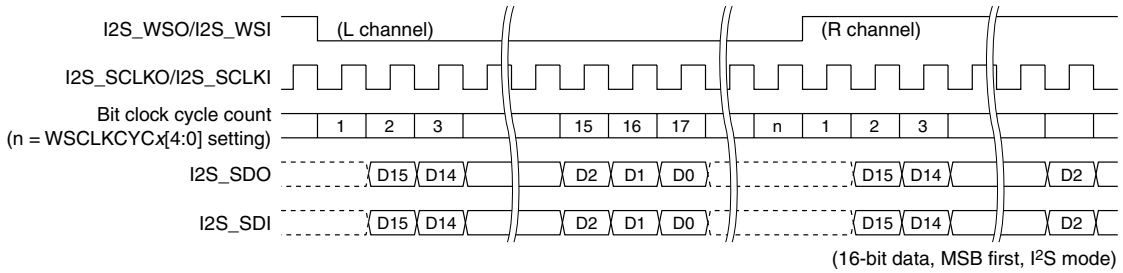


Figure 22.8.13 Sampling Clock Period

D[7:0] BCLKDIV0[7:0]: I²S Ch.0 Bit Clock Division Ratio Select Bits

Configures the bit clock to be output from I²S Ch.0.

The I²S module generates the bit clock to be output from the I2S_SCLKO pin of I²S Ch.0 by dividing the source clock selected for generating I2S_MCLKO. Specify the division ratio using BCLKDIV0[7:0].

Table 22.8.7 Bit Clock (Division Ratio) Settings

BCLKDIV0[7:0]	Division ratio
0xff	1/512
0xfe	1/510
0xfd	1/508
:	:
0x2	1/6
0x1	1/4
0x0	1/2

(Source clock = PCLK or I2S_MCLKI input clock, default: 0x0)

The I²S Ch.0 bit clock frequency is calculated as below.

$$f_{I2S\_SCLKO} = \frac{f_{SRC\_CLK}}{(BCLKDIV + 1) \times 2} \text{ [Hz]}$$

f_{I2S_SCLKO}: I²S Ch.0 bit clock frequency [Hz]
f_{SRC_CLK}: PCLK or I2S_MCLKI input clock frequency [Hz]
BCLKDIV: BCLKDIV0[7:0] set value (0x0–0xff)

I²S Ch.1 uses the bit clock input from the I2S_SCLKI pin, therefore the above setting is not applied to Ch.1.

I²S Start/Stop Register (I2S_START)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Start/Stop Register (I2S_START)	0x302410 (32 bits)	D31–9	–	reserved	–	–	–	0 when being read.
		D8	I2SSTART1	I²S Ch.1 start/stop control	1 Start (run) 0 Stop	0	R/W	
		D7	I2SBUSY	I²S 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[31:9] Reserved

D8 I2SSTART1: I²S Ch.1 Start/Stop Control Bit

Starts/stops data reception of I²S Ch.1.
1 (R/W): Start
0 (R/W): Stop (default)

Writing 1 to I2SSTART1 starts serial data reception through the I2S_SDI 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 I2SBUSY: I²S Busy Flag Bit

Indicates the operating status of I²S Ch.0.
1 (R): Busy
0 (R): Idle (default)

I2SBUSY is set to 1 when I²S Ch.0 starts data transfer and stays 1 while data is being transferred. This flag is cleared to 0 upon completion of the data transfer operation. I2SBUSY is effective only in I²S Ch.0.

D[6:1] Reserved

D0 I2SSTART0: I²S Ch.0 Start/Stop Control Bit

Starts/stops data output of I²S Ch.0.
1 (R/W): Start
0 (R/W): Stop (default)

Writing 1 to I2SSTART0 starts serial data transmission through the I2S_SDO pin.
Writing 0 to I2SSTART0 stops transmission after all the data currently stored in the I²S Ch.0 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 and I2S_DV_BCLK registers when I2SSTART0 is 1.

I²S FIFO Status Register (I2S_STATUS)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S FIFO Status Register (I2S_STATUS)	0x302414 (32 bits)	D31–10	–	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[31:10] 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 (24 or 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 22.8.8 Monitoring Transmit FIFO State

FIFOSTAT0[2:0]	State
0x7–0x6	Reserved
0x5	FLUSH: Transmit FIFO is flushing the remained audio data before it stops. This means that I²S Ch.0 is stopped by setting I2SSTART0 to 0, but there are some audio data remained in the FIFO. The I²S module enters FLUSH state until all the remained audio data has been shifting out. After that, the FIFOSTAT0[2:0] changes from FLUSH to STOP. If the application program restarts I²S Ch.0 by setting I2SSTART0 to 1 again in FLUSH state, FIFOSTAT0[2:0] changes back to FULL or LACK according to the current FIFO status.
0x4	EMPTY: Transmit FIFO is empty. This means that the FIFO became completely empty, but it has not been filled to full yet. Once the FIFO becomes completely empty, FIFOSTAT0[2:0] is set to EMPTY. When the application program fills the FIFO with less than four stereo data (24 or 16 bits × 2 channels (L & R) × 4), FIFOSTAT0[2:0] retains EMPTY and I²S Ch.0 stops shifting out data until the FIFO becomes full again. After the FIFO becomes full, FIFOSTAT0[2:0] is set to FULL and I²S Ch.0 starts shifting out data again.
0x3	LACK: Transmit FIFO is not full and not empty. This means that the FIFO has data, but is not full and also not empty.

FIFOSTAT0[2:0]	State
0x2	FULL: Transmit FIFO is full. This means that the FIFO becomes full with four stereo data. I²S Ch.0 will start shifting out the buffered data.
0x1	INIT: Initialize all four entries of transmit FIFO. This means that I²S Ch.0 is started and waits for filling the FIFO with the first four stereo data.
0x0	STOP: Transmit FIFO is idle. This means that I²S Ch.0 is stopped.

(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 (24 or 16 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 sent.

I²S Interrupt Control Register (I2S_INT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Interrupt Control Register (I2S_INT)	**0x302418** (32 bits)	D31–15	**–**	reserved	–	–	–	0 when being read.
		D14	**I2SWFIF1**	I²S Ch.1 FIFO whole full int. flag	1 Occurred 0 Not occurred	0	R/W	Reset by writing 1.
		D13	**–**	reserved	–	–	–	0 when being read.
		D12	**I2SOFIF1**	I²S Ch.1 FIFO one data int. flag	1 Occurred 0 Not occurred	0	R/W	Reset by writing 1.
		D11	**–**	reserved	–	–	–	0 when being read.
		D10	**I2SWEIF0**	I²S Ch.0 FIFO whole empty int. flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D9	**I2SHEIF0**	I²S Ch.0 FIFO half empty int. flag		0	R/W	
		D8	**I2SOEIF0**	I²S Ch.0 FIFO one empty int. flag		0	R/W	
		D7	**–**	reserved	–	–	–	0 when being read.
		D6	**I2SWFIE1**	I²S Ch.1 FIFO whole full int. enable	1 Enable 0 Disable	0	R/W	
		D5	**–**	reserved	–	–	–	0 when being read.
		D4	**I2SOFIE1**	I²S Ch.1 FIFO one data int. enable	1 Enable 0 Disable	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2	**I2SWEIE0**	I²S Ch.0 FIFO whole empty int. enable	1 Enable 0 Disable	0	R/W	
		D1	**I2SHEIE0**	I²S Ch.0 FIFO half empty int. enable	1 Enable 0 Disable	0	R/W	
		D0	**I2SOEIE0**	I²S Ch.0 FIFO one empty int. enable	1 Enable 0 Disable	0	R/W	

D[31:15] Reserved

D14 I2SWFIF1: I²S Ch.1 FIFO Whole Full Interrupt Flag Bit

Indicates whether the cause of I²S Ch.1 FIFO whole full interrupt has occurred or not.

1 (R): Cause of interrupt has occurred

0 (R): No cause of interrupt has occurred (default)

1 (W): Flag is reset

0 (W): Ignored

When the I²S Ch.1 FIFO becomes full with four received data (four stereo data), the I²S module sets I2SWFIF1 to 1, indicating that the FIFO is completely full. If I²S FIFO whole full interrupts are enabled (I2SWFIE1 = 1), an interrupt request is sent simultaneously to the ITC. The interrupt handler needs to read four stereo data (24 or 16 bits × 2 channels (L & R) × 4) from the FIFO. Then reset I2SWFIF1 by writing 1 at the end of the interrupt handler.

D13 Reserved

D12 I2SOFIF1: I²S Ch.1 FIFO One data Interrupt Flag Bit

Indicates whether the cause of I²S Ch.1 FIFO one data interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

When data (one stereo data) is loaded to the I²S Ch.1 FIFO, the I²S module sets I2SOFIF1 to 1, indicating that the FIFO contains a received data. If one data interrupts are enabled (I2SOFIE1 = 1), an interrupt request is sent simultaneously to the ITC. The interrupt handler needs to fill the FIFO with one stereo data (24 or 16 bits × 2 channels (L & R) × 1). Then reset I2SOFIF1 by writing 1 at the end of the interrupt handler.

D11 Reserved

D10 I2SWEIF0: I²S Ch.0 FIFO Whole Empty Interrupt Flag Bit

Indicates whether the cause of I²S Ch.0 FIFO whole empty interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

When all data (four stereo data) have been read from the I²S Ch.0 FIFO to output, the I²S module sets I2SWEIF0 to 1, indicating that the FIFO is empty. If whole empty interrupts are enabled (I2SWEIE0 = 1), an interrupt request is sent simultaneously to the ITC. The interrupt handler needs to fill the FIFO with four stereo data (24 or 16 bits × 2 channels (L & R) × 4). Then reset I2SWEIF0 by writing 1 at the end of the interrupt handler.

D9 I2SHEIF0: I²S Ch.0 FIFO Half Empty Interrupt Flag Bit

Indicates whether the cause of I²S Ch.0 FIFO half empty interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

When a free space for two stereo data becomes available in the I²S Ch.0 FIFO, the I²S module sets I2SHEIF0 to 1. If half empty interrupts are enabled (I2SHEIE0 = 1), an interrupt request is sent simultaneously to the ITC. The interrupt handler needs to fill the FIFO with two stereo data (24 or 16 bits × 2 channels (L & R) × 2). Then reset I2SHEIF0 by writing 1 at the end of the interrupt handler.

D8 I2SOEIF0: I²S Ch.0 FIFO One Empty Interrupt Flag Bit

Indicates whether the cause of I²S Ch.0 FIFO one empty interrupt has occurred or not.

1 (R): Cause of interrupt has occurred
0 (R): No cause of interrupt has occurred (default)
1 (W): Flag is reset
0 (W): Ignored

When a free space for one stereo data becomes available in the I²S Ch.0 FIFO, the I²S module sets I2SOEIF0 to 1. If one empty interrupts are enabled (I2SOEIE0 = 1), an interrupt request is sent simultaneously to the ITC. The interrupt handler needs to fill the FIFO with one stereo data (24 or 16 bits × 2 channels (L & R) × 1). Then reset I2SOEIF0 by writing 1 at the end of the interrupt handler.

D7 Reserved

D6 I2SWFIE1: I²S Ch.1 FIFO Whole Full Interrupt Enable Bit

Enables or disables I²S Ch.1 FIFO whole full interrupts.

1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting I2SWFIE1 to 1 enables I²S Ch.1 FIFO whole full interrupt requests to the ITC. Setting it to 0 disables interrupts.

D5 Reserved

D4 I2SOFIE1: I²S Ch.1 FIFO One Data Interrupt Enable Bit

Enables or disables I²S Ch.1 FIFO one data interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting I2SOFIE1 to 1 enables I²S Ch.1 FIFO one data interrupt requests to the ITC. Setting it to 0 disables interrupts.

D3 Reserved

D2 I2SWEIE0: I²S Ch.0 FIFO Whole Empty Interrupt Enable Bit

Enables or disables I²S Ch.0 FIFO whole empty interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting I2SWEIE0 to 1 enables I²S Ch.0 FIFO whole empty interrupt requests to the ITC. Setting it to 0 disables interrupts.

D1 I2SHEIE0: I²S Ch.0 FIFO Half Empty Interrupt Enable Bit

Enables or disables I²S Ch.0 FIFO half empty interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting I2SHEIE0 to 1 enables I²S Ch.0 FIFO half empty interrupt requests to the ITC. Setting it to 0 disables interrupts.

D0 I2SOEIE0: I²S Ch.0 FIFO One Empty Interrupt Enable Bit

Enables or disables I²S Ch.0 FIFO one empty interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting I2SOEIE0 to 1 enables I²S Ch.0 FIFO one empty interrupt requests to the ITC. Setting it to 0 disables interrupts.

Note: Always be sure to select only one interrupt mode for each channel by setting one of the interrupt enable bit (Ch.0: I2SWEIE0, I2SHEIE0, or I2SOEIE0, Ch.1: I2SWFIE1, or I2SOFIE1).

I²S Mode Select Register (I2S_MODE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Mode Select Register (I2S_MODE)	**0x30241c** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9	**I2SDMA1**	I²S Ch.1 DMA mode select	1 Dual ch. 0 Single ch.	0	R/W	
		D8	**I2SDMA0**	I²S Ch.0 DMA mode select	1 Dual ch. 0 Single ch.	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**I2SPIN**	I²S pin mode select	1 5 pins 0 8 pins	0	R/W	

D[31:10] Reserved

D9 I2SDMA1: I²S Ch.1 DMA Mode Select Bit

Selects the DMA mode for I²S Ch.1.
1 (R/W): Dual channel DMA mode (16 bits × 2 channels, 32 bits × 2 channels)
0 (R/W): Single channel DMA mode (32 bits × 1 channel) (default)

When the cause of I²S Ch.1 FIFO one data interrupt occurs, the I²S module output a DMA request signal to DMAC Ch.2 and Ch.3. This allows use of DMA transfer for reading received data from the FIFO.

Single channel DMA mode

When storing L-channel and R-channel audio data to one memory area sequentially, use single channel DMA mode. In this mode, either DMAC Ch.2 or Ch.3 can be used for data transfer.

Perform 32-bit data transfer to read both L and R data from the FIFO (address 0x302430) for each DMA request regardless of the data size. Note that 16-bit and 8-bit data transfers cannot be specified in single channel DMA mode.
When reading 16-bit data, transfer both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) from address 0x302430 (fixed address).
When reading 24-bit data, perform data transfer twice. First read 24-bit L-channel data from address 0x302430, and then 24-bit R-channel data from address 0x302430.

Dual channel DMA mode

When storing L-channel and R-channel audio data to different locations, use dual channel DMA mode to transfer L and R data separately from the FIFO. Since DMAC Ch.2 has higher priority than Ch.3, use DMAC Ch.2 to transfer L-channel data first and DMAC Ch.3 to transfer the following R-channel data.
When reading 16-bit data from the FIFO, perform 16-bit transfer for each DMA request. Note that 8-bit and 32-bit transfer cannot be specified. First transfer 16-bit L-channel data from address 0x302430 using DMAC Ch.2, and then 16-bit R-channel data from address 0x302432 using DMAC Ch.3.
When reading 24-bit data from the FIFO, perform 32-bit transfer for each DMA request. Note that 8-bit and 16-bit transfer cannot be specified. First transfer 24-bit L-channel data from address 0x302430 using DMAC Ch.2, and then 24-bit R-channel data from address 0x302430 using DMAC Ch.3.

D8 I2SDMA0: I²S Ch.0 DMA Mode Select Bit

Selects the DMA mode for I²S Ch.0.
1 (R/W): Dual channel DMA mode (16 bits × 2 channels, 32 bits × 2 channels)
0 (R/W): Single channel DMA mode (32 bits × 1 channel) (default)

When the cause of I²S Ch.0 FIFO one empty interrupt occurs, the I²S module output a DMA request signal to DMAC Ch.0 and Ch.1. This allows use of DMA transfer for writing transmit data to the FIFO.

Single channel DMA mode

If L-channel and R-channel audio data are sequentially stored in one memory area, use single channel DMA mode. In this mode, either DMAC Ch.0 or Ch.1 can be used for data transfer.
Perform 32-bit data transfer to write both L and R data to the FIFO (address 0x302420) for each DMA request regardless of the data size. Note that 16-bit and 8-bit data transfers cannot be specified in single channel DMA mode.
When writing 16-bit data, transfer both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) to address 0x302420 (fixed address).
When writing 24-bit data, perform data transfer twice. First write 24-bit L-channel data to address 0x302420, and then 24-bit R-channel data to address 0x302420.

Dual channel DMA mode

If L-channel and R-channel audio data are stored in different locations, use dual channel DMA mode to transfer L and R data separately to the FIFO. Since DMAC Ch.0 has higher priority than Ch.1, use DMAC Ch.0 to transfer L-channel data first and DMAC Ch.1 to transfer the following R-channel data.
When writing 16-bit data to the FIFO, perform 16-bit transfer for each DMA request. Note that 8-bit and 32-bit transfer cannot be specified. First transfer 16-bit L-channel data to address 0x302420 using DMAC Ch.0, and then 16-bit R-channel data to address 0x302422 using DMAC Ch.1.
When writing 24-bit data to the FIFO, perform 32-bit transfer for each DMA request. Note that 8-bit and 16-bit transfer cannot be specified. First transfer 24-bit L-channel data to address 0x302420 using DMAC Ch.0, and then 24-bit R-channel data to address 0x302420 using DMAC Ch.1.

For more information on DMA transfer, see the “DMA Controller (DMAC)” chapter.

D[7:1] Reserved

D0 I2SPIN: I²S Pin Mode Select Bit

Selects the pin mode.
1 (R/W): 5-pin mode
0 (R/W): 8-pin mode (default)

When I2SPIN is set to 0 (default), the I²S module is set to 8-pin mode; when I2SPIN is set to 1, the I²S module is set to 5-pin mode.
In 8-pin mode, all the I2S_MCLKO, I2S_SCLKO, I2S_WSO, I2S_SDO, I2S_MCLKI, I2S_SCLKI, I2S_WSI, and I2S_SDI pins are available. Ch.0 (master output device) and Ch.1 (slave input device) can be used independently. The clock pins of Ch.0 and Ch.1 are not connected inside the IC.
In 5-pin mode, only the I2S_MCLKO, I2S_SCLKO, I2S_WSO, I2S_SDO, and I2S_SDI pins are used for data transfer from/to external I²S slave devices. The Ch.0 clock pins are connected to those of Ch.1 inside the IC and the Ch.0 output clocks are directly fed to Ch.1. This enables the Ch.1 clock pins to be used for other modules when a 5-pin device (e.g., 5-pin DAC) is connected.

I²S Ch.0 FIFO Register (I2S_FIFO0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Ch.0 FIFO Register (I2S_FIFO0)	**0x302420** (32 bits)	D31–0	**I2SFIFO0 [31:0]**	I²S Ch.0 output FIFO	0x0 to 0xffffffff	0x0	W	for 16-bit data (W write) or 24-bit data (W write) 0 when being read.
	0x302420 (16 bits)	D15–0	**I2SFIFO0 [15:0]**	I²S Ch.0 output FIFO	0x0 to 0xffff	0x0	W	for 16-bit data (HW write)
	0x302422 (16 bits)	D15–0	**I2SFIFO0 [31:16]**	I²S Ch.0 output FIFO	0x0 to 0xffff	0x0	W	0 when being read.

D[31:0] I2SFIFO0[31:0]: I²S Ch.0 Output FIFO Bits

This register is used to write output data to the I²S Ch.0 FIFO. (Default: 0x0)
Regardless of the data size, up to four stereo data (24 or 16 bits × 2 channels (L & R) × 4) can be written to the FIFO.
When writing 16-bit data, use a 16-bit or 32-bit memory write instruction. Note that 8-bit memory write instructions cannot be used. When a 16-bit memory write instruction is used, first write 16-bit L-channel data to address 0x302420, and then 16-bit R-channel data to address 0x302422. When a 32-bit memory write instruction is used, write both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) to address 0x302420.
When writing 24-bit data, use a 32-bit memory write instruction. Note that 8-bit and 16-bit memory write instructions cannot be used. First write 24-bit L-channel data to address 0x302420, and then 24-bit R-channel data to address 0x302420.
Both channel data must be written as a pair even if "mono" is selected as the output channel mode.
The sequence for writing four groups of stereo data (24 or 16 bits × 2 channels (L & R) × 4) to the FIFO is shown in Tables 22.8.9 to 22.8.11 (the instructions to write the data to the CPU register and the procedures have been omitted).

Table 22.8.9 Writing 16-bit Data to the FIFO using 16-bit Memory Write Instructions

No.	Memory write instruction	Write details
1	ld.h [0x302420], %rs1	Data1-L[15:0] → 0x302420
2	ld.h [0x302422], %rs2	Data1-R[15:0] → 0x302422
3	ld.h [0x302420], %rs3	Data2-L[15:0] → 0x302420
4	ld.h [0x302422], %rs4	Data2-R[15:0] → 0x302422
5	ld.h [0x302420], %rs5	Data3-L[15:0] → 0x302420
6	ld.h [0x302422], %rs6	Data3-R[15:0] → 0x302422
7	ld.h [0x302420], %rs7	Data4-L[15:0] → 0x302420
8	ld.h [0x302422], %rs8	Data4-R[15:0] → 0x302422

Table 22.8.10 Writing 16-bit Data to the FIFO using 32-bit Write Instructions

No.	Memory write instruction	Write details
1	ld.w [0x302420], %rs12	{Data1-R[15:0], L[15:0]} → 0x302420
2	ld.w [0x302420], %rs34	{Data2-R[15:0], L[15:0]} → 0x302420
3	ld.w [0x302420], %rs56	{Data3-R[15:0], L[15:0]} → 0x302420
4	ld.w [0x302420], %rs78	{Data4-R[15:0], L[15:0]} → 0x302420

(R = D[31:16], L = D[15:0])

Table 22.8.11 Writing 24-bit Data to the FIFO Using 32-bit Write Instructions

No.	Memory write instruction	Write details
1	ld.w [0x302420], %rs1	Data1-L[23:0] → 0x302420
2	ld.w [0x302420], %rs2	Data1-R[23:0] → 0x302420
3	ld.w [0x302420], %rs3	Data2-L[23:0] → 0x302420
4	ld.w [0x302420], %rs4	Data2-R[23:0] → 0x302420
5	ld.w [0x302420], %rs5	Data3-L[23:0] → 0x302420
6	ld.w [0x302420], %rs6	Data3-R[23:0] → 0x302420
7	ld.w [0x302420], %rs7	Data4-L[23:0] → 0x302420
8	ld.w [0x302420], %rs8	Data4-R[23:0] → 0x302420

Notes:
- Do not perform byte access to write 16-bit data.
  Do not perform byte access or half-word access to write 24-bit data.
- When using the DMAC to transfer 24-bit data, or when using single channel DMA mode to transfer 16-bit data, be sure to set the transfer data unit to word size. Transfer in byte size or half-word size is not supported.
- When using dual channel DMA mode to transfer 16-bit data, be sure to set the transfer data unit to half-word size. Transfer in byte size or word size is not supported.

I²S Ch.1 FIFO Register (I2S_FIFO1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Ch.1 FIFO Register (I2S_FIFO1)	**0x302430** (32 bits)	D31–0	**I2SFIFO1 [31:0]**	I²S Ch.1 input FIFO	0x0 to 0xffffffff	0x0	R	for 16-bit data (W read) or 24-bit data (W read) 0 when being read.
	0x302430 (16 bits)	D15–0	**I2SFIFO1 [15:0]**	I²S Ch.1 input FIFO	0x0 to 0xffff	0x0	R	for 16-bit data (HW read)
	0x302432 (16 bits)	D15–0	**I2SFIFO1 [31:16]**	I²S Ch.1 input FIFO	0x0 to 0xffff	0x0	R	0 when being read.

D[31:0] I2SFIFO1[31:0]: I²S Ch.1 input FIFO Bits

This register is used to read received data from the I²S Ch.1 FIFO. (Default: 0x0)
Up to four stereo data (24 or 16 bits × 2 channels (L & R) × 4) can be stored to the FIFO.
When reading 16-bit data, use a 16-bit or 32-bit memory read instruction. Note that 8-bit memory read instructions cannot be used. When a 16-bit memory read instruction is used, first read 16-bit L-channel data from address 0x302430, and then 16-bit R-channel data from address 0x302432. When a 32-bit memory read instruction is used, read both L-channel data (low-order 16-bits) and R-channel data (high-order 16-bits) from address 0x302430.
When reading 24-bit data, use a 32-bit memory read instruction. Note that 8-bit and 16-bit memory read instructions cannot be used. First read 24-bit L-channel data from address 0x302420, and then 24-bit R-channel data from address 0x302420.
Both channel data must be read as a pair.
The sequence for reading four groups of stereo data (24 or 16 bits × 2 channels (L & R) × 4) from the FIFO is shown in Tables 22.8.12 to 22.8.14 (the instructions to store the read data to memory and the procedures have been omitted).

Table 22.8.12 Reading 16-bit Data from the FIFO using 16-bit Memory Read Instructions

No.	Memory read instruction	Read details
1	ld.h %rd1, [0x302430]	0x302430 → Data1-L[15:0]
2	ld.h %rd2, [0x302432]	0x302432 → Data1-R[15:0]
3	ld.h %rd3, [0x302430]	0x302430 → Data2-L[15:0]
4	ld.h %rd4, [0x302432]	0x302432 → Data2-R[15:0]
5	ld.h %rd5, [0x302430]	0x302430 → Data3-L[15:0]
6	ld.h %rd6, [0x302432]	0x302432 → Data3-R[15:0]
7	ld.h %rd7, [0x302430]	0x302430 → Data4-L[15:0]
8	ld.h %rd8, [0x302432]	0x302432 → Data4-R[15:0]

Table 22.8.13 Reading 16-bit Data from the FIFO using 32-bit Read Instructions

No.	Memory read instruction	Read details
1	ld.w %rd12, [0x302430]	0x302430 → {Data1-R[15:0], L[15:0]}
2	ld.w %rd34, [0x302430]	0x302430 → {Data2-R[15:0], L[15:0]}
3	ld.w %rd56, [0x302430]	0x302430 → {Data3-R[15:0], L[15:0]}
4	ld.w %rd78, [0x302430]	0x302430 → {Data4-R[15:0], L[15:0]}

(R = D[31:16], L = D[15:0])

Table 22.8.14 Reading 24-bit Data from the FIFO Using 32-bit Read Instructions

No.	Memory read instruction	Read details
1	ld.w %rd1, [0x302430]	0x302430 → Data1-L[23:0]
2	ld.w %rd2, [0x302430]	0x302430 → Data1-R[23:0]
3	ld.w %rd3, [0x302430]	0x302430 → Data2-L[23:0]
4	ld.w %rd4, [0x302430]	0x302430 → Data2-R[23:0]
5	ld.w %rd5, [0x302430]	0x302430 → Data3-L[23:0]
6	ld.w %rd6, [0x302430]	0x302430 → Data3-R[23:0]
7	ld.w %rd7, [0x302430]	0x302430 → Data4-L[23:0]
8	ld.w %rd8, [0x302430]	0x302430 → Data4-R[23:0]

Notes:
- Do not perform byte access to read 16-bit data.
  Do not perform byte access or half-word access to read 24-bit data.
- When using the DMAC to transfer 24-bit data, or when using single channel DMA mode to transfer 16-bit data, be sure to set the transfer data unit to word size. Transfer in byte size or half-word size is not supported.
- When using dual channel DMA mode to transfer 16-bit data, be sure to set the transfer data unit to half-word size. Transfer in byte size or word size is not supported.

22.9 I²S Clock Settings

This section explains how to configure the I2S_MCLKO, I2S_WSO, and I2S_SCLKO clocks.

Generating I2S_MCLKO from the internal clock

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_WSO) is in sync with the master clock (I2S_MCLKO), so the following equation is formulated:

$$\frac{f_{I2S\_MCLKO}}{f_{I2S\_WSO}} = \text{Integer}$$

where f_{I2S_MCLKO} is the output master clock (I2S_MCLKO) frequency and f_{I2S_WSO} is the sampling clock (I2S_WSO) frequency.

$$f_{I2S\_MCLKO} = \frac{48\ \text{MHz}}{\text{MCLKDIV}[5:0] + 1} \qquad \text{(eq1)}$$

$$f_{I2S\_WSO} = \frac{48\ \text{MHz}}{(\text{BCLKDIV0}[7:0] + 1) \times 2 \times (\text{WSCLKCYC0}[4:0] + 16) \times 2} \qquad \text{(eq2)}$$

$$\frac{(BCLKDIV0[7:0] + 1) \times 2 \times (WSCLKCYC0[4:0] + 16) \times 2}{MCLKDIV[5:0] + 1} = \text{Integer} \qquad \text{(eq3)}$$

Table 22.9.1 I2S_MCLKO (Master Clock) Settings

MCLKDIV[5:0]	PCLK division ratio
0x3f	1/64
0x3e	1/63
:	:
0x2	1/3
0x1	1/2
0x0	Reserved

Table 22.9.2 I2S_SCLKO (Bit Clock) Settings

BCLKDIV0[7:0]	PCLK division ratio
0xff	1/512
0xfe	1/510
:	:
0x2	1/6
0x1	1/4
0x0	1/2

Table 22.9.3 Sample Clock Period Settings

WSCLKCYC0[4:0]	Sample clock period (number of bit clock cycles)
0x1f–0x11	Reserved
0x10	32 clocks
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.

Table 22.9.4 List of Sample Clock Frequencies

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	SysClk	BCLKDIV0[7:0]	WSCLKCYC0[4:0]																
2	[kHz]		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	48000	0	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	375.00
4		1	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	187.50
5		2	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	125.00
6		3	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	93.75
7		4	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	75.00
8		5	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	62.50
9		6	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	53.57
10		7	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	46.88
11		8	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	41.67
12		9	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	37.50
13		10	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	34.09
14		11	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	31.25
15		12	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	28.85
16		13	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	26.79
17		14	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	25.00
18		15	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	23.44
19		16	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	22.06
20		17	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	20.83
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
257		254	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	1.47
258		255	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	1.46

Cell [A3]: System clock frequency (48000 kHz)
Cells [B3:B258]: BCLKDIV0[7:0] settings (0 to 255)
Cells [C2:S2]: WSCLKCYC0[4:0] settings (0 to 16)
Cells [C3:S258]: f_{I2S_WSO} calculated by Equation 2 (eq2) according to the BCLKDIV0[7:0] and WSCLKCYC0[4:0] settings

Cell [C3] = $A$3/(($B3+1)*2*(C$2+16)*2) Cell [S3] = $A$3/(($B3+1)*2*(S$2+16)*2)
:
Cell [C258] = $A$3/(($B258+1)*2*(C$2+16)*2) Cell [S258] = $A$3/(($B258+1)*2*(S$2+16)*2)

If you use another system clock frequency, enter the frequency in kHz to Cell [A3]. Cells [C3:S258] will be corrected according to the entered value.

Find "44.1 (kHz)" or an approximate value from the table. You may choose "44.12" in Cell [D18].
You may get the BCLKDIV0[7:0] and WSCLKCYC0[4:0] values as 15 and 1, respectively.

Substituting these values in Equation 3 (eq3) yields the MCLKDIV[5:0] values.

$$\frac{(\boldsymbol{15}+1)\times 2\times(\boldsymbol{1}+16)\times 2}{\mathrm{MCLKDIV}[5:0]+1} = \mathrm{Integer}$$

The table below is made from Equitation 3 (eq3) using Excel.

Table 22.9.5 MCLKDIV[5:0] Valid Values

	U	V
1	BCLKDIV0[7:0]	WSCLKCYC0[4:0]
2	15	1
3		
4	MCLKDIV[5:0]	Results
5	1	Integer
6	2	–
7	3	Integer
:	:	–
11	7	Integer
:	:	–
19	15	Integer
20	16	Integer
:	:	–
35	31	Integer
36	32	–
37	33	Integer
:	:	–
67	63	Integer

Cell [U2]: BCLKDIV0[7:0] setting (15)
Cell [V2]: WSCLKCYC0[4:0] setting (1)

Cells [V5:V68]: Results of Equation 3 (eq3)

Cell [V5] = IF(MOD(($U$2+1)*2*($V$2+16)*2, (U5+1))=0, “Integer”, “-”)

:

Cell [V67] = IF(MOD(($U$2+1)*2*($V$2+16)*2, (U67+1))=0, “Integer”, “-”)

Enter the selected BCLKDIV0[7:0] and WSCLKCYC0[4:0] values to Cells U2 and V2, respectively. “Integer” appears in the cell corresponding to the MCLKDIV[5:0] value that can be set.

MCLKDIV[5:0] = 1, 3, 7, 15, 16, 31, 33, 63

Table 22.9.6 Master Clock Frequency

MCLKDIV[5:0]	f_{I2S_MCLKO}	
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)

Select an appropriate MCLKDIV[5:0] value according to the value listed in the above table.

Using external master clock (I2S_MCLKI 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 audio data sampling rate is 44.1 kHz (= fs) and the master clock is 320 fs.

Master Clock Frequency = 320 × 44.1 kHz = 14.112 MHz

Find “320 fs” from the table below. The following two possible settings are valid.

1) BCLKDIV0[7:0] = 4, WSCLKCYC0[4:0] = 0 (16 cycles)
2) BCLKDIV0[7:0] = 3, WSCLKCYC0[4:0] = 4 (20 cycles)

The sampling clock (I2S_WSO) frequency is 44.1 kHz.

Table 22.9.7 External Master Clock Frequency

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	BCLKDIV0[7:0]	WSCLKCYC0[4:0]																
2		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	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	128 fs
4	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	256 fs
5	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	384 fs
6	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	512 fs
7	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	640 fs
8	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	768 fs
9	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	896 fs
10	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	1024 fs
11	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	1152 fs
12	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	1280 fs
13	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	1408 fs
14	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	1536 fs
15	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	1664 fs
16	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	1792 fs
17	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	1920 fs
18	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	2048 fs

Cells [A3:A18]: BCLKDIV0[7:0] settings (0 to 15)
Cells [B2:R2]: WSCLKCYC0[4:0] settings (0 to 16)
Cells [B3:R18]: (BCLKDIV0[7:0] + 1) × 2 × (WSCLKCYC0[4:0] + 16) × 2 [fs]

Cell [B3] = ($A3+1)*2*(B$2+16)*2 Cell [R3] = ($A3+1)*2*(R$2+16)*2)

: :

Cell [C18] = ($A18+1)*2*(B$2+16)*2 Cell [R18] = ($A18+1)*2*(R$2+16)*2)

23 Remote Controller (REMC)

23.1 REMC Module Overview

The S1C33L27 incorporates a remote controller (REMC) module for generating infrared remote control communication signals.
The following shows the features of the REMC module:

- Supports input and output infrared remote control communication signals.
- Incorporates a carrier generator for generating a carrier signal using the prescaler output clock.
- Incorporates an 8-bit down-counter for counting the transfer data length.
- Incorporates a modulator for generating transmission data of the specified carrier length.
- Incorporates an edge detector for detecting input signal rising and falling edges.
- Can generate counter underflow interrupts indicating that the specified data length has been transmitted and input rising/falling edge detection interrupts for data receive processing.

Figure 23.1.1 shows the configuration of the REMC module.

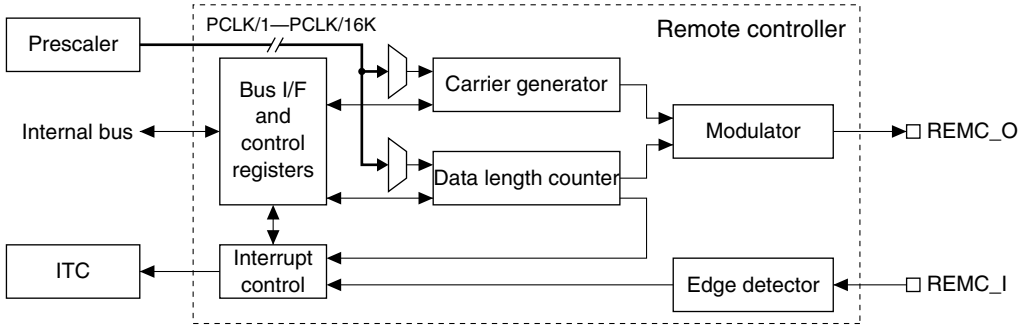


Figure 23.1.1 REMC Module Configuration

23.2 REMC Input/Output Pins

Table 23.2.1 lists the REMC input/output pins.

Table 23.2.1 List of REMC Pins

Pin name	I/O	Qty	Function
REMC_I	I	1	Remote control receive data input pin Inputs receive data.
REMC_O	O	1	Remote control transmit data output pin Outputs modulated remote control transmit data.

The REMC input/output pins (REMC_I, REMC_O) are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as REMC input/output pins. For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

23.3 Carrier Generation

The REMC module incorporates a carrier generator that generates a carrier signal for transmission in accordance with the clock set by software and carrier H and L section lengths.
The prescaler (PSC Ch.2) output clock is used for the carrier signal generation clock. The prescaler generates 15 different clocks, dividing the PCLK clock by 1 to 16K. One of these clocks is selected by CGCLK[3:0]/REMC_CFG register.

Table 23.3.1 Carrier Generation Clock (PCLK Division Ratio) Selection

CGCLK[3:0]	Division ratio	CGCLK[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

For more information on prescaler control, see the "Prescaler (PSC)" chapter.

Note: The prescaler must be run before the REMC module can operate.

The carrier H and L section lengths are set by REMCH[5:0]/REMC_CAR register and REMCL[5:0]/REMC_CAR register, respectively. Set a value corresponding to the number of clock (selected as above) cycles + 1 to these registers.
The carrier H and L section lengths can be calculated as follows:

$$\text{Carrier H section length} = \frac{\text{REMCH} + 1}{\text{clk\_in}} \text{ [s]}$$

$$\text{Carrier L section length} = \frac{\text{REMCL} + 1}{\text{clk\_in}} \text{ [s]}$$

REMCH: Carrier H section length data value
REMCL: Carrier L section length data value
clk_in: Prescaler (PSC Ch.2) output clock frequency

The carrier signal is generated from these settings as shown in Figure 23.3.1.

Example: CGCLK[3:0] = 0x2 (PCLK/4), REMCH[5:0] = 2, REMCL[5:0] = 1

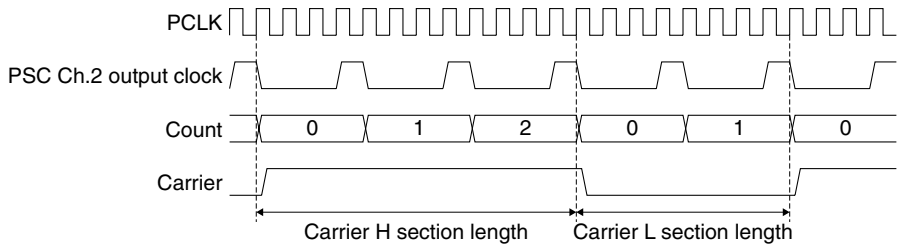


Figure 23.3.1 Carrier Signal Generation

23.4 Data Length Counter Clock Settings

The data length counter is an 8-bit counter for setting data lengths when transmitting data.
When a value corresponding to the data pulse width is written during data transmission, the data length counter begins counting down from that value and stops after generating an underflow interrupt cause when the counter reaches 0. The subsequent transmit data is set using this interrupt.

This counter is also used for data receiving, enabling measurement of the received data length. Interrupts can be generated at the input signal rising or falling edges when receiving data. The data pulse length can be obtained from the difference between data pulse edges by setting the data length counter to 0xff using the interrupt when the input changes and by reading out the count value when a subsequent interrupt occurs due to input changes.

This data length counter count clock also uses a prescaler output clock and can select one of 15 different types. The prescaler output clock is selected by LCCLK[3:0]/REMC_CFG register provided separately to the carrier generation clock select bits.

Table 23.4.1 Data Length Counter Clock (PCLK Division Ratio) Selection

LCCLK[3:0]	Division ratio	LCCLK[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

The data length counter can count up to 256. The count clock should be selected to ensure that the data length fits within this range.

23.5 Data Transfer Control

Make the following settings before starting data transfers.
(1) Configure the carrier signal. (See Section 23.3.)
(2) Select the data length counter clock. (See Section 23.4.)
(3) Set the interrupt conditions. (See Section 23.6.)

Note: Make sure the REMC module is halted (REMEN/REMC_CFG register = 0) before changing the above settings.

Data transmission control

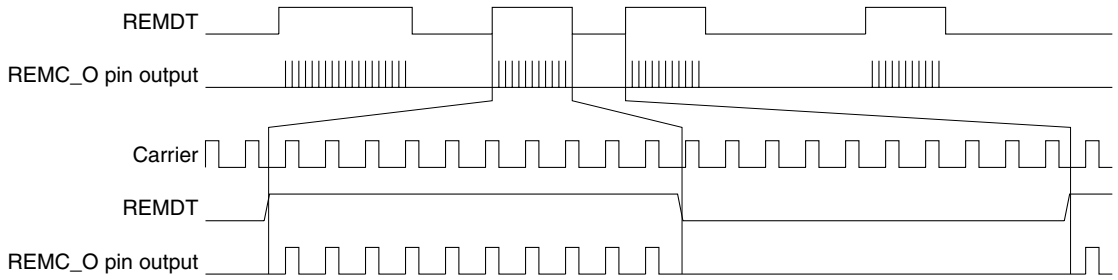


Figure 23.5.1 Data Transmission

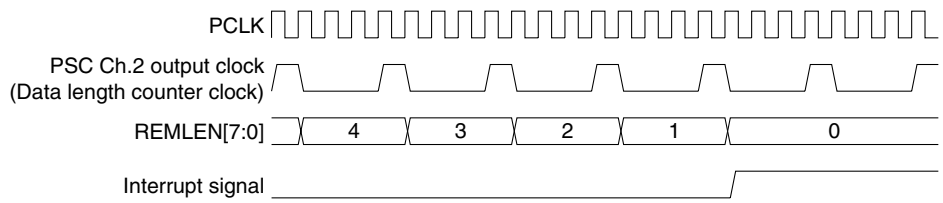


Figure 23.5.2 Underflow Interrupt Generation Timing

(1) Data transmit mode setting
Set REMC to transmit mode by writing 0 to REMMD/REMC_CFG register.

(2) Enabling data transmission
Enable REMC operation by setting REMEN/REMC_CFG register to 1. This initiates REMC transmission.

Set REMDT/REMC_LCNT register to 0 and REMLEN[7:0]/REMC_LCNT register to 0x0 before setting REMEN to 1 to prevent unnecessary data transmission.

(3) Transmission data setting
Set the data to be transmitted (High or Low) to REMDT/REMC_LCNT register.
Setting REMDT to 1 outputs High; setting it to 0 outputs Low from the REMC_O pin after being modulated by the carrier signal.

(4) Data pulse length setting

Set the value corresponding to the data pulse length (High or Low section) to REMLEN[7:0]/REMC_LCNT register to set to the data length counter.
Given below is the value to which the data length counter is set:

Setting value = Data pulse length (seconds) × Prescaler output clock frequency (Hz)

The data length counter starts counting down from the value written using the prescaler output clock selected.
A cause of underflow interrupt occurs when the data length counter value reaches 0. If the interrupt is enabled, an REMC interrupt request is output to the interrupt controller (ITC). The data length counter stops counting at the same time with the counter value 0 maintained.

(5) Interrupt handling

To transmit the subsequent data, set the subsequent data (Step 3) and set the data pulse length (Step 4) in the interrupt handler routine executed by the data length counter underflow.

(6) Terminating data transmission

To terminate data transmission, set REMEN to 0 after the final data transmission has completed (after an underflow interrupt has occurred).

Data reception control

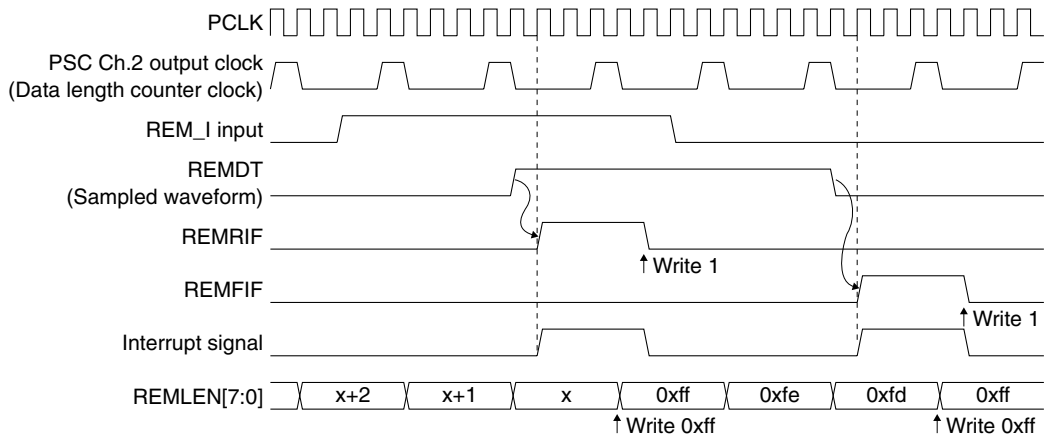


Figure 23.5.3 Data Reception

(1) Data receive mode setting

Set REMC to receive mode by writing 1 to REMMD/REMC_CFG register.

(2) Enabling data reception

Enable REMC operation by setting REMEN/REMC_CFG register to 1. This initiates REMC reception (input edge detection).
REMC detects an input transition (signal rising or falling edges) by sampling the input signal from the REMC_I pin using the prescaler output clock selected for carrier generation. If a signal edge is detected, a cause of rising or falling edge interrupt is generated. An REMC interrupt request is output to the ITC if the interrupt is enabled. Rising edge and falling edge interrupts can be individually enabled or disabled.

Note that if the signal level after the input has changed is not detected for at least two continuous sampling clock cycles, the input signal transition is interpreted as noise, and no rising or falling edge interrupt is generated.

(3) Interrupt handling

When a rising edge or falling edge interrupt occurs, write 0xff to REMLEN[7:0]/REMC_LCNT register in the interrupt handler routine to set the value to the data length counter.

The data length counter starts counting down using the selected prescaler output clock from the value written.

The data received can be read out from REMDT/REMC_LCNT register.

The subsequent falling or rising edge interrupt is generated at the termination of the data pulse. Read the data length counter at that point. The data length can be calculated from the difference between 0xff and the value read. To receive the subsequent data, set the data length counter to 0xff once again, then wait for the subsequent interrupt.

If the data length counter becomes 0 after being set to 0xff without the occurrence of an edge interrupt, either no more data is left or a receive error has occurred. Data length counter underflow interrupts are generated even when receiving data and should be used for terminate/error handling.

(4) Terminating data reception

To terminate data reception, write 0 to REMEN after the final data has been received.

23.6 REMC Interrupts

The REMC module includes a function for generating the following three different types of interrupts.
- Underflow interrupt
- Rising edge interrupt
- Falling edge interrupt

The REMC module outputs one interrupt signal shared by the three interrupt causes above to the interrupt controller (ITC). To identify the cause of interrupt occurred, check the interrupt flag status in the REMC module.

Underflow interrupt

Generated when the data length counter has counted down to 0, this interrupt cause sets the interrupt flag REMUIF/REMC_INT register inside the REMC to 1.
When data is being transmitted, the underflow interrupt indicates that the specified data length has been transmitted. When receiving data, the underflow interrupt indicates that data has been received or a receive error has occurred.
To use this interrupt, set REMUIE/REMC_INT register to 1. If REMUIE is set to 0 (default), the interrupt request attributable to this cause will not be sent to the ITC.
When REMUIF is set to 1, REMC outputs an interrupt request to the ITC. An interrupt will be generated if the ITC and C33 PE Core interrupt conditions are met.
REMUIF should be inspected in the REMC interrupt handler routine to determine whether the REMC interrupt is attributable to data length counter underflow.
The interrupt cause should be cleared in the interrupt handler routine by resetting (writing 1 to) REMUIF.

Rising edge interrupt

Generated when the REMC_I pin input signal changes from Low to High, this interrupt cause sets the interrupt flag REMRIF/REMC_INT register to 1 within the REMC.
By running the data length counter between this interrupt and a falling edge interrupt when data is being received, the received data pulse width can be calculated from that count value.
To use this interrupt, set REMRIE/REMC_INT register to 1. If REMRIE is set to 0 (default), the interrupt request attributable to this cause will not be sent to the ITC.
When REMRIF is set to 1, REMC outputs an interrupt request to the ITC. An interrupt will be generated if the ITC and C33 PE Core interrupt conditions are met.
REMRIF should be inspected in the REMC interrupt handler routine to determine whether the REMC interrupt is attributable to input signal rising edge.
The interrupt cause should be cleared in the interrupt handler routine by resetting (writing 1 to) REMRIF.

Falling edge interrupt

Generated when the REMC_I pin input signal changes from High to Low, this interrupt cause sets the interrupt flag REMFIF/REMC_INT register to 1 within the REMC.
By running the data length counter between this interrupt and a rising edge interrupt when data is being received, the received data pulse width can be calculated from that count value.

To use this interrupt, set REMFIE/REMC_INT register to 1. If REMFIE is set to 0 (default), the interrupt request attributable to this cause will not be sent to the ITC.
When REMFIF is set to 1, REMC outputs an interrupt request to the ITC. An interrupt will be generated if the ITC and C33 PE Core interrupt conditions are met.
REMFIF should be inspected in the REMC interrupt handler routine to determine whether the REMC interrupt is attributable to input signal falling edge.
The interrupt cause should be cleared in the interrupt handler routine by resetting (writing 1 to) REMFIF.

For more information on interrupt processing, see the "Interrupt Controller (ITC)" chapter.

23.7 Control Register Details

Table 23.7.1 List of REMC Registers

Address	Register name		Function
0x301500	REMC_CFG	REMC Configuration Register	Controls the clock and data transfer.
0x301502	REMC_CAR	REMC Carrier Length Setup Register	Sets the carrier H/L section lengths.
0x301504	REMC_LCNT	REMC Length Counter Register	Sets the transmit/receive data length.
0x301506	REMC_INT	REMC Interrupt Control Register	Controls interrupts.

The REMC registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

REMC Configuration Register (REMC_CFG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Configuration Register (REMC_CFG)	**0x301500** (16 bits)	D15–12	**CGCLK[3:0]**	Carrier generator clock division ratio select (Prescaler output clock)	CGCLK[3:0] / LCCLK[3:0]: 0xf reserved; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
		D11–8	**LCCLK[3:0]**	Length counter clock division ratio select (Prescaler output clock)		0x0	R/W	
		D7–2	–	reserved	–	–	–	0 when being read.
		D1	**REMMD**	REMC mode select	1 Receive 0 Transmit	0	R/W	
		D0	**REMEN**	REMC enable	1 Enable 0 Disable	0	R/W	

D[15:12] CGCLK[3:0]: Carrier Generator Clock Division Ratio Select Bits

Selects a carrier generation clock from the 15 prescaler (PSC Ch.2) output clocks (PCLK division ratio).

Table 23.7.2 Carrier Generation Clock (PCLK Division Ratio) Selection

CGCLK[3:0]	Division ratio	CGCLK[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

D[11:8] LCCLK[3:0]: Length Counter Clock Division Ratio Select Bits

Selects a data length counter clock from the 15 prescaler (PSC Ch.2) output clocks (PCLK division ratio).

Table 23.7.3 Data Length Counter Clock (PCLK Division Ratio) Selection

LCCLK[3:0]	Division ratio	LCCLK[3:0]	Division ratio
0xf	Reserved	0x7	1/128
0xe	1/16384	0x6	1/64
0xd	1/8192	0x5	1/32
0xc	1/4096	0x4	1/16
0xb	1/2048	0x3	1/8
0xa	1/1024	0x2	1/4
0x9	1/512	0x1	1/2
0x8	1/256	0x0	1/1

(Default: 0x0)

Note: The clock should be set only while the REMC module is stopped (REMEN = 0).

D[7:2] Reserved

D1 REMMD: REMC Mode Select Bit

Selects the transfer direction.
1 (R/W): Reception
0 (R/W): Transmission (default)

D0 REMEN: REMC Enable Bit

Enables or disables data transfer by the REMC module.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Setting REMEN to 1 starts transmission or receiving in accordance with REMMD settings.
Setting REMEN to 0 disables REMC module operations.

REMC Carrier Length Setup Register (REMC_CAR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Carrier Length Setup Register (REMC_CAR)	0x301502 (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13–8	REMCL[5:0]	Carrier L length setup	0x0 to 0x3f	0x0	R/W	
		D7–6	–	reserved	–	–	–	0 when being read.
		D5–0	REMCH[5:0]	Carrier H length setup	0x0 to 0x3f	0x0	R/W	

D[15:14] Reserved

D[13:8] REMCL[5:0]: Carrier L Length Setup Bits

Sets the carrier signal L section length. (Default: 0x0)
Specify a value corresponding to the number of carrier generation clock cycles selected by CGCLK[3:0]/ REMC_CFG register + 1. Calculate carrier L section length as follows:

$$\text{Carrier L section length} = \frac{\text{REMCL} + 1}{\text{clk\_in}} \text{ [s]}$$

REMCL: REMCL[5:0] setting
clk_in: Prescaler (PSC Ch.2) output clock frequency

The H section length is specified by REMCH[5:0]. The carrier signal is generated from these settings as shown in Figure 23.7.1.

D[7:6] Reserved

D[5:0] REMCH[5:0]: Carrier H Length Setup Bits

Sets the carrier signal H section length. (Default: 0x0)
Specify a value corresponding to the number of carrier generation clock cycles selected by CGCLK[3:0]/ REMC_CFG register + 1. Calculate carrier H section length as follows:

$$\text{Carrier H section length} = \frac{\text{REMCH} + 1}{\text{clk\_in}} \text{ [s]}$$

REMCH: REMCH[5:0] setting
clk_in: Prescaler (PSC Ch.2) output clock frequency

The L section length is specified by REMCL[5:0]. The carrier signal is generated from these settings as shown in Figure 23.7.1.

Example: CGCLK[3:0] = 0x2 (PCLK/4), REMCH[5:0] = 2, REMCL[5:0] = 1

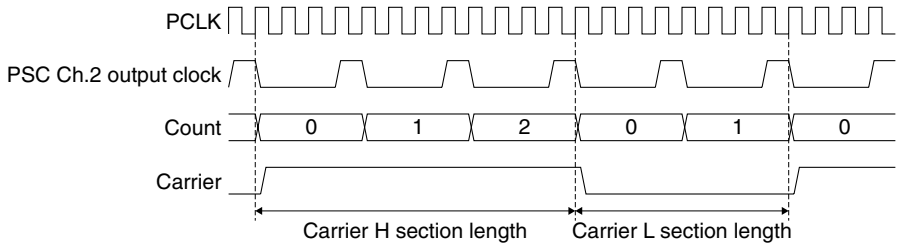


Figure 23.7.1 Carrier Signal Generation

REMC Length Counter Register (REMC_LCNT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Length Counter Register (REMC_LCNT)	**0x301504** (16 bits)	D15–8	**REMLEN[7:0]**	Transmit/receive data length count (down counter)	0x0 to 0xff	0x0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**REMDT**	Transmit/receive data	1 1 (H) 0 0 (L)	0	R/W	

D[15:8] REMLEN[7:0]: Transmit/Receive Data Length Count Bits

Sets the data length counter value and starts counting. (Default: 0x0)
The counter stops when it reaches 0 and generates a cause of underflow interrupt.

For data transmission

Set the transmit data length for data transmission.
When a value corresponding to the data pulse width is written, the data length counter starts counting down from that value. The counter stops counting and generates a cause of underflow interrupt when it reaches 0. Set the subsequent transmit data using this interrupt.

For data receiving

Interrupts can be generated at the input signal rising or falling edges when receiving data. The data pulse length can be obtained from the difference between 0xff set to the data length counter using the interrupt when the input changes and the count value read out when the next interrupt occurs due to an input change.

D[7:1] Reserved

D0 REMDT: Transmit/Receive Data Bit

Sets the transmit data for data transmission. Receive data can be read when receiving data.
1 (R/W): 1 (H)
0 (R/W): 0 (L) (default)

If REMEN/REMC_CFG register is set to 1, the REMDT setting is modulated by the carrier signal for data transmission and output from the REMC_O pin. For data receiving, this bit is set to the value corresponding to the signal level of the data pulse input.

REMC Interrupt Control Register (REMC_INT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Interrupt Control Register (REMC_INT)	**0x301506** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10	**REMFIF**	Falling edge interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D9	**REMRIF**	Rising edge interrupt flag		0	R/W	
		D8	**REMUIF**	Underflow interrupt flag		0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2	**REMFIE**	Falling edge interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**REMRIE**	Rising edge interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**REMUIE**	Underflow interrupt enable	1 Enable 0 Disable	0	R/W	

This register controls the data length counter underflow, input signal rising edge, and input signal falling edge interrupts. The interrupt flag is set to 1 when the data length counter underflows, or when an input signal rising edge or falling edge is detected. If the corresponding interrupt enable bit has been set to 1, the REMC outputs an interrupt request signal to the ITC at the same time. An interrupt will be generated if the ITC and C33 PE Core interrupt conditions are met. When an REMC interrupt occurs, check the interrupt flag status in this register to identify the cause of interrupt occurred. If the interrupt enable bit is set to 0, the interrupt is disabled.

Notes: • To prevent interrupt recurrences, the REMC module interrupt flag must be reset in the interrupt handler routine after an REMC interrupt has occurred.

• To prevent generating unnecessary interrupts, reset the interrupt flag before enabling interrupts by the interrupt enable bit.

D[15:11] Reserved

D10 REMFIF: Falling Edge Interrupt Flag Bit

Indicates 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): Ignored

REMFIF is set to 1 at the input signal falling edge. REMFIF is reset to 0 by writing 1.

D9 REMRIF: Rising Edge Interrupt Flag Bit

Indicates 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): Ignored

REMRIF is set to 1 at the input signal rising edge. REMRIF is reset to 0 by writing 1.

D8 REMUIF: Underflow Interrupt Flag Bit

Indicates 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): Ignored

REMUIF is set to 1 when a data length counter underflow occurs. REMUIF is reset to 0 by writing 1.

D[7:3] Reserved

D2 REMFIE: Falling Edge Interrupt Enable Bit

Enables or disables input signal falling edge interrupts.

1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

D1 REMRIE: Rising Edge Interrupt Enable Bit

Enables or disables input signal rising edge interrupts.

1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

D0 REMUIE: Underflow Interrupt Enable Bit

Enables or disables data length counter underflow interrupts.

1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

24 Card Interface (CARD)

24.1 CARD Module Overview

The S1C33L27 includes a card interface (CARD) module equipped with a hardware Reed-Solomon Code decoding accelerator to reduce Error Correction processing loads. The CARD module serves as the C language interface required to use the hardware Reed-Solomon Code decoding accelerator.

- Generates the #NAND_RD and #NAND_WR signals.
  (Use general-purpose input/output ports to control the signals specific to NAND Flash or SmartMedia card.)
- Supports 8-bit NAND Flash and SmartMedia card interface.
- The data and address signals of the device can be connected to the external bus.
- Includes a Reed-Solomon (hereafter called RS) codec to support an error correction function in the 8-bit interface.
  - Supports automatic error detection. (Error Detection Code)
  - Supports error correction. (Error Correction Code)
  - Supports two kinds of RS algorithm in 2-symbol ECC and 10-symbol ECC.
  - Supports variable length encoding and decoding in both two RS algorithms.

25 SD/MMC interface (SD_MMC)

25.1 SD_MMC Module Overview

The S1C33L27 is equipped with an SD/MMC interface (SD_MMC) module that provides the interface functions for multimedia cards (MMC) and SD/SDHC memory cards.
The following shows the main features of the SD/MMC interface:

- Complies with the SD Memory Card Physical Layer Specification Version 2.00.
- Complies with the MMC System Specification Version 2.2.
- Variable clock rate: Max. 30 MHz
- Supports 4-bit (wide bus) and 1-bit (standard bus) SD bus interface.
- Transfer data format: MSB first
- Internal CRC7 and CRC16 generators/error checkers.
- Internal 2-word transmit/receive FIFO.
- Can generate transmit FIFO half-empty, receive FIFO half-full, and transfer error interrupts.
- Can issue DMA requests using the transmit FIFO half-empty and receive FIFO half-full interrupt causes.

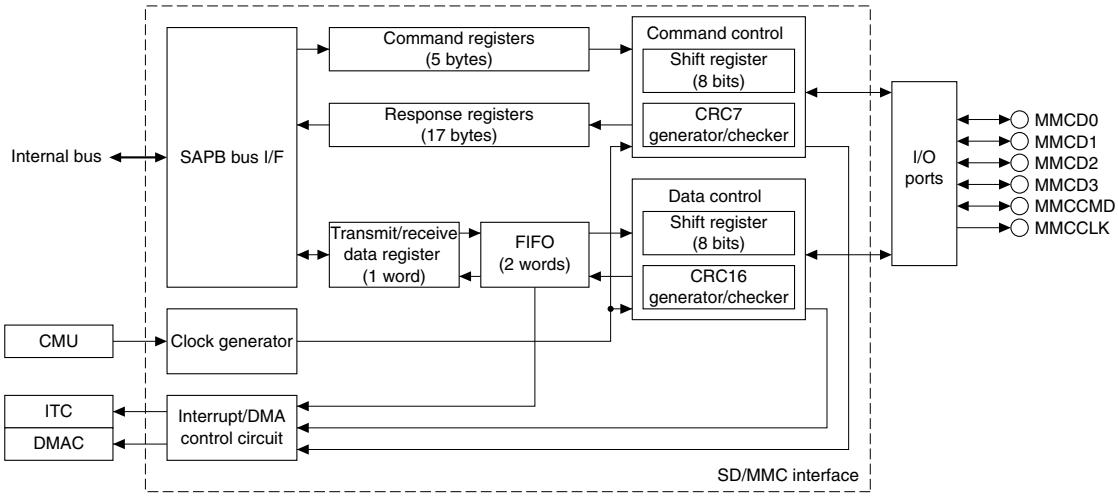


Figure 25.1.1 SD/MMC interface Configuration

Note: The SD_MMC module does not support SPI mode. To transfer data in SPI mode, use the USI module.

26 I/O Ports (GPIO)

26.1 GPIO Module Overview

The S1C33L27 includes general-purpose I/O ports that allow software to switch input/output direction. These share internal peripheral module input/output pins, but pins not used for peripheral modules can be used as general-purpose I/O ports.
The following shows the features of the GPIO module:

- Maximum 91 I/O ports (P0[7:0], P1[7:0], P2[1:0], P3[6:0], P4[2:0], P5[6:0], P60, P8[3:0], P9[7:0], PA[7:0], PB[7:0], PC[7:0], PD[7:0], PE[5:0], PF[4:0]) and eight input ports (P7[7:0]) are available in the PFBGA12U-180 package or chip model.
- Maximum 72 I/O ports (P0[7:0], P1[7:0], P2[1:0], P3[6:0], P4[2:0], P5[6:0], P60, P8[3:0], P9[7:0], PA[7:0], PB[7:0], PC[7:0]) and four input ports (P70/72/74/76) are available in the TQFP24-144pin or QFP20-144pin package model.
- Maximum 56 I/O ports (P0[7:0], P1[7:0], P2[1:0], P3[6:0], P4[2:0], P5[6:0], P60, P8[3:0], P9[7:0], PC[7:0]) and four input ports (P70/72/74/76) are available in the TQFP15-128pin package model.

  * The GPIO ports are shared with other peripheral function pins (UART, timer inputs/outputs etc.). Therefore, the number of GPIO ports depends on the peripheral functions used.

- Can generate maximum 8 port input interrupts selected from 64 ports via software.
- Can generate key input interrupts from 32 ports.
- Interrupt input signal conditions (level or edge trigger, and polarity) can be specified.
- The input interrupt circuit includes chattering filters.
- All port provide a port function select bit to configure the pin function (for GPIO or peripheral functions).

Figure 26.1.1 shows the I/O port configuration.

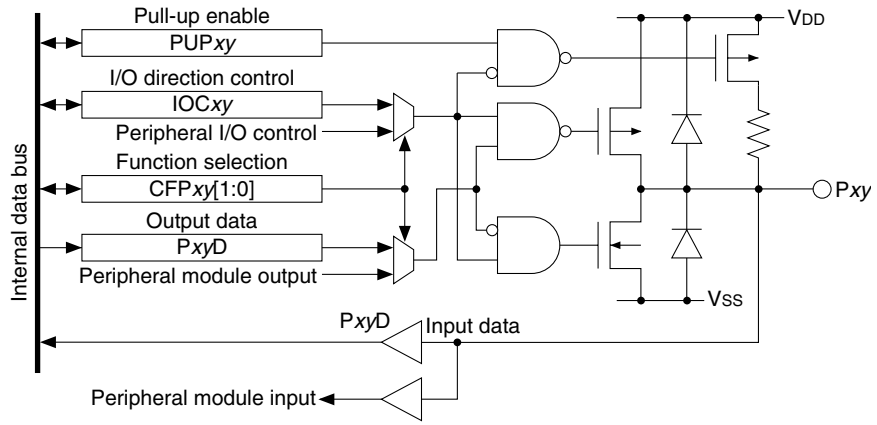


Figure 26.1.1 I/O Port Configuration

Notes: • The PCLK clock must be supplied from the CMU to access the I/O port. The prescaler (PSC Ch.2) output clock is also needed to operate the chattering filters. Turn on the prescaler when using this function.

• The "*xy*" in the register and bit names refers to the port number (P*xy*, *x* = 0 to F, *y* = 0 to 7).
Example: P*xy*D/GPIO_P*x*_DAT register
P00: P00D/GPIO_P0_DAT register
P17: P17D/GPIO_P1_DAT register

26.2 Input/Output Pin Function Selection (Port MUX)

The I/O port pins share peripheral module input/output pins. Each pin can be configured for use as an I/O port or for a peripheral module function via the corresponding port function select bits. Pins not used for peripheral modules can be used as general-purpose I/O ports.
The pin function can be selected using CFP*xy*[1:0]/PMUX_P*x*_∗ register.

Table 26.2.1 Input/Output Pin Function Selection

Package			Pin function 1 CFP*xy*[1:0] = 0x0 (default)	Pin function 2 CFP*xy*[1:0] = 0x1	Pin function 3 CFP*xy*[1:0] = 0x2	Pin function 4 CFP*xy*[1:0] = 0x3	Optional debug pins ∗
180pin /chip	144pin	128pin					
○	○	○	**P00**	USI_DI0	SIN	#HIF_WR	
○	○	○	**P01**	USI_DO0	SOUT	#HIF_RD	
○	○	○	**P02**	USI_CS0	SCLK	#HIF_CS	
○	○	○	**P03**	USI_CK0	REMC_I	HIF_D0	
○	○	○	**P04**	USI_DI1	T16A_ATMA_0	HIF_D1	
○	○	○	**P05**	USI_DO1	T16A_ATMB_0	HIF_D2	
○	○	○	**P06**	USI_CS1	T16A_ATMA_1	HIF_D3	
○	○	○	**P07**	USI_CK1	T16A_ATMB_1	HIF_D4	
○	○	○	**A24**	TFT_CTL0	P10	CMU_CLK	
○	○	○	**A25**	TFT_CTL1	P11	SOUT	
○	○	○	**P12**	I2S_SDO	PWM_H0	HIF_D5	
○	○	○	**P13**	I2S_WSO	PWM_L0	HIF_D6	
○	○	○	**P14**	I2S_SCLKO	PWM_H1	HIF_D7	
○	○	○	**P15**	I2S_MCLKO	PWM_L1	#HIF_WAIT	**DST0**
○	○	○	**P16**	#CE4	CMU_CLK	#NAND_WR	**DST1**
○	○	○	**P17**	#CE5	REMC_O	#NAND_RD	**DPCO**
○	○	○	**SDCKE**	P20			
○	○	○	**SDCLK**	P21			
○	○	○	**P30**	I2S_SDI	T16A_ATMA_2	HIF_A0	
○	○	○	**P31**	I2S_WSI	T16A_ATMB_2	HIF_A1	
○	○	○	**P32**	I2S_SCLKI	T16A_ATMA_3	HIF_A2	
○	○	○	**P33**	I2S_MCLKI	T16A_ATMB_3	HIF_A3	
○	○	○	**DCLK**	P34			
○	○	○	**DSIO**	P35			
○	○	○	**DST2**	P36			
○	○	○	**A21**	P40			
○	○	○	**A22**	P41	TFT_CTL2		
○	○	○	**A23**	P42	TFT_CTL3		
○	○	○	**#CE7**	P50	#SDCS		
○	○	○	**#CE8**	P51	#HIF_INT		
○	○	○	**#CE9**	P52			
○	○	○	**#CE10**	P53			
○	○	○	**#RD**	P54			
○	○	○	**#WRL**	P55			
○	○	○	**#WRH/#BSH**	P56			
○	○	○	**P60**	#WAIT	WDT_CLK	#WDT_NMI	
○	○	○	**P70**	AIN0	EXCL0		
○	N/A	N/A	**P71**	AIN1	I2S_WSI		
○	○	○	**P72**	AIN2	EXCL1		
○	N/A	N/A	**P73**	AIN3	I2S_SCLKI		
○	○	○	**P74**	AIN4	WDT_EXCL		
○	N/A	N/A	**P75**	AIN5	I2S_MCLKI		
○	○	○	**P76**	AIN6	#WAIT	#ADTRIG	
○	N/A	N/A	**P77**	AIN7			
○	○	○	**P80**	FPFRAME	USIL_DI	MMCD0	
○	○	○	**P81**	FPLINE	USIL_DO	MMCD1	
○	○	○	**P82**	FPSHIFT	USIL_CS	MMCD2	
○	○	○	**P83**	FPDRDY	USIL_CK	MMCD3	
○	○	○	**P90**	FPDAT0	USIL_LCD_D0	MMCCMD	
○	○	○	**P91**	FPDAT1	USIL_LCD_D1	MMCCLK	
○	○	○	**P92**	FPDAT2	USIL_LCD_D2	#NAND_WR	
○	○	○	**P93**	FPDAT3	USIL_LCD_D3	#NAND_RD	
○	○	○	**P94**	FPDAT4	USIL_LCD_D4	USI_DI2	
○	○	○	**P95**	FPDAT5	USIL_LCD_D5	USI_DO2	

Package			Pin function 1 CFP*xy*[1:0] = 0x0 (default)	Pin function 2 CFP*xy*[1:0] = 0x1	Pin function 3 CFP*xy*[1:0] = 0x2	Pin function 4 CFP*xy*[1:0] = 0x3	Optional debug pins *
180pin /chip	144pin	128pin					
○	○	○	**P96**	FPDAT6	USIL_LCD_D6	USI_CS2	
○	○	○	**P97**	FPDAT7	USIL_LCD_D7	USI_CK2	
○	○	N/A	**PA0**	HIF_D8	MMCD0	FPDAT16	
○	○	N/A	**PA1**	HIF_D9	MMCD1	FPDAT17	
○	○	N/A	**PA2**	HIF_D10	MMCD2	FPDAT18	
○	○	N/A	**PA3**	HIF_D11	MMCD3	FPDAT19	
○	○	N/A	**PA4**	HIF_D12	MMCCMD	FPDAT20	
○	○	N/A	**PA5**	HIF_D13	MMCCLK	FPDAT21	
○	○	N/A	**PA6**	HIF_D14	T16A_ATMA_0	FPDAT22	
○	○	N/A	**PA7**	HIF_D15	T16A_ATMB_0	FPDAT23	
○	○	N/A	**PB0**	FPDAT8	#NAND_WR	MMCD0	
○	○	N/A	**PB1**	FPDAT9	#NAND_RD	MMCD1	
○	○	N/A	**PB2**	FPDAT10	REMC_O	MMCD2	
○	○	N/A	**PB3**	FPDAT11	REMC_I	MMCD3	
○	○	N/A	**PB4**	FPDAT12	USI_DI2	MMCCMD	
○	○	N/A	**PB5**	FPDAT13	USI_DO2	MMCCLK	
○	○	N/A	**PB6**	FPDAT14	USI_CS2	T16A_ATMA_1	
○	○	N/A	**PB7**	FPDAT15	USI_CK2	T16A_ATMB_1	
○	○	○	**D8**	PC0			
○	○	○	**D9**	PC1			
○	○	○	**D10**	PC2			
○	○	○	**D11**	PC3			
○	○	○	**D12**	PC4			
○	○	○	**D13**	PC5			
○	○	○	**D14**	PC6			
○	○	○	**D15**	PC7			
○	N/A	N/A	**PD0**	FPDAT16	USI_DI0	SIN	
○	N/A	N/A	**PD1**	FPDAT17	USI_DO0	SOUT	
○	N/A	N/A	**PD2**	FPDAT18	USI_CS0	SCLK	
○	N/A	N/A	**PD3**	FPDAT19	USI_CK0		
○	N/A	N/A	**PD4**	FPDAT20	USI_DI1	#NAND_WR	
○	N/A	N/A	**PD5**	FPDAT21	USI_DO1	#NAND_RD	
○	N/A	N/A	**PD6**	FPDAT22	USI_CS1	T16A_ATMA_0	
○	N/A	N/A	**PD7**	FPDAT23	USI_CK1	T16A_ATMB_0	
○	N/A	N/A	**PE0**	MMCD0	T16A_ATMA_2	USI_DI2	
○	N/A	N/A	**PE1**	MMCD1	T16A_ATMB_2	USI_DO2	
○	N/A	N/A	**PE2**	MMCD2	T16A_ATMA_3	USI_CS2	
○	N/A	N/A	**PE3**	MMCD3	T16A_ATMB_3	USI_CK2	
○	N/A	N/A	**PE4**	MMCCMD	T16A_ATMA_1	REMC_O	
○	N/A	N/A	**PE5**	MMCCLK	T16A_ATMB_1	REMC_I	
○	N/A	N/A	**PF0**	I2S_SDO	PWM_H0	TFT_CTL0	
○	N/A	N/A	**PF1**	I2S_WSO	PWM_L0	TFT_CTL1	
○	N/A	N/A	**PF2**	I2S_SCLKO	PWM_H1	TFT_CTL2	
○	N/A	N/A	**PF3**	I2S_MCLKO	PWM_L1	TFT_CTL3	
○	N/A	N/A	**PF4**	I2S_SDI	#ADTRIG		

* When the PC trace debugging function is enabled (TRACE/MISC_TRACE = 0)

At initial reset, each I/O port pin (P*xy*) is initialized for the default function ("Pin function 1" in Table 26.2.1).

For information on functions other than the I/O ports, see "Pin Descriptions" in the "Overview" chapter or the descriptions of peripheral modules.
The sections below describe port functions with the pins set as general-purpose I/O ports.

Note: The port function select registers (PMUX_P*x*_*) are write-protected. Before these registers can be rewritten, the write protection must be removed by writing data 0x96 to PPROT[7:0]/GPIO_PROTECT register. Note that since unnecessary rewrites to the port function select registers could lead to erratic system operation, PPROT[7:0] should be set to other than 0x96 unless the port function select registers must be rewritten.

26.3 Data Input/Output

Data input/output control

The I/O ports allow selection of the data input/output direction for each bit using IOC*xy*/GPIO_P*x*_IOC register. Set IOC*xy* to 0 (default) to configure the P*xy* port for input; set IOC*xy* to 1 to configure the P*xy* port for output.

The input/output direction of ports with a peripheral module function selected is controlled by the peripheral module. IOC*xy* settings are ignored.

Data input

To input an external signal and read out the value, set IOC*xy* to 0 (input mode, default). The I/O port is placed into high-impedance status and it functions as an input port.

In input mode, the external signal level can be read out directly from P*xy*D/GPIO_P*x*_DAT register. The value read will be 1 when the input pin is at high level and 0 when it is at low level.

The port pin status can also be read in output mode (IOC*xy* = 1). In this case, the value actually output the port can be read out from P*xy*D.

Data output

To output data from the port pin, set IOC*xy* to 1 (output mode). The I/O port then functions as an output port, and the value set in the P*xy*D is output from the port pin. The port pin outputs high level when P*xy*D is set to 1 and low level when set to 0. Writing to P*xy*D is possible without affecting pin status, even in input mode.

26.4 Pull-up Control

The I/O port contains a pull-up resistor that can be enabled or disabled individually for each bit using PUP*xy*/GPIO_P*x*_PUP register. Setting PUP*xy* to 1 enables the pull-up resistor and pulls up the port pin in input mode. It will not be pulled up if set to 0. The PUP*xy* setting is ignored and not pulled up in output mode.

Unused I/O ports should be set with pull-up enabled.

This pull-up setting is also enabled for ports for which the peripheral module function has been selected.

26.5 Input Interrupts and DMA

The GPIO module has four interrupt systems (port input interrupts 0 and 1, key input interrupts 0 and 1).

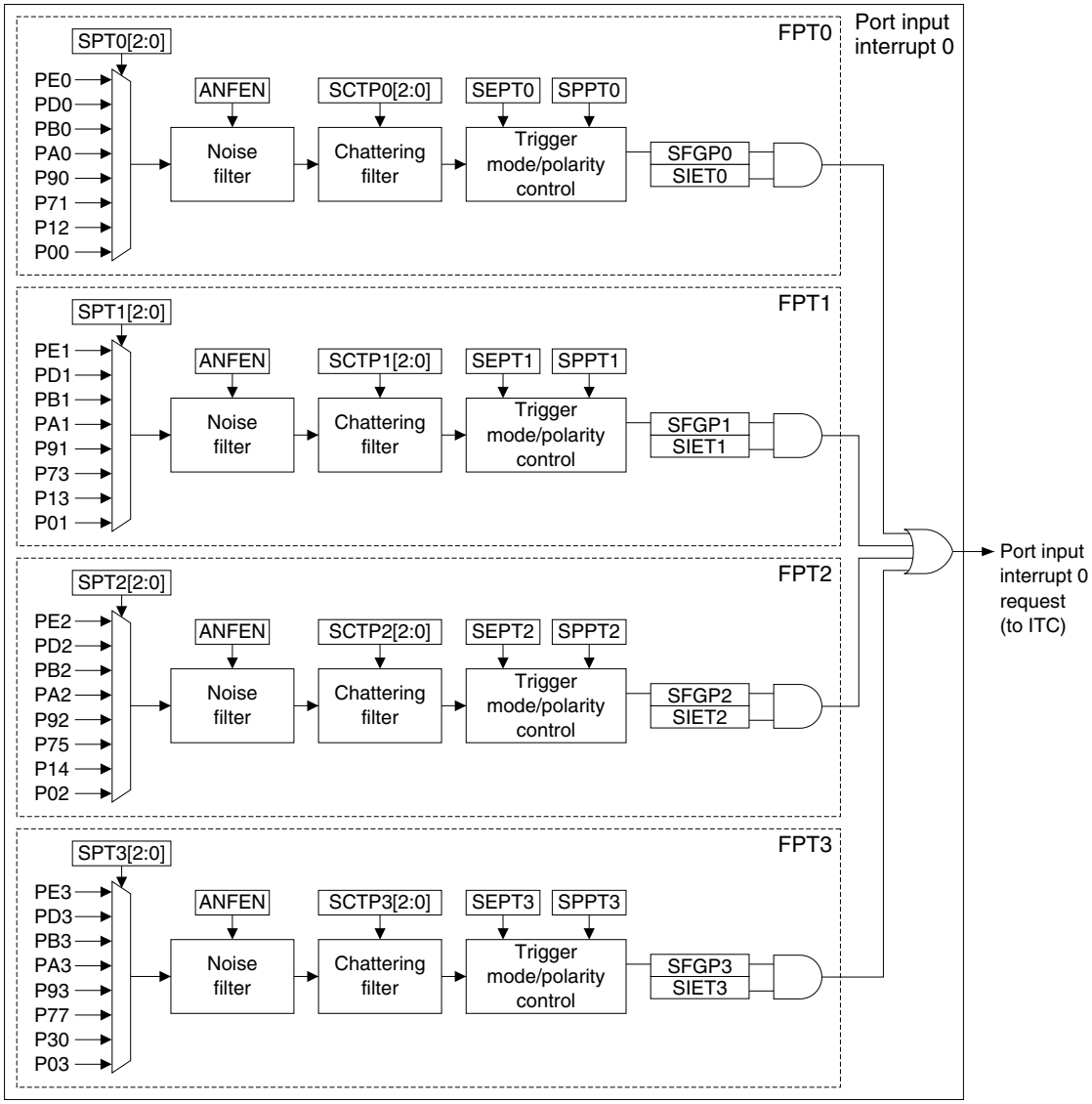
26.5.1 Port Input Interrupt

The port input interrupt system consists of four ports × two groups.

Port input interrupt 0 (group 0): FPT0–FPT3
Port input interrupt 1 (group 1): FPT4–FPT7

Each group can issue interrupt requests to the ITC.

Figure 26.5.1.1 shows the configuration of the port input interrupt circuit.


FPT0
Port input interrupt 0
SPT0[2:0]
PE0
PD0
PB0
PA0
P90
P71
P12
P00
ANFEN
SCTP0[2:0]
SEPT0
SPPT0
Noise filter
Chattering filter
Trigger mode/polarity control
SFGP0
SIET0
FPT1
SPT1[2:0]
PE1
PD1
PB1
PA1
P91
P73
P13
P01
ANFEN
SCTP1[2:0]
SEPT1
SPPT1
Noise filter
Chattering filter
Trigger mode/polarity control
SFGP1
SIET1
FPT2
SPT2[2:0]
PE2
PD2
PB2
PA2
P92
P75
P14
P02
ANFEN
SCTP2[2:0]
SEPT2
SPPT2
Noise filter
Chattering filter
Trigger mode/polarity control
SFGP2
SIET2
FPT3
SPT3[2:0]
PE3
PD3
PB3
PA3
P93
P77
P30
P03
ANFEN
SCTP3[2:0]
SEPT3
SPPT3
Noise filter
Chattering filter
Trigger mode/polarity control
SFGP3
SIET3
Port input interrupt 0 request (to ITC)

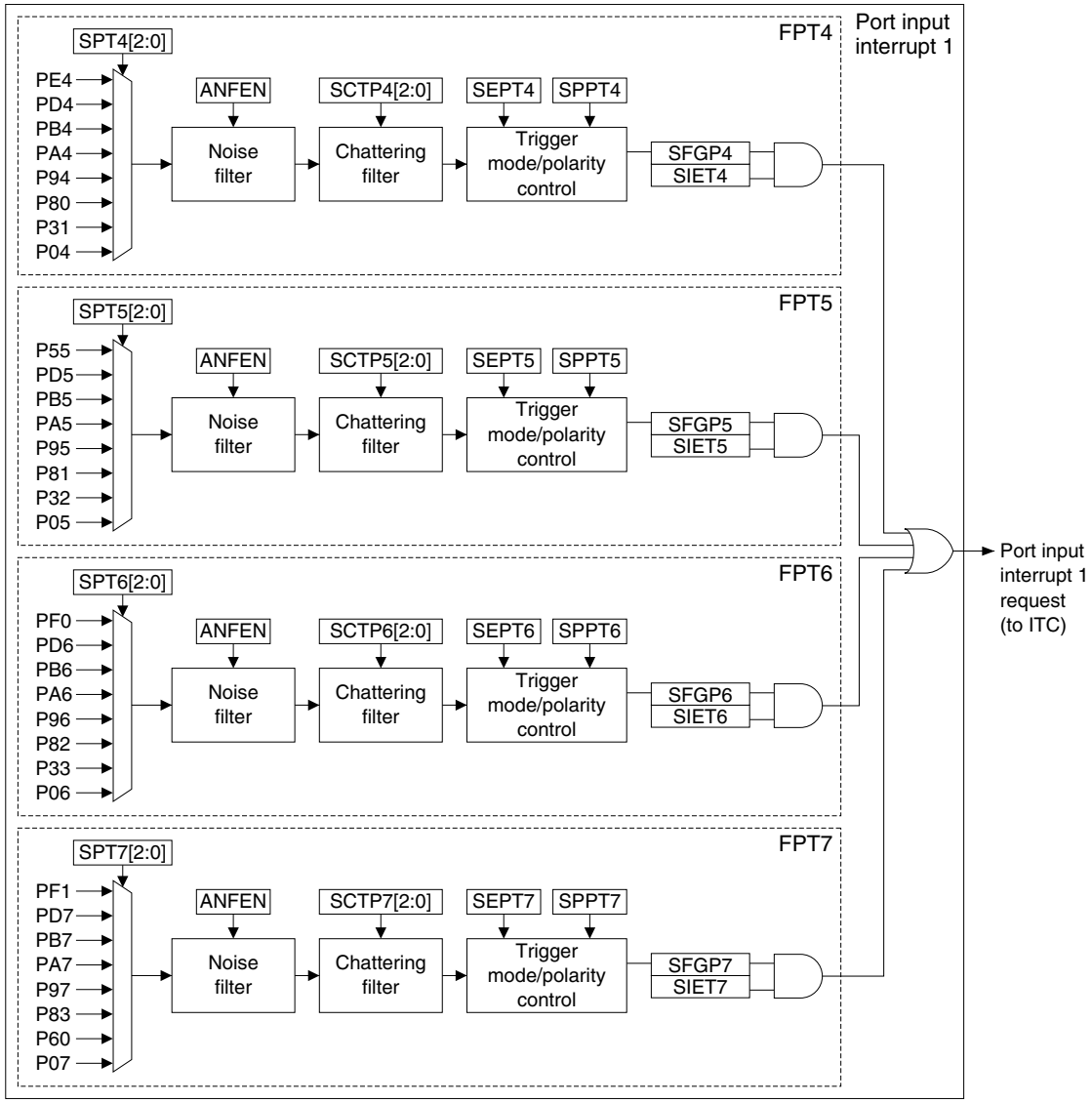


Figure 26.5.1.1 Port Input Interrupt Circuit Configuration

Interrupt port selection

Each port input interrupt system uses eight ports to generate interrupts. Select the ports using SPT*n*[2:0]/GPIO_FPT*nn*_SEL register (*n* = 0 to 7, *nn* = 01, 23, 45, or 67).

Table 26.5.1.1 Selecting Ports Used For Port Input Interrupt 0

SPT*n*[2:0] setting	FPT0 (SPT0[2:0])	FPT1 (SPT1[2:0])	FPT2 (SPT2[2:0])	FPT3 (SPT3[2:0])
0x7	PE0	PE1	PE2	PE3
0x6	PD0	PD1	PD2	PD3
0x5	PB0	PB1	PB2	PB3
0x4	PA0	PA1	PA2	PA3
0x3	P90	P91	P92	P93
0x2	P71	P73	P75	P77
0x1	P12	P13	P14	P30
0x0 (default)	P00	P01	P02	P03

Table 26.5.1.2 Selecting Ports Used For Port Input Interrupt 1

SPT*n*[2:0] setting	FPT4 (SPT4[2:0])	FPT5 (SPT5[2:0])	FPT6 (SPT6[2:0])	FPT7 (SPT7[2:0])
0x7	PE4	P55	PF0	PF1
0x6	PD4	PD5	PD6	PD7
0x5	PB4	PB5	PB6	PB7
0x4	PA4	PA5	PA6	PA7
0x3	P94	P95	P96	P97
0x2	P80	P81	P82	P83
0x1	P31	P32	P33	P60
0x0 (default)	P04	P05	P06	P07

Interrupt mode and polarity selection

The port input interrupt circuit provides two interrupt modes to set the interrupt flags: edge trigger mode and level trigger mode. The interrupt mode for each FPT line can be selected using SEPT*n*/GPIO_FPT*nn*_MOD register.

When SEPT*n* bit is set to 1 (default), the corresponding port is set to edge trigger mode. In edge trigger mode, the interrupt flag is set at the active edge of the input signal and it retains 1 until reset via software.

When SEPT*n* is set to 0, the corresponding port is set to level trigger mode. In level trigger mode, the interrupt flag is set when the input signal goes the active level and it retains 1 until reset via software.

SLEEP mode can be canceled by causing a port input interrupt regardless of how the GPIO interrupt mode (edge trigger/level trigger) is set.

The active level/edge of the input signal can be selected using SPPT*n*/GPIO_FPT*nn*_POL register.

When SPPT*n* is set to 1 (default), high level (in level trigger mode) or rising edge (in edge trigger mode) is selected.

When SPPT*n* is set to 0, low level (in level trigger mode) or falling edge (in edge trigger mode) is selected.

Table 26.5.1.3 Port input Interrupt Conditions

SEPT*n*	SPPT*n*	Port input interrupt condition
1	1	Rising edge input
1	0	Falling edge input
0	1	High level input
0	0	Low level input

Interrupt flags

The port input interrupt circuit provides eight interrupt flags (SFGP*n*/GPIO_FPT*nn*_FLG register) corresponding to the FPT interrupt ports.

In level trigger mode, the interrupt flag is set according to the input signal level.

In edge trigger mode, the interrupt flag is set at the active edge of the input signal.

The interrupt flag must be reset by writing 1 after an interrupt occurs.

Interrupt enable bits

Each FPT interrupt port can be enabled or disabled to generate interrupts using the corresponding interrupt enable bit (SIET*n*/GPIO_FPT*nn*_MSK register).

To use port input interrupts, the interrupt port pins must be configured as an I/O port using the corresponding port function select bits. Before setting SIET*n* to 1, the corresponding SFGP*n* must be cleared to 0.

To enable interrupts, set SIET*n* to 1. To disable interrupts, set SIET*n* to 0.

When SFGP*n* is set to 1 while the corresponding SIET*n* is set to 1, an interrupt request signal is output to the ITC. An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied.

For specific information on interrupt processing, see the “Interrupt Controller (ITC)” chapter.

Chattering Filters

Each FPT line includes a chattering filter circuit that can be disabled or enabled with a sampling clock specified using SCTP*n*[2:0]/GPIO_FPT*nn*_CHAT register.

When the chattering filter is enabled, glitches shorter than the filter sampling time × 2 will be filtered as noise. Pulses longer than the filter sampling time × 4 will not be filtered and can generate port input interrupts. If filter sampling time × 2 ≤ pulse width ≤ filter sampling time × 4, the pulse may be filtered depending on the input timing.

Table 26.5.1.4 Chattering Filter Settings

SCTP*n*[2:0]	Filter sampling time	Invalid pulse (glitch) width that will be filtered	Valid pulse width that will be accepted
0x7	64/f_{PCLK}	< 64/f_{PCLK} × 2	> 64/f_{PCLK} × 4
0x6	32/f_{PCLK}	< 32/f_{PCLK} × 2	> 32/f_{PCLK} × 4
0x5	16/f_{PCLK}	< 16/f_{PCLK} × 2	> 16/f_{PCLK} × 4
0x4	8/f_{PCLK}	< 8/f_{PCLK} × 2	> 8/f_{PCLK} × 4
0x3	4/f_{PCLK}	< 4/f_{PCLK} × 2	> 4/f_{PCLK} × 4
0x2	2/f_{PCLK}	< 2/f_{PCLK} × 2	> 2/f_{PCLK} × 4
0x1	1/f_{PCLK}	< 1/f_{PCLK} × 2	> 1/f_{PCLK} × 4
0x0	Not filtered		

(Default: 0x0)

Notes:
- The prescaler (PSC Ch.2) output is used as the filter clock. Make sure the prescaler (PSC Ch.2) is turned on before using the chattering filter. Do not enable the chattering filter when the prescaler (PSC Ch.2) is turned off, as undesired port input interrupts may be generated.
- The chattering filter stops operating in SLEEP mode, as no clock is supplied. In order to cancel SLEEP mode using a port input interrupt, the chattering filter will be automatically bypassed (Not filtered) in SLEEP mode until the CPU exits SLEEP mode even if the chattering filter is set to on.
- Setting the GPIO_FPT*nn*_CHAT register while the interrupt is enabled may generate an undesired port input interrupt. Therefore, the port input interrupt must be disabled before setting the GPIO_FPT*nn*_CHAT register. Furthermore, be sure to clear the port input interrupt flag before enabling the interrupt again after setting the GPIO_FPT*nn*_CHAT register. In this case, clear the interrupt flag after the wait time shown below has elapsed from the GPIO_FPT*nn*_CHAT register setting.

  Wait time [µs] = Filter sampling time [µs] × 4

  Example: When the filter sampling time is 64/f_{PCLK} and f_{PCLK} = 32 MHz
  Wait time = 64 × 4 / 32 = 8 [µs]

  The port input interrupt flag should be cleared after waiting 8 µs or more when the GPIO_FPT*nn*_CHAT register is set to 0x7.

26.5.2 Key Input Interrupt

The key input interrupt system consists of 16 ports × two groups.

Key input interrupt 0 (group 0): FPK000–FPK015
Key input interrupt 1 (group 1): FPK100–FPK115

Each group can issue interrupt requests to the ITC.
Figure 26.5.2.1 shows the configuration of the key input interrupt circuit.

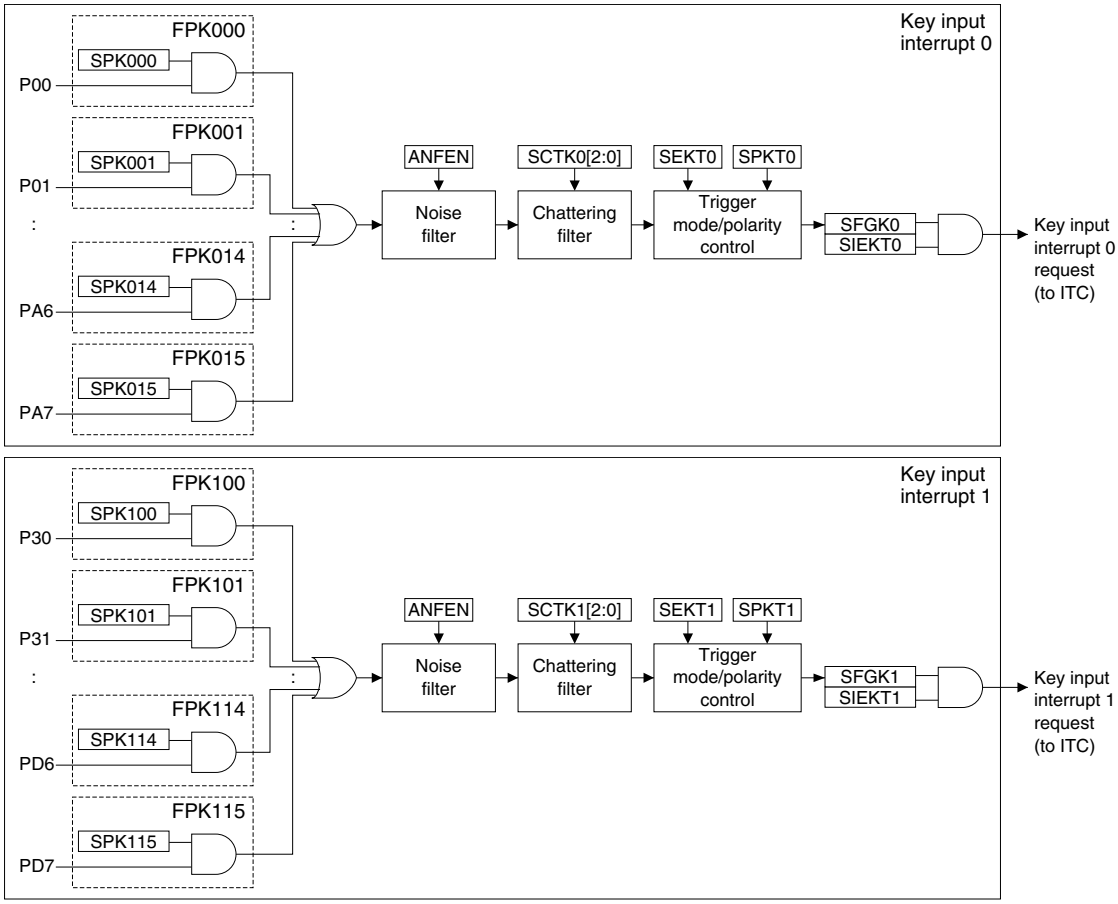


Figure 26.5.2.1 Key Input Interrupt Circuit Configuration

Interrupt port selection

Each key input interrupt system has 16 ports to generate interrupts. Enable the ports to generate key input interrupts by setting the GPIO_FPK*n*_07_SEL and GPIO_FPK*n*_8F_SEL registers (*n* = 0 or 1).

Table 26.5.2.1 Key Input Interrupt Port Assignment

Key input interrupt group	Key input interrupt source	I/O port	Interrupt select bit
FPK0	FPK000	P00	SPK000/GPIO_FPK0_07 register
	FPK001	P01	SPK001/GPIO_FPK0_07 register
	FPK002	P02	SPK002/GPIO_FPK0_07 register
	FPK003	P03	SPK003/GPIO_FPK0_07 register
	FPK004	P04	SPK004/GPIO_FPK0_07 register
	FPK005	P05	SPK005/GPIO_FPK0_07 register
	FPK006	P06	SPK006/GPIO_FPK0_07 register
	FPK007	P07	SPK007/GPIO_FPK0_07 register
	FPK008	PA0	SPK008/GPIO_FPK0_8F register
	FPK009	PA1	SPK009/GPIO_FPK0_8F register
	FPK010	PA2	SPK010/GPIO_FPK0_8F register
	FPK011	PA3	SPK011/GPIO_FPK0_8F register
	FPK012	PA4	SPK012/GPIO_FPK0_8F register
	FPK013	PA5	SPK013/GPIO_FPK0_8F register
	FPK014	PA6	SPK014/GPIO_FPK0_8F register
	FPK015	PA7	SPK015/GPIO_FPK0_8F register

Key input interrupt group	Key input interrupt source	I/O port	Interrupt select bit
FPK1	FPK100	P30	SPK100/GPIO_FPK1_07 register
	FPK101	P31	SPK101/GPIO_FPK1_07 register
	FPK102	P32	SPK102/GPIO_FPK1_07 register
	FPK103	P33	SPK103/GPIO_FPK1_07 register
	FPK104	P70	SPK104/GPIO_FPK1_07 register
	FPK105	P72	SPK105/GPIO_FPK1_07 register
	FPK106	P74	SPK106/GPIO_FPK1_07 register
	FPK107	P76	SPK107/GPIO_FPK1_07 register
	FPK108	PD0	SPK108/GPIO_FPK1_8F register
	FPK109	PD1	SPK109/GPIO_FPK1_8F register
	FPK110	PD2	SPK110/GPIO_FPK1_8F register
	FPK111	PD3	SPK111/GPIO_FPK1_8F register
	FPK112	PD4	SPK112/GPIO_FPK1_8F register
	FPK113	PD5	SPK113/GPIO_FPK1_8F register
	FPK114	PD6	SPK114/GPIO_FPK1_8F register
	FPK115	PD7	SPK115/GPIO_FPK1_8F register

Interrupt mode and polarity selection

The key input interrupt circuit provides two interrupt modes to set the interrupt flags: edge trigger mode and level trigger mode. The interrupt mode for the FPK*n* group can be selected using SEKT*n*/GPIO_FPK01_MOD register.

When SEKT*n* bit is set to 1 (default), the FPK*n* ports are set to edge trigger mode. In edge trigger mode, the interrupt flag is set at the active edge of the input signal and it retains 1 until reset via software.

When SEKT*n* is set to 0, the FPK*n* ports are set to level trigger mode. In level trigger mode, the interrupt flag is set when the input signal goes the active level and it retains 1 until reset via software.

SLEEP mode can be canceled by causing a key input interrupt regardless of how the GPIO interrupt mode (edge trigger/level trigger) is set.

The active level/edge of the input signal can be selected using SPKT*n*/GPIO_FPK01_POL register. When SPKT*n* is set to 1 (default), high level (in level trigger mode) or rising edge (in edge trigger mode) is selected. When SPKT*n* is set to 0, low level (in level trigger mode) or falling edge (in edge trigger mode) is selected.

Table 26.5.2.2 Key input Interrupt Conditions

SEKT*n*	SPKT*n*	Key input interrupt condition
1	1	Rising edge input
1	0	Falling edge input
0	1	High level input
0	0	Low level input

Interrupt flags

The key input interrupt circuit provides two interrupt flags (SFGK*n*/GPIO_FPK01_FLG register) corresponding to the FPK interrupt groups.

In level trigger mode, the interrupt flag is set according to the input signal level.

In edge trigger mode, the interrupt flag is set at the active edge of the input signal.

The interrupt flag must be reset by writing 1 after an interrupt occurs.

Interrupt enable bits

Key input interrupts can be enabled or disabled using the corresponding interrupt enable bit (SIEKT*n*/GPIO_FP01_MSK register).

To use key input interrupts, the interrupt port pins must be configured as an I/O port using the corresponding port function select bits. Before setting SIEKT*n* to 1, the corresponding SFGK*n* must be cleared to 0.

To enable interrupts, set SIKET*n* to 1. To disable interrupts, set SIEKT*n* to 0.

When SFGK*n* is set to 1 while the corresponding SIEKT*n* is set to 1, an interrupt request signal is output to the ITC. An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied.

For specific information on interrupt processing, see the “Interrupt Controller (ITC)” chapter.

Chattering Filters

The key input interrupt circuit includes a chattering filter for key entry that can be disabled or enabled with a sampling clock specified using SCTK*n*[2:0]/GPIO_FPK01_CHAT register.
When the chattering filter is enabled, glitches shorter than the filter sampling time × 2 will be filtered as noise. Pulses longer than the filter sampling time × 4 will not be filtered and can generate key input interrupts. If filter sampling time × 2 ≤ pulse width ≤ filter sampling time × 4, the pulse may be filtered depending on the input timing.

Table 26.5.2.3 Chattering Filter Settings

SCTK*n*[2:0]	Filter sampling time	Invalid pulse (glitch) width that will be filtered	Valid pulse width that will be accepted
0x7	64/fPCLK	< 64/fPCLK × 2	> 64/fPCLK × 4
0x6	32/fPCLK	< 32/fPCLK × 2	> 32/fPCLK × 4
0x5	16/fPCLK	< 16/fPCLK × 2	> 16/fPCLK × 4
0x4	8/fPCLK	< 8/fPCLK × 2	> 8/fPCLK × 4
0x3	4/fPCLK	< 4/fPCLK × 2	> 4/fPCLK × 4
0x2	2/fPCLK	< 2/fPCLK × 2	> 2/fPCLK × 4
0x1	1/fPCLK	< 1/fPCLK × 2	> 1/fPCLK × 4
0x0	Not filtered		

(Default: 0x0)

Notes:
- The prescaler (PSC Ch.2) output is used as the filter clock. Make sure the prescaler (PSC Ch.2) is turned on before using the chattering filter. Do not enable the chattering filter when the prescaler (PSC Ch.2) is turned off, as undesired key input interrupts may be generated.
- The chattering filter stops operating in SLEEP mode, as no clock is supplied. In order to cancel SLEEP mode using a key input interrupt, the chattering filter will be automatically bypassed (Not filtered) in SLEEP mode until the CPU exits SLEEP mode even if the chattering filter is set to on.
- Setting the GPIO_FPK01_CHAT register while the interrupt is enabled may generate an undesired key input interrupt. Therefore, the key input interrupt must be disabled before setting the GPIO_FPK01_CHAT register. Furthermore, be sure to clear the key input interrupt flag before enabling the interrupt again after setting the GPIO_FPK01_CHAT register. In this case, clear the interrupt flag after the wait time shown below has elapsed from the GPIO_FPK01_CHAT register setting.

  Wait time [µs] = Filter sampling time [µs] × 4

  Example: When the filter sampling time is 64/fPCLK and fPCLK = 32 MHz
  Wait time = 64 × 4 / 32 = 8 [µs]

  The key input interrupt flag should be cleared after waiting 8 µs or more when the GPIO_FPK*nn*_CHAT register is set to 0x7.

26.5.3 DMA Trigger

A cause of port input interrupt can invoke DMA transfer. An FPT interrupt port can be selected as a DMA trigger source using the SDMAT[2:0]/GPIO_DMA register.

Table 26.5.3.1 DMA Trigger Source Selection

SDMAT[2:0]	Trigger source
0x7	FPT7
0x6	FPT6
0x5	FPT5
0x4	FPT4
0x3	FPT3
0x2	FPT2
0x1	FPT1
0x0	FPT0

(Default: 0x0)

The interrupt signal of the selected FPT line, which is generated according to the interrupt mode and polarity settings regardless of the SIET*n* setting (even if the interrupt is disabled), is sent to the DMAC to trigger a DMA transfer. For more information on DMA transfer, see the "DMA Controller (DMAC)" chapter.

Note: After setting SCTP*n*[2:0]/GPIO_FPT*nn*_CHAT register, the port may send an undesired DMA trigger to the DMAC. Disable the port DMA request for at least the wait time shown below after the GPIO_FPT*nn*_CHAT register is set.

Wait time [µs] = Filter sampling time [µs] × 4

Example: When the filter sampling time is 64/f_{PCLK} and f_{PCLK} = 32 MHz
Wait time = 64 × 4 / 32 = 8 [µs]

The port DMA function can be used after waiting 8 µs or more when the GPIO_FPT*nn*_CHAT register is set to 0x7.

26.6 Input Port Noise Filters

The S1C33L27 provides noise filters to remove noise on the signals input from the ports shown below.

USI: USI_DI0, USI_DI1, USI_DI2, USI_CS0, USI_CS1, USI_CS2, USI_CK0, USI_CK1, USI_CK2
USIL: USIL_DI, USIL_CS, USIL_CK
UART: SIN, SCLK
REMC: REMC_I
I2S: I2S_SDI, I2S_WSI, I2S_SCLKI, I2S_MCLKI
T16A6: EXCL0, EXCL1, T16A_ATMA_0, T16A_ATMA_1, T16A_ATMA_2, T16A_ATMA_3, T16A_ATMB_0, T16A_ATMB_1, T16A_ATMB_2, T16A_ATMB_3
ADC10: #ADTRIG
WDT: WDT_EXCL
SD_MMC: MMCD[3:0], MMCCMD
GPIO: Ports used for FPT0–FPT7 port input interrupts and FPK0–FPK1 key input interrupts (See note below.)

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

Notes:
- These noise filters cannot be enabled individually.
- The noise filters are not effective if these ports are used as general-purpose input port. However, the noise filters for the general-purpose input ports that are selected as FPT interrupt ports (FPT0 to FPT7) or FPK interrupt ports (FPK0 and FPK1) are effective.
- The GPIO_FILTER register is write-protected. Before the register can be rewritten, the write protection must be removed by writing data 0x96 to PPROT[7:0]/GPIO_PROTECT register.

26.7 Bus Drive Control

The external data bus (D[15:0]) pins and the external address bus (A[25:0]) pins can be forcibly set to low level using LDRVDB/GPIO_BUS_DRV register and LDRVAD/GPIO_BUS_DRV register, respectively. This function is useful when turning off the power of the external device connected to the bus.
When the control bit is set to 1, the corresponding bus signals go low. When the control bit is set to 0, the signal control goes back to the SRAMC/SDRAMC.

Notes:
- The low-drive control bit is disabled when the pin is used as a general-purpose I/O port (P*xy*).
- If the above bus signals are forcibly driven low when the CPU is running by the instructions fetched from an external memory, the CPU will not be able to run after that point. To drive the signals low, the CPU must be running with the program stored in the internal RAM.

26.8 Control Register Details

Table 26.8.1 List of GPIO and Port MUX Registers

Address	Register name		Function
0x300300	GPIO_P0_DAT	P0 Port Data Register	P0 port input/output data
0x300301	GPIO_P0_IOC	P0 Port I/O Control Register	Control P0 port input/output direction
0x300302	GPIO_P1_DAT	P1 Port Data Register	P1 port input/output data
0x300303	GPIO_P1_IOC	P1 Port I/O Control Register	Control P1 port input/output direction
0x300304	GPIO_P2_DAT	P2 Port Data Register	P2 port input/output data
0x300305	GPIO_P2_IOC	P2 Port I/O Control Register	Control P2 port input/output direction
0x300306	GPIO_P3_DAT	P3 Port Data Register	P3 port input/output data
0x300307	GPIO_P3_IOC	P3 Port I/O Control Register	Control P3 port input/output direction
0x300308	GPIO_P4_DAT	P4 Port Data Register	P4 port input/output data
0x300309	GPIO_P4_IOC	P4 Port I/O Control Register	Control P4 port input/output direction
0x30030a	GPIO_P5_DAT	P5 Port Data Register	P5 port input/output data
0x30030b	GPIO_P5_IOC	P5 Port I/O Control Register	Control P5 port input/output direction
0x30030c	GPIO_P6_DAT	P6 Port Data Register	P6 port input/output data
0x30030d	GPIO_P6_IOC	P6 Port I/O Control Register	Control P6 port input/output direction
0x30030e	GPIO_P7_DAT	P7 Port Data Register	P7 port input data
0x300310	GPIO_P8_DAT	P8 Port Data Register	P8 port input/output data
0x300311	GPIO_P8_IOC	P8 Port I/O Control Register	Control P8 port input/output direction
0x300312	GPIO_P9_DAT	P9 Port Data Register	P9 port input/output data
0x300313	GPIO_P9_IOC	P9 Port I/O Control Register	Control P9 port input/output direction
0x300314	GPIO_PA_DAT	PA Port Data Register	PA port input/output data
0x300315	GPIO_PA_IOC	PA Port I/O Control Register	Control PA port input/output direction
0x300316	GPIO_PB_DAT	PB Port Data Register	PB port input/output data
0x300317	GPIO_PB_IOC	PB Port I/O Control Register	Control PB port input/output direction
0x300318	GPIO_PC_DAT	PC Port Data Register	PC port input/output data
0x300319	GPIO_PC_IOC	PC Port I/O Control Register	Control PC port input/output direction
0x30031a	GPIO_PD_DAT	PD Port Data Register	PD port input/output data
0x30031b	GPIO_PD_IOC	PD Port I/O Control Register	Control PD port input/output direction
0x30031c	GPIO_PE_DAT	PE Port Data Register	PE port input/output data
0x30031d	GPIO_PE_IOC	PE Port I/O Control Register	Control PE port input/output direction
0x30031e	GPIO_PF_DAT	PF Port Data Register	PF port input/output data
0x30031f	GPIO_PF_IOC	PF Port I/O Control Register	Control PF port input/output direction
0x300340	GPIO_FPT01_SEL	FPT0–1 Interrupt Port Select Register	Select ports used for FPT0–1 interrupts
0x300341	GPIO_FPT23_SEL	FPT2–3 Interrupt Port Select Register	Select ports used for FPT2–3 interrupts
0x300342	GPIO_FPT45_SEL	FPT4–5 Interrupt Port Select Register	Select ports used for FPT4–5 interrupts
0x300343	GPIO_FPT67_SEL	FPT6–7 Interrupt Port Select Register	Select ports used for FPT6–7 interrupts
0x300344	GPIO_FPT03_POL	FPT0–3 Interrupt Polarity Select Register	Select input signal polarity for FPT0–3 interrupts
0x300345	GPIO_FPT47_POL	FPT4–7 Interrupt Polarity Select Register	Select input signal polarity for FPT4–7 interrupts
0x300346	GPIO_FPT03_MOD	FPT0–3 Interrupt Mode Select Register	Select edge/level mode for FPT0–3 interrupts
0x300347	GPIO_FPT47_MOD	FPT4–7 Interrupt Mode Select Register	Select edge/level mode for FPT4–7 interrupts
0x300348	GPIO_FPT03_MSK	FPT0–3 Interrupt Mask Register	Enable/disable FPT0–3 interrupts
0x300349	GPIO_FPT47_MSK	FPT4–7 Interrupt Mask Register	Enable/disable FPT4–7 interrupts
0x30034a	GPIO_FPT03_FLG	FPT0–3 Interrupt Flag Register	Indicate FPT0–3 interrupt cause status
0x30034b	GPIO_FPT47_FLG	FPT4–7 Interrupt Flag Register	Indicate FPT4–7 interrupt cause status
0x30034c	GPIO_FPT01_CHAT	FPT0–1 Interrupt Chattering Filter Control Register	Control FPT0–1 chattering filter
0x30034d	GPIO_FPT23_CHAT	FPT2–3 Interrupt Chattering Filter Control Register	Control FPT2–3 chattering filter
0x30034e	GPIO_FPT45_CHAT	FPT4–5 Interrupt Chattering Filter Control Register	Control FPT4–5 chattering filter
0x30034f	GPIO_FPT67_CHAT	FPT6–7 Interrupt Chattering Filter Control Register	Control FPT6–7 chattering filter
0x300350	GPIO_DMA	Port DMA Trigger Source Select Register	Select port DMA trigger source
0x300360	GPIO_FPK0_07_SEL	FPK000–007 Interrupt Select Register	Select ports used for FPK0 interrupts
0x300361	GPIO_FPK0_8F_SEL	FPK008–015 Interrupt Select Register	
0x300362	GPIO_FPK1_07_SEL	FPK100–107 Interrupt Select Register	Select ports used for FPK1 interrupts
0x300363	GPIO_FPK1_8F_SEL	FPK108–115 Interrupt Select Register	
0x300370	GPIO_FPK01_POL	FPK0–1 Interrupt Polarity Select Register	Select input signal polarity for FPK0–1 interrupts
0x300371	GPIO_FPK01_MOD	FPK0–1 Interrupt Mode Select Register	Select edge/level mode for FPK0–1 interrupts
0x300372	GPIO_FPK01_MSK	FPK0–1 Interrupt Mask Register	Enable/disable FPK0–1 interrupts
0x300373	GPIO_FPK01_FLG	FPK0–1 Interrupt Flag Register	Indicate FPK0–1 interrupt cause status
0x300374	GPIO_FPK01_CHAT	FPK0–1 Interrupt Chattering Filter Control Register	Control FPK0–1 chattering filter
0x300800	PMUX_P0_03	P0[3:0] Port Function Select Register	Select P0[3:0] port functions
0x300801	PMUX_P0_47	P0[7:4] Port Function Select Register	Select P0[7:4] port functions
0x300802	PMUX_P1_03	P1[3:0] Port Function Select Register	Select P1[3:0] port functions
0x300803	PMUX_P1_47	P1[7:4] Port Function Select Register	Select P1[7:4] port functions
0x300804	PMUX_P2_01	P2[1:0] Port Function Select Register	Select P2[1:0] port functions
0x300806	PMUX_P3_03	P3[3:0] Port Function Select Register	Select P3[3:0] port functions
0x300807	PMUX_P3_46	P3[6:4] Port Function Select Register	Select P3[6:4] port functions

Address	Register name		Function
0x300808	PMUX_P4_02	P4[2:0] Port Function Select Register	Select P4[2:0] port functions
0x30080a	PMUX_P5_03	P5[3:0] Port Function Select Register	Select P5[3:0] port functions
0x30080b	PMUX_P5_46	P5[6:4] Port Function Select Register	Select P5[6:4] port functions
0x30080c	PMUX_P6_0	P60 Port Function Select Register	Select P60 port function
0x30080e	PMUX_P7_03	P7[3:0] Port Function Select Register	Select P7[3:0] port functions
0x30080f	PMUX_P7_47	P7[7:4] Port Function Select Register	Select P7[7:4] port functions
0x300810	PMUX_P8_03	P8[3:0] Port Function Select Register	Select P8[3:0] port functions
0x300812	PMUX_P9_03	P9[3:0] Port Function Select Register	Select P9[3:0] port functions
0x300813	PMUX_P9_47	P9[7:4] Port Function Select Register	Select P9[7:4] port functions
0x300814	PMUX_PA_03	PA[3:0] Port Function Select Register	Select PA[3:0] port functions
0x300815	PMUX_PA_47	PA[7:4] Port Function Select Register	Select PA[7:4] port functions
0x300816	PMUX_PB_03	PB[3:0] Port Function Select Register	Select PB[3:0] port functions
0x300817	PMUX_PB_47	PB[7:4] Port Function Select Register	Select PB[7:4] port functions
0x300818	PMUX_PC_03	PC[3:0] Port Function Select Register	Select PC[3:0] port functions
0x300819	PMUX_PC_47	PC[7:4] Port Function Select Register	Select PC[7:4] port functions
0x30081a	PMUX_PD_03	PD[3:0] Port Function Select Register	Select PD[3:0] port functions
0x30081b	PMUX_PD_47	PD[7:4] Port Function Select Register	Select PD[7:4] port functions
0x30081c	PMUX_PE_03	PE[3:0] Port Function Select Register	Select PE[3:0] port functions
0x30081d	PMUX_PE_45	PE[5:4] Port Function Select Register	Select PE[5:4] port functions
0x30081e	PMUX_PF_03	PF[3:0] Port Function Select Register	Select PF[3:0] port functions
0x30081f	PMUX_PF_4	PF4 Port Function Select Register	Select PF4 port function
0x300820	GPIO_P0_PUP	P0 Port Pull-up Control Register	Enable/disable P0 port pull-up resistors
0x300821	GPIO_P1_PUP	P1 Port Pull-up Control Register	Enable/disable P1 port pull-up resistors
0x300822	GPIO_P2_PUP	P2 Port Pull-up Control Register	Enable/disable P2 port pull-up resistors
0x300823	GPIO_P3_PUP	P3 Port Pull-up Control Register	Enable/disable P3 port pull-up resistors
0x300824	GPIO_P4_PUP	P4 Port Pull-up Control Register	Enable/disable P4 port pull-up resistors
0x300825	GPIO_P5_PUP	P5 Port Pull-up Control Register	Enable/disable P5 port pull-up resistors
0x300826	GPIO_P6_PUP	P6 Port Pull-up Control Register	Enable/disable P6 port pull-up resistors
0x300827	GPIO_P7_PUP	P7 Port Pull-up Control Register	Enable/disable P7 port pull-up resistors
0x300828	GPIO_P8_PUP	P8 Port Pull-up Control Register	Enable/disable P8 port pull-up resistors
0x300829	GPIO_P9_PUP	P9 Port Pull-up Control Register	Enable/disable P9 port pull-up resistors
0x30082a	GPIO_PA_PUP	PA Port Pull-up Control Register	Enable/disable PA port pull-up resistors
0x30082b	GPIO_PB_PUP	PB Port Pull-up Control Register	Enable/disable PB port pull-up resistors
0x30082d	GPIO_PD_PUP	PD Port Pull-up Control Register	Enable/disable PD port pull-up resistors
0x30082e	GPIO_PE_PUP	PE Port Pull-up Control Register	Enable/disable PE port pull-up resistors
0x30082f	GPIO_PF_PUP	PF Port Pull-up Control Register	Enable/disable PF port pull-up resistors
0x300830	GPIO_BUS_DRV	Bus Drive Control Register	Set external data and address bus signals to low
0x30083e	GPIO_FILTER	Port Noise Filter Control Register	Enable/disable port input noise filter
0x30083f	GPIO_PROTECT	GPIO/PMUX Write Protect Register	Enable/disable write protection for PMUX, GPIO_BUS_DRV, GPIO_FILTER, and GPIO_P*x*_PUP registers

The I/O port registers are described in detail below.

Notes: • When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

• The GPIO_BUS_DRV register, GPIO_P*x*_PUP registers, GPIO_FILTER register, and PMUX_P*x_yy* registers are write-protected. Before these registers can be rewritten, the write protection must be removed by writing data 0x96 to PPROT[7:0]GPIO_PROTECT register. Note that since unnecessary rewrites to the port function select registers could lead to erratic system operation, PPROT[7:0] should be set to other than 0x96 unless the registers above must be rewritten.

P*x* Port Data Registers (GPIO_P*x*_DAT)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
P*x* Port Data Register (GPIO_P*x*_DAT)	**0x300300 0x300302 | 0x30031e** (8 bits)	D7–0	**P*x*[7:0]D**	P*x*[7:0] I/O port data	1	1 (High)	0	0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.

Note: The P*xy*D bits for unavailable pins are read only bits from which 0 is always read out.

D[7:0] P*x*[7:0]D: P*x*[7:0] I/O Port Data Bits

These bits 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

P*xy*D corresponds directly to the P*xy* pin.
The pin voltage level can be read out (even if the port is set to output mode (IOC*xy*/GPIO_P*x*_IOC register = 1)). The value read out will be 1 when the pin voltage is high and 0 when low.
When the port is set to output mode (IOC*xy*/GPIO_P*x*_IOC register = 1), the data written will be output unchanged from the port pins. The port pin will be high when the data bit is set to 1 and low when set to 0. Port data can also be written in input mode (IOC*xy* = 0) (the pin status is unaffected).

P*x* Port I/O Control Registers (GPIO_P*x*_IOC)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
P*x* Port I/O Control Register (GPIO_P*x*_IOC)	**0x300301 0x300303 | 0x30031f** (8 bits)	D7–0	**IOC*x*[7:0]**	P*x*[7:0] I/O control	1	Output	0	Input	0x0	R/W	

Note: The IOC*xy* bits for unavailable pins are read only bits from which 0 is always read out.

D[7:0] IOC*x*[7:0]: P*x*[7:0] I/O Control Bits

Sets the port to input or output mode.
1 (R/W): Output mode
0 (R/W): Input mode (default)

IOC*xy* is the I/O direction control bit that corresponds directly to P*xy* port. Setting to 1 enables output and the data set in P*xy*D is output from the port pin. Output is disabled when IOC*xy* is set to 0, and the port pin is set into high-impedance status for inputting an external signal. The peripheral module determines whether output is enabled or disabled when the port is used for a peripheral module function.

FPT0–1 Interrupt Port Select Register (GPIO_FPT01_SEL)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT0–1 Interrupt Port Select Register (GPIO_FPT01_SEL)	**0x300340** (8 bits)	D7	**–**	reserved	–		–	–	0 when being read.
		D6–4	**SPT1[2:0]**	FPT1 interrupt input port select	SPT1[2:0]	Port	0x0	R/W	
					0x7	PE1			
					0x6	PD1			
					0x5	PB1			
					0x4	PA1			
					0x3	P91			
					0x2	P73			
					0x1	P13			
					0x0	P01			
		D3	**–**	reserved	–		–	–	0 when being read.
		D2–0	**SPT0[2:0]**	FPT0 interrupt input port select	SPT0[2:0]	Port	0x0	R/W	
					0x7	PE0			
					0x6	PD0			
					0x5	PB0			
					0x4	PA0			
					0x3	P90			
					0x2	P71			
					0x1	P12			
					0x0	P00			

D7 Reserved

D[6:4] SPT1[2:0]: FPT1 Interrupt Input Port Select Bits

Selects an FPT1 port used for generating port input interrupt 0.

D3 Reserved

D[2:0] SPT0[2:0]: FPT0 Interrupt Input Port Select Bits

Selects an FPT0 port used for generating port input interrupt 0.

Table 26.8.2 Selecting Ports Used For Port Input Interrupt 0

SPT*n*[2:0] setting	FPT0 (SPT0[2:0])	FPT1 (SPT1[2:0])	FPT2 (SPT2[2:0])	FPT3 (SPT3[2:0])
0x7	PE0	PE1	PE2	PE3
0x6	PD0	PD1	PD2	PD3
0x5	PB0	PB1	PB2	PB3
0x4	PA0	PA1	PA2	PA3
0x3	P90	P91	P92	P93
0x2	P71	P73	P75	P77
0x1	P12	P13	P14	P30
0x0 (default)	P00	P01	P02	P03

FPT2–3 Interrupt Port Select Register (GPIO_FPT23_SEL)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT2–3 Interrupt Port Select Register (GPIO_FPT23_SEL)	0x300341 (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SPT3[2:0]	FPT3 interrupt input port select	SPT3[2:0]	Port	0x0	R/W	
					0x7	PE3			
					0x6	PD3			
					0x5	PB3			
					0x4	PA3			
					0x3	P93			
					0x2	P77			
					0x1	P30			
					0x0	P03			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SPT2[2:0]	FPT2 interrupt input port select	SPT2[2:0]	Port	0x0	R/W	
					0x7	PE2			
					0x6	PD2			
					0x5	PB2			
					0x4	PA2			
					0x3	P92			
					0x2	P75			
					0x1	P14			
					0x0	P02			

D7 Reserved

D[6:4] SPT3[2:0]: FPT3 Interrupt Input Port Select Bits

Selects an FPT3 port used for generating port input interrupt 0.

D3 Reserved

D[2:0] SPT2[2:0]: FPT2 Interrupt Input Port Select Bits

Selects an FPT2 port used for generating port input interrupt 0.

FPT4–5 Interrupt Port Select Register (GPIO_FPT45_SEL)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT4–5 Interrupt Port Select Register (GPIO_FPT45_SEL)	0x300342 (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SPT5[2:0]	FPT5 interrupt input port select	SPT5[2:0]	Port	0x0	R/W	
					0x7	P55			
					0x6	PD5			
					0x5	PB5			
					0x4	PA5			
					0x3	P95			
					0x2	P81			
					0x1	P32			
					0x0	P05			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SPT4[2:0]	FPT4 interrupt input port select	SPT4[2:0]	Port	0x0	R/W	
					0x7	PE4			
					0x6	PD4			
					0x5	PB4			
					0x4	PA4			
					0x3	P94			
					0x2	P80			
					0x1	P31			
					0x0	P04			

D7 Reserved

D[6:4] SPT5[2:0]: FPT5 Interrupt Input Port Select Bits

Selects an FPT5 port used for generating port input interrupt 1.

D3 Reserved

D[2:0] SPT4[2:0]: FPT4 Interrupt Input Port Select Bits

Selects an FPT4 port used for generating port input interrupt 1.

Table 26.8.3 Selecting Ports Used For Port Input Interrupt 1

SPT*n*[2:0] setting	FPT4 (SPT4[2:0])	FPT5 (SPT5[2:0])	FPT6 (SPT6[2:0])	FPT7 (SPT7[2:0])
0x7	PE4	P55	PF0	PF1
0x6	PD4	PD5	PD6	PD7
0x5	PB4	PB5	PB6	PB7
0x4	PA4	PA5	PA6	PA7
0x3	P94	P95	P96	P97
0x2	P80	P81	P82	P83
0x1	P31	P32	P33	P60
0x0 (default)	P04	P05	P06	P07

FPT6–7 Interrupt Port Select Register (GPIO_FPT67_SEL)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT6–7 Interrupt Port Select Register (GPIO_FPT67_SEL)	0x300343 (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SPT7[2:0]	FPT7 interrupt input port select	SPT7[2:0]	Port	0x0	R/W	
					0x7	PF1			
					0x6	PD7			
					0x5	PB7			
					0x4	PA7			
					0x3	P97			
					0x2	P83			
					0x1	P60			
					0x0	P07			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SPT6[2:0]	FPT6 interrupt input port select	SPT6[2:0]	Port	0x0	R/W	
					0x7	PF0			
					0x6	PD6			
					0x5	PB6			
					0x4	PA6			
					0x3	P96			
					0x2	P82			
					0x1	P33			
					0x0	P06			

D7 Reserved

D[6:4] SPT7[2:0]: FPT7 Interrupt Input Port Select Bits

Selects an FPT7 port used for generating port input interrupt 1.

D3 Reserved

D[2:0] SPT6[2:0]: FPT6 Interrupt Input Port Select Bits

Selects an FPT6 port used for generating port input interrupt 1.

FPT0–3 Interrupt Polarity Select Register (GPIO_FPT03_POL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPT0–3 Interrupt Polarity Select Register (GPIO_FPT03_POL)	0x300344 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SPPT3	FPT3 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D2	SPPT2	FPT2 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D1	SPPT1	FPT1 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D0	SPPT0	FPT0 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	

D[7:4] Reserved

D[3:0] SPPT[3:0]: FPT[3:0] Input Polarity Select Bits

Selects the interrupt trigger level or edge for the ports used for port input interrupt 0 (FPT0–FPT3).

1 (R/W): High level/Rising edge (default)

0 (R/W): Low level/Falling edge

When SPPT*n* is set to 1 (default), high level (in level trigger mode) or rising edge (in edge trigger mode) is selected as the interrupt generating condition for FPT*n* port.

When SPPT*n* is set to 0, low level (in level trigger mode) or falling edge (in edge trigger mode) is selected.

FPT4–7 Interrupt Polarity Select Register (GPIO_FPT47_POL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPT4–7 Interrupt Polarity Select Register (GPIO_FPT47_POL)	0x300345 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SPPT7	FPT7 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D2	SPPT6	FPT6 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D1	SPPT5	FPT5 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D0	SPPT4	FPT4 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	

D[7:4] Reserved

D[3:0] SPPT[7:4]: FPT[7:4] Input Polarity Select Bits

Selects the interrupt trigger level or edge for the ports used for port input interrupt 1 (FPT4–FPT7).
1 (R/W): High level/Rising edge (default)
0 (R/W): Low level/Falling edge

See the descriptions of SPPT[3:0]/GPIO_FPT03_POL register.

FPT0–3 Interrupt Mode Select Register (GPIO_FPT03_MOD)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPT0–3 Interrupt Mode Select Register (GPIO_FPT03_MOD)	0x300346 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SEPT3	FPT3 interrupt mode select	1	Edge	0	Level	1	R/W	
		D2	SEPT2	FPT2 interrupt mode select	1	Edge	0	Level	1	R/W	
		D1	SEPT1	FPT1 interrupt mode select	1	Edge	0	Level	1	R/W	
		D0	SEPT0	FPT0 interrupt mode select	1	Edge	0	Level	1	R/W	

D[7:4] Reserved

D[3:0] SEPT[3:0]: FPT[3:0] Interrupt Mode Select Bits

Selects trigger modes of the ports used for port input interrupt 0 (FPT0–FPT3).
1 (R/W): Edge trigger mode (default)
0 (R/W): Level trigger mode

When SEPT*n* is set to 1 (default), the corresponding port is set to edge trigger mode. In edge trigger mode, the interrupt flag is set at the active edge of the input signal and it retains 1 until reset via software.
When SEPT*n* is set to 0, the corresponding port is set to level trigger mode. In level trigger mode, the interrupt flag is set when the input signal goes the active level and it retains 1 until reset via software. SLEEP mode can be canceled by causing a port input interrupt regardless of how the GPIO interrupt mode (edge trigger/level trigger) is set.

FPT4–7 Interrupt Mode Select Register (GPIO_FPT47_MOD)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPT4–7 Interrupt Mode Select Register (GPIO_FPT47_MOD)	0x300347 (8 bits)	D7–4	–	reserved	–				–	–	0 when being read.
		D3	SEPT7	FPT7 interrupt mode select	1	Edge	0	Level	1	R/W	
		D2	SEPT6	FPT6 interrupt mode select	1	Edge	0	Level	1	R/W	
		D1	SEPT5	FPT5 interrupt mode select	1	Edge	0	Level	1	R/W	
		D0	SEPT4	FPT4 interrupt mode select	1	Edge	0	Level	1	R/W	

D[7:4] Reserved

D[3:0] SEPT[7:4]: FPT[7:4] Interrupt Mode Select Bits

Selects trigger modes of the ports used for port input interrupt 1 (FPT4–FPT7).
1 (R/W): Edge trigger mode (default)
0 (R/W): Level trigger mode

See the descriptions of SEPT[3:0]/GPIO_FPT03_MOD register.

FPT0–3 Interrupt Mask Register (GPIO_FPT03_MSK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT0–3 Interrupt Mask Register (GPIO_FPT03_MSK)	0x300348 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	SIET3	FPT3 interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	SIET2	FPT2 interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	SIET1	FPT1 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	SIET0	FPT0 interrupt enable	1 Enable 0 Disable	0	R/W	

D[7:4] Reserved

D[3:0] SIET[3:0]: FPT[3:0] Interrupt Enable Bits

Enables or disables the ports to generate port input interrupt 0 (FPT0–FPT3).

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

FPT4–7 Interrupt Mask Register (GPIO_FPT47_MSK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT4–7 Interrupt Mask Register (GPIO_FPT47_MSK)	0x300349 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	SIET7	FPT7 interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	SIET6	FPT6 interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	SIET5	FPT5 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	SIET4	FPT4 interrupt enable	1 Enable 0 Disable	0	R/W	

D[7:4] Reserved

D[3:0] SIET[7:4]: FPT[7:4] Interrupt Enable Bits

Enables or disables the ports to generate port input interrupt 1 (FPT4–FPT7).

1 (R/W): Interrupt enabled

0 (R/W): Interrupt disabled (default)

FPT0–3 Interrupt Flag Register (GPIO_FPT03_FLG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT0–3 Interrupt Flag Register (GPIO_FPT03_FLG)	0x30034a (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	SFGP3	FPT3 interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	X	R/W	Reset by writing 1.
		D2	SFGP2	FPT2 interrupt flag		X	R/W	
		D1	SFGP1	FPT1 interrupt flag		X	R/W	
		D0	SFGP0	FPT0 interrupt flag		X	R/W	

D[7:4] Reserved

D[3:0] SFGP[3:0]: FPT[3:0] Interrupt Flag Bits

These are interrupt flags indicating the port input interrupt 0 (FPT0–FPT3) occurrence status. (Default: undefined)

1 (R): Interrupt cause occurred

0 (R): No interrupt cause occurred

1 (W): Reset flag

0 (W): Ignored

SFGP*n* is the interrupt flag corresponding to the individual ports for port input interrupts and is set to 1 at the specified edge (rising or falling edge) or level (high or low) of the input signal. When the corresponding SIET*n* bit has been set to 1, a port input interrupt request signal is also output to the ITC at the same time. An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied. SFGP*n* is reset by writing 1.

FPT4–7 Interrupt Flag Register (GPIO_FPT47_FLG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT4–7 Interrupt Flag Register (GPIO_FPT47_FLG)	0x30034b (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	SFGP7	FPT7 interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	X	R/W	Reset by writing 1.
		D2	SFGP6	FPT6 interrupt flag		X	R/W	
		D1	SFGP5	FPT5 interrupt flag		X	R/W	
		D0	SFGP4	FPT4 interrupt flag		X	R/W	

D[7:4] Reserved

D[3:0] SFGP[7:4]: FPT[7:4] Interrupt Flag Bits

These are interrupt flags indicating the port input interrupt 1 (FPT4–FPT7) occurrence status. (Default: undefined)

1 (R): Interrupt cause occurred
0 (R): No interrupt cause occurred
1 (W): Reset flag
0 (W): Ignored

See the descriptions of SFGP[3:0]/GPIO_FPT03_FLG register.

FPT0–1 Interrupt Chattering Filter Control Register (GPIO_FPT01_CHAT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT0–1 Interrupt Chattering Filter Control Register (GPIO_FPT01_CHAT)	0x30034c (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SCTP1[2:0]	FPT1 chattering filter time select	SCTP1[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SCTP0[2:0]	FPT0 chattering filter time select	SCTP0[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			

D7 Reserved

D[6:4] SCTP1[2:0]: FPT1 Chattering Filter Time Select Bits

Configures the chattering filter circuit for the FPT1 port.

D3 Reserved

D[2:0] SCTP0[2:0]: FPT0 Chattering Filter Time Select Bits

Configures the chattering filter circuit for the FPT0 port.

The FPT interrupt input ports include a chattering filter circuit that can be disabled or enabled with a filter sampling time specified individually for each FPT port using SCTP*n*[2:0].

Table 26.8.4 Chattering Filter Settings

SCTP*n*[2:0]	Filter sampling time	Invalid pulse (glitch) width that will be filtered	Valid pulse width that will be accepted
0x7	64/fPCLK	< 64/fPCLK × 2	> 64/fPCLK × 4
0x6	32/fPCLK	< 32/fPCLK × 2	> 32/fPCLK × 4
0x5	16/fPCLK	< 16/fPCLK × 2	> 16/fPCLK × 4
0x4	8/fPCLK	< 8/fPCLK × 2	> 8/fPCLK × 4
0x3	4/fPCLK	< 4/fPCLK × 2	> 4/fPCLK × 4
0x2	2/fPCLK	< 2/fPCLK × 2	> 2/fPCLK × 4
0x1	1/fPCLK	< 1/fPCLK × 2	> 1/fPCLK × 4
0x0	Not filtered		

(Default: 0x0)

Notes: • The prescaler (PSC Ch.2) output is used as the filter clock. Make sure the prescaler (PSC Ch.2) is turned on before using the chattering filter. Do not enable the chattering filter when the prescaler (PSC Ch.2) is turned off, as undesired port input interrupts may be generated.

- The chattering filter stops operating in SLEEP mode, as no clock is supplied. In order to cancel SLEEP mode using a port input interrupt, the chattering filter will be automatically bypassed (Not filtered) in SLEEP mode until the CPU exits SLEEP mode even if the chattering filter is set to on.
- Setting the GPIO_FPT*nn*_CHAT register while the interrupt is enabled may generate an undesired port input interrupt. Therefore, the port input interrupt must be disabled before setting the GPIO_FPT*nn*_CHAT register. Furthermore, be sure to clear the port input interrupt flag before enabling the interrupt again after setting the GPIO_FPT*nn*_CHAT register. In this case, clear the interrupt flag after the wait time shown below has elapsed from the GPIO_FPT*nn*_CHAT register setting.

  Wait time [µs] = Filter sampling time [µs] × 4

  Example: When the filter sampling time is 64/f_{PCLK} and f_{PCLK} = 32 MHz
  Wait time = 64 × 4 / 32 = 8 [µs]

  The port input interrupt flag should be cleared after waiting 8 µs or more when the GPIO_FPT*nn*_CHAT register is set to 0x7.

FPT2–3 Interrupt Chattering Filter Control Register (GPIO_FPT23_CHAT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT2–3 Interrupt Chattering Filter Control Register (GPIO_FPT23_CHAT)	0x30034d (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SCTP3[2:0]	FPT3 chattering filter time select	SCTP3[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/f_{PCLK}			
					0x6	32/f_{PCLK}			
					0x5	16/f_{PCLK}			
					0x4	8/f_{PCLK}			
					0x3	4/f_{PCLK}			
					0x2	2/f_{PCLK}			
					0x1	1/f_{PCLK}			
					0x0	None			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SCTP2[2:0]	FPT2 chattering filter time select	SCTP2[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/f_{PCLK}			
					0x6	32/f_{PCLK}			
					0x5	16/f_{PCLK}			
					0x4	8/f_{PCLK}			
					0x3	4/f_{PCLK}			
					0x2	2/f_{PCLK}			
					0x1	1/f_{PCLK}			
					0x0	None			

D7 Reserved

D[6:4] SCTP3[2:0]: FPT3 Chattering Filter Time Select Bits
Configures the chattering filter circuit for the FPT3 port.

D3 Reserved

D[2:0] SCTP2[2:0]: FPT2 Chattering Filter Time Select Bits
Configures the chattering filter circuit for the FPT2 port.

See the descriptions of SCTP0[2:0]/GPIO_FPT01_CHAT register.

FPT4–5 Interrupt Chattering Filter Control Register (GPIO_FPT45_CHAT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT4–5 Interrupt Chattering Filter Control Register (GPIO_FPT45_CHAT)	0x30034e (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SCTP5[2:0]	FPT5 chattering filter time select	SCTP5[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SCTP4[2:0]	FPT4 chattering filter time select	SCTP4[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			

D7 Reserved

D[6:4] SCTP5[2:0]: FPT5 Chattering Filter Time Select Bits
Configures the chattering filter circuit for the FPT5 port.

D3 Reserved

D[2:0] SCTP4[2:0]: FPT4 Chattering Filter Time Select Bits
Configures the chattering filter circuit for the FPT4 port.

See the descriptions of SCTP0[2:0]/GPIO_FPT01_CHAT register.

FPT6–7 Interrupt Chattering Filter Control Register (GPIO_FPT67_CHAT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPT6–7 Interrupt Chattering Filter Control Register (GPIO_FPT67_CHAT)	0x30034f (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SCTP7[2:0]	FPT7 chattering filter time select	SCTP7[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SCTP6[2:0]	FPT6 chattering filter time select	SCTP6[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			

D7 Reserved

D[6:4] SCTP7[2:0]: FPT7 Chattering Filter Time Select Bits
Configures the chattering filter circuit for the FPT7 port.

D3 Reserved

D[2:0] SCTP6[2:0]: FPT6 Chattering Filter Time Select Bits
Configures the chattering filter circuit for the FPT6 port.

See the descriptions of SCTP0[2:0]/GPIO_FPT01_CHAT register.

Port DMA Trigger Source Select Register (GPIO_DMA)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Port DMA Trigger Source Select Register (GPIO_DMA)	0x300350 (8 bits)	D7–3	–	reserved	–		–	–	0 when being read.
		D2–0	SDMAT[2:0]	Port DMA trigger source select	SDMAT[2:0]	Trigger source	0x0	R/W	
					0x7	FPT7			
					0x6	FPT6			
					0x5	FPT5			
					0x4	FPT4			
					0x3	FPT3			
					0x2	FPT2			
					0x1	FPT1			
					0x0	FPT0			

D[7:3] Reserved

D[2:0] SDMAT[2:0]: Port DMA Trigger Source Select Bits

Selects an FPT port as a DMA trigger source to invoke DMA transfer.

Table 26.8.5 DMA Trigger Source Selection

SDMAT[2:0]	Trigger source
0x7	FPT7
0x6	FPT6
0x5	FPT5
0x4	FPT4
0x3	FPT3
0x2	FPT2
0x1	FPT1
0x0	FPT0

(Default: 0x0)

The interrupt signal of the selected FPT line, which is generated according to the interrupt mode and polarity settings regardless of the SIET*n* setting (even if the interrupt is disabled), is sent to the DMAC to trigger a DMA transfer. For more information on DMA transfer, see the "DMA Controller (DMAC)" chapter.

Note: After setting SCTP*n*[2:0]/GPIO_FPT*nn*_CHAT register, the port may send an undesired DMA trigger to the DMAC. Disable the port DMA request for at least the wait time shown below after the GPIO_FPT*nn*_CHAT register is set.

Wait time [µs] = Filter sampling time [µs] × 4

Example: When the filter sampling time is 64/f_{PCLK} and f_{PCLK} = 32 MHz
Wait time = 64 × 4 / 32 = 8 [µs]

The port DMA function can be used after waiting 8 µs or more when the GPIO_FPT*nn*_CHAT register is set to 0x7.

FPK000–007 Interrupt Select Register (GPIO_FPK0_07_SEL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPK000–007 Interrupt Select Register (GPIO_FPK0_07_SEL)	0x300360 (8 bits)	D7	SPK007	FPK007 interrupt select	1	P07 input	0	Disable	0	R/W	
		D6	SPK006	FPK006 interrupt select	1	P06 input	0	Disable	0	R/W	
		D5	SPK005	FPK005 interrupt select	1	P05 input	0	Disable	0	R/W	
		D4	SPK004	FPK004 interrupt select	1	P04 input	0	Disable	0	R/W	
		D3	SPK003	FPK003 interrupt select	1	P03 input	0	Disable	0	R/W	
		D2	SPK002	FPK002 interrupt select	1	P02 input	0	Disable	0	R/W	
		D1	SPK001	FPK001 interrupt select	1	P01 input	0	Disable	0	R/W	
		D0	SPK000	FPK000 interrupt select	1	P00 input	0	Disable	0	R/W	

D[7:0] SPK000–SPK007: FPK000–FPK007 Interrupt Select Bits

Selects whether the FPK port is used for key input interrupt 0 or not.
1 (R/W): Used
0 (R/W): Disabled (not used) (default)

The table below lists the port assignment of the key input interrupt 0 system.

Table 26.8.6 Key Input Interrupt 0 Port Assignment

Key input interrupt group	Key input interrupt source	I/O port
FPK0	FPK000	P00
	FPK001	P01
	FPK002	P02
	FPK003	P03
	FPK004	P04
	FPK005	P05
	FPK006	P06
	FPK007	P07
	FPK008	PA0
	FPK009	PA1
	FPK010	PA2
	FPK011	PA3
	FPK012	PA4
	FPK013	PA5
	FPK014	PA6
	FPK015	PA7

FPK008–015 Interrupt Select Register (GPIO_FPK0_8F_SEL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPK008–015 Interrupt Select Register (GPIO_FPK0_8F_SEL)	**0x300361** (8 bits)	D7	**SPK015**	FPK015 interrupt select	1	PA7 input	0	Disable	0	R/W	
		D6	**SPK014**	FPK014 interrupt select	1	PA6 input	0	Disable	0	R/W	
		D5	**SPK013**	FPK013 interrupt select	1	PA5 input	0	Disable	0	R/W	
		D4	**SPK012**	FPK012 interrupt select	1	PA4 input	0	Disable	0	R/W	
		D3	**SPK011**	FPK011 interrupt select	1	PA3 input	0	Disable	0	R/W	
		D2	**SPK010**	FPK010 interrupt select	1	PA2 input	0	Disable	0	R/W	
		D1	**SPK009**	FPK009 interrupt select	1	PA1 input	0	Disable	0	R/W	
		D0	**SPK008**	FPK008 interrupt select	1	PA0 input	0	Disable	0	R/W	

D[7:0] SPK008–SPK015: FPK008–FPK015 Interrupt Select Bits

Selects whether the FPK port is used for key input interrupt 0 or not.

1 (R/W): Used

0 (R/W): Disabled (not used) (default)

FPK100–107 Interrupt Select Register (GPIO_FPK1_07_SEL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPK100–107 Interrupt Select Register (GPIO_FPK1_07_SEL)	**0x300362** (8 bits)	D7	**SPK107**	FPK107 interrupt select	1	P76 input	0	Disable	0	R/W	
		D6	**SPK106**	FPK106 interrupt select	1	P74 input	0	Disable	0	R/W	
		D5	**SPK105**	FPK105 interrupt select	1	P72 input	0	Disable	0	R/W	
		D4	**SPK104**	FPK104 interrupt select	1	P70 input	0	Disable	0	R/W	
		D3	**SPK103**	FPK103 interrupt select	1	P33 input	0	Disable	0	R/W	
		D2	**SPK102**	FPK102 interrupt select	1	P32 input	0	Disable	0	R/W	
		D1	**SPK101**	FPK101 interrupt select	1	P31 input	0	Disable	0	R/W	
		D0	**SPK100**	FPK100 interrupt select	1	P30 input	0	Disable	0	R/W	

D[7:0] SPK100–SPK107: FPK100–FPK107 Interrupt Select Bits

Selects whether the FPK port is used for key input interrupt 1 or not.

1 (R/W): Used

0 (R/W): Disabled (not used) (default)

The table below lists the port assignment of the key input interrupt 1 system.

Table 26.8.7 Key Input Interrupt 1 Port Assignment

Key input interrupt group	Key input interrupt source	I/O port
FPK1	FPK100	P30
	FPK101	P31
	FPK102	P32
	FPK103	P33
	FPK104	P70
	FPK105	P72
	FPK106	P74
	FPK107	P76
	FPK108	PD0
	FPK109	PD1
	FPK110	PD2
	FPK111	PD3
	FPK112	PD4
	FPK113	PD5
	FPK114	PD6
	FPK115	PD7

FPK108–115 Interrupt Select Register (GPIO_FPK1_8F_SEL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPK108–115 Interrupt Select Register (GPIO_FPK1_ 8F_SEL)	**0x300363** (8 bits)	D7	**SPK115**	FPK115 interrupt select	1	PD7 input	0	Disable	0	R/W	
		D6	**SPK114**	FPK114 interrupt select	1	PD6 input	0	Disable	0	R/W	
		D5	**SPK113**	FPK113 interrupt select	1	PD5 input	0	Disable	0	R/W	
		D4	**SPK112**	FPK112 interrupt select	1	PD4 input	0	Disable	0	R/W	
		D3	**SPK111**	FPK111 interrupt select	1	PD3 input	0	Disable	0	R/W	
		D2	**SPK110**	FPK110 interrupt select	1	PD2 input	0	Disable	0	R/W	
		D1	**SPK109**	FPK109 interrupt select	1	PD1 input	0	Disable	0	R/W	
		D0	**SPK108**	FPK108 interrupt select	1	PD0 input	0	Disable	0	R/W	

D[7:0] SPK108–SPK115: FPK108–FPK115 Interrupt Select Bits

Selects whether the FPK port is used for key input interrupt 1 or not.

1 (R/W): Used

0 (R/W): Disabled (not used) (default)

FPK0–1 Interrupt Polarity Select Register (GPIO_FPK01_POL)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
FPK0–1 Interrupt Polarity Select Register (GPIO_FPK01_ POL)	**0x300370** (8 bits)	D7–2	**–**	reserved	–				–	–	0 when being read.
		D1	**SPKT1**	FPK1 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	
		D0	**SPKT0**	FPK0 input polarity select	1	High / ↑	0	Low / ↓	1	R/W	

D[7:2] Reserved

D1 SPKT1: FPK1 Input Polarity Select Bit

Selects the interrupt trigger level or edge for the ports used for key input interrupt 1 (FPK1).

1 (R/W): High level/Rising edge (default)

0 (R/W): Low level/Falling edge

D0 SPKT0: FPK0 Input Polarity Select Bit

Selects the interrupt trigger level or edge for the ports used for key input interrupt 0 (FPK0).

1 (R/W): High level/Rising edge (default)

0 (R/W): Low level/Falling edge

When SPKT*n* is set to 1 (default), high level (in level trigger mode) or rising edge (in edge trigger mode) is selected as the interrupt generating condition for FPK*n*.

When SPKT*n* is set to 0, low level (in level trigger mode) or falling edge (in edge trigger mode) is selected.

FPK0–1 Interrupt Mode Select Register (GPIO_FPK01_MOD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPK0–1 Interrupt Mode Select Register (GPIO_FPK01_MOD)	0x300371 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	SEKT1	FPK1 interrupt mode select	1 Edge 0 Level	1	R/W	
		D0	SEKT0	FPK0 interrupt mode select	1 Edge 0 Level	1	R/W	

D[7:2] Reserved

D1 SEKT1: FPK1 Interrupt Mode Select Bit

Selects the trigger mode for key input interrupt 1 (FPK1).
1 (R/W): Edge trigger mode (default)
0 (R/W): Level trigger mode

D0 SEKT0: FPK0 Interrupt Mode Select Bit

Selects the trigger mode for key input interrupt 0 (FPK0).
1 (R/W): Edge trigger mode (default)
0 (R/W): Level trigger mode

When SEKT*n* is set to 1 (default), the FPK*n* ports are set to edge trigger mode. In edge trigger mode, the interrupt flag is set at the active edge of the input signal and it retains 1 until reset via software.
When SEKT*n* is set to 0, the FPK*n* ports are set to level trigger mode. In level trigger mode, the interrupt flag is set when the input signal goes the active level and it retains 1 until reset via software. SLEEP mode can be canceled by causing a key input interrupt regardless of how the GPIO interrupt mode (edge trigger/level trigger) is set.

FPK0–1 Interrupt Mask Register (GPIO_FPK01_MSK)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPK0–1 Interrupt Mask Register (GPIO_FPK01_MSK)	0x300372 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	SIEKT1	FPK1 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	SIEKT0	FPK0 interrupt enable	1 Enable 0 Disable	0	R/W	

D[7:2] Reserved

D1 SIEKT1: FPK1 Interrupt Enable Bit

Enables or disables FPK1 interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

D0 SIEKT0: FPK0 Interrupt Enable Bit

Enables or disables FPK0 interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

FPK0–1 Interrupt Flag Register (GPIO_FPK01_FLG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPK0–1 Interrupt Flag Register (GPIO_FPK01_FLG)	0x300373 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	SFGK1	FPK1 interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	X	R/W	
		D0	SFGK0	FPK0 interrupt flag		X	R/W	

D[7:2] Reserved

D1 SFGK1: FPK1 Interrupt Flag Bit

Indicating the FPK1 interrupt occurrence status. (Default: undefined)

1 (R): Interrupt cause occurred
0 (R): No interrupt cause occurred
1 (W): Reset flag
0 (W): Ignored

D0 SFGK0: FPK0 Interrupt Flag Bit

Indicating the FPK0 interrupt occurrence status. (Default: undefined)

1 (R): Interrupt cause occurred
0 (R): No interrupt cause occurred
1 (W): Reset flag
0 (W): Ignored

SFGK*n* is the interrupt flag corresponding to the FPK*n* key input interrupt and is set to 1 at the specified edge (rising or falling edge) or level (high or low) of the input signal. When the corresponding SIEKT*n* bit has been set to 1, a key input interrupt request signal is also output to the ITC at the same time. An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied.
SFGK*n* is reset by writing 1.

FPK0–1 Interrupt Chattering Filter Control Register (GPIO_FPK01_CHAT)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
FPK0–1 Interrupt Chattering Filter Control Register (GPIO_FPK01_CHAT)	0x300374 (8 bits)	D7	–	reserved	–		–	–	0 when being read.
		D6–4	SCTK1[2:0]	FPK1 chattering filter time select	SCTK1[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			
		D3	–	reserved	–		–	–	0 when being read.
		D2–0	SCTK0[2:0]	FPK0 chattering filter time select	SCTK0[2:0]	Filter sampling time	0x0	R/W	
					0x7	64/fPCLK			
					0x6	32/fPCLK			
					0x5	16/fPCLK			
					0x4	8/fPCLK			
					0x3	4/fPCLK			
					0x2	2/fPCLK			
					0x1	1/fPCLK			
					0x0	None			

D7 Reserved

D[6:4] SCTK1[2:0]: FPK1 Chattering Filter Time Select Bits

Configures the chattering filter circuit for the FPK1 key input interrupt ports.

D3 Reserved

D[2:0] SCTK0[2:0]: FPK0 Chattering Filter Time Select Bits

Configures the chattering filter circuit for the FPK0 key input interrupt ports.

The FPK*n* key input interrupt ports include a chattering filter circuit that can be disabled or enabled with a filter sampling time specified individually for each FPK group using SCTK*n*[2:0].

Table 26.8.8 Chattering Filter Settings

SCTK*n*[2:0]	Filter sampling time	Invalid pulse (glitch) width that will be filtered	Valid pulse width that will be accepted
0x7	64/f_{PCLK}	< 64/f_{PCLK} × 2	> 64/f_{PCLK} × 4
0x6	32/f_{PCLK}	< 32/f_{PCLK} × 2	> 32/f_{PCLK} × 4
0x5	16/f_{PCLK}	< 16/f_{PCLK} × 2	> 16/f_{PCLK} × 4
0x4	8/f_{PCLK}	< 8/f_{PCLK} × 2	> 8/f_{PCLK} × 4
0x3	4/f_{PCLK}	< 4/f_{PCLK} × 2	> 4/f_{PCLK} × 4
0x2	2/f_{PCLK}	< 2/f_{PCLK} × 2	> 2/f_{PCLK} × 4
0x1	1/f_{PCLK}	< 1/f_{PCLK} × 2	> 1/f_{PCLK} × 4
0x0	Not filtered		

(Default: 0x0)

Notes: • The prescaler (PSC Ch.2) output is used as the filter clock. Make sure the prescaler (PSC Ch.2) is turned on before using the chattering filter. Do not enable the chattering filter when the prescaler (PSC Ch.2) is turned off, as undesired key input interrupts may be generated.

- The chattering filter stops operating in SLEEP mode, as no clock is supplied. In order to cancel SLEEP mode using a key input interrupt, the chattering filter will be automatically bypassed (Not filtered) in SLEEP mode until the CPU exits SLEEP mode even if the chattering filter is set to on.
- Setting the GPIO_FPK01_CHAT register while the interrupt is enabled may generate an undesired key input interrupt. Therefore, the key input interrupt must be disabled before setting the GPIO_FPK01_CHAT register. Furthermore, be sure to clear the key input interrupt flag before enabling the interrupt again after setting the GPIO_FPK01_CHAT register. In this case, clear the interrupt flag after the wait time shown below has elapsed from the GPIO_FPK01_CHAT register setting.

  Wait time [µs] = Filter sampling time [µs] × 4

  Example: When the filter sampling time is 64/f_{PCLK} and f_{PCLK} = 32 MHz
  Wait time = 64 × 4 / 32 = 8 [µs]

  The key input interrupt flag should be cleared after waiting 8 µs or more when the GPIO_FPK*nn*_CHAT register is set to 0x7.

P0[3:0] Port Function Select Register (PMUX_P0_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P0[3:0] Port Function Select Register (PMUX_P0_03)	0x300800 (8 bits)	D7–6	CFP03[1:0]	P03 port function select	CFP03[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D0			
					0x2	REMC_I			
					0x1	USI_CK0			
					0x0	P03			
		D5–4	CFP02[1:0]	P02 port function select	CFP02[1:0]	Function	0x0	R/W	
					0x3	#HIF_CS			
					0x2	SCLK			
					0x1	USI_CS0			
					0x0	P02			
		D3–2	CFP01[1:0]	P01 port function select	CFP01[1:0]	Function	0x0	R/W	
					0x3	#HIF_RD			
					0x2	SOUT			
					0x1	USI_DO0			
					0x0	P01			
		D1–0	CFP00[1:0]	P00 port function select	CFP00[1:0]	Function	0x0	R/W	
					0x3	#HIF_WR			
					0x2	SIN			
					0x1	USI_DI0			
					0x0	P00			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): HIF_D0 (HIF)
0x2 (R/W): REMC_I (REMC)
0x1 (R/W): USI_CK0 (USI Ch.0)
0x0 (R/W): P03 (GPIO) (default)

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

0x3 (R/W): #HIF_CS (HIF)
0x2 (R/W): SCLK (UART)
0x1 (R/W): USI_CS0 (USI Ch.0)
0x0 (R/W): P02 (GPIO) (default)

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

0x3 (R/W): #HIF_RD (HIF)
0x2 (R/W): SOUT (UART)
0x1 (R/W): USI_DO0 (USI Ch.0)
0x0 (R/W): P01 (GPIO) (default)

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

0x3 (R/W): #HIF_WR (HIF)
0x2 (R/W): SIN (UART)
0x1 (R/W): USI_DI0 (USI Ch.0)
0x0 (R/W): P00 (GPIO) (default)

P0[7:4] Port Function Select Register (PMUX_P0_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P0[7:4] Port Function Select Register (PMUX_P0_47)	**0x300801** (8 bits)	D7–6	**CFP07[1:0]**	P07 port function select	CFP07[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D4			
					0x2	T16A_ATMB_1			
					0x1	USI_CK1			
					0x0	P07			
		D5–4	**CFP06[1:0]**	P06 port function select	CFP06[1:0]	Function	0x0	R/W	
					0x3	HIF_D3			
					0x2	T16A_ATMA_1			
					0x1	USI_CS1			
					0x0	P06			
		D3–2	**CFP05[1:0]**	P05 port function select	CFP05[1:0]	Function	0x0	R/W	
					0x3	HIF_D2			
					0x2	T16A_ATMB_0			
					0x1	USI_DO1			
					0x0	P05			
		D1–0	**CFP04[1:0]**	P04 port function select	CFP04[1:0]	Function	0x0	R/W	
					0x3	HIF_D1			
					0x2	T16A_ATMA_0			
					0x1	USI_DI1			
					0x0	P04			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): HIF_D4 (HIF)
0x2 (R/W): T16A_ATMB_1 (T16A6 Ch.1)
0x1 (R/W): USI_CK1 (USI Ch.1)
0x0 (R/W): P07 (GPIO) (default)

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

0x3 (R/W): HIF_D3 (HIF)
0x2 (R/W): T16A_ATMA_1 (T16A6 Ch.1)
0x1 (R/W): USI_CS1 (USI Ch.1)
0x0 (R/W): P06 (GPIO) (default)

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

0x3 (R/W): HIF_D2 (HIF)
0x2 (R/W): T16A_ATMB_0 (T16A6 Ch.0)
0x1 (R/W): USI_DO1 (USI Ch.1)
0x0 (R/W): P05 (GPIO) (default)

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

0x3 (R/W): HIF_D1 (HIF)
0x2 (R/W): T16A_ATMA_0 (T16A6 Ch.0)
0x1 (R/W): USI_DI1 (USI Ch.1)
0x0 (R/W): P04 (GPIO) (default)

P1[3:0] Port Function Select Register (PMUX_P1_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P1[3:0] Port Function Select Register (PMUX_P1_03)	0x300802 (8 bits)	D7–6	CFP13[1:0]	P13 port function select	CFP13[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D6			
					0x2	PWM_L0			
					0x1	I2S_WSO			
					0x0	P13			
		D5–4	CFP12[1:0]	P12 port function select	CFP12[1:0]	Function	0x0	R/W	
					0x3	HIF_D5			
					0x2	PWM_H0			
					0x1	I2S_SDO			
					0x0	P12			
		D3–2	CFP11[1:0]	P11 port function select	CFP11[1:0]	Function	0x0	R/W	
					0x3	SOUT			
					0x2	P11			
					0x1	TFT_CTL1			
					0x0	A25			
		D1–0	CFP10[1:0]	P10 port function select	CFP10[1:0]	Function	0x0	R/W	
					0x3	CMU_CLK			
					0x2	P10			
					0x1	TFT_CTL0			
					0x0	A24			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): HIF_D6 (HIF)
0x2 (R/W): PWM_L0 (T16P Ch.0)
0x1 (R/W): I2S_WSO (I2S)
0x0 (R/W): P13 (GPIO) (default)

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

0x3 (R/W): HIF_D5 (HIF)
0x2 (R/W): PWM_H0 (T16P Ch.0)
0x1 (R/W): I2S_SDO (I2S)
0x0 (R/W): P12 (GPIO) (default)

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

0x3 (R/W): SOUT (UART)
0x2 (R/W): P11 (GPIO)
0x1 (R/W): TFT_CTL1 (LCDC)
0x0 (R/W): A25 (SRAMC) (default)

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

0x3 (R/W): CMU_CLK (CMU)
0x2 (R/W): P10 (GPIO)
0x1 (R/W): TFT_CTL0 (LCDC)
0x0 (R/W): A24 (SRAMC) (default)

P1[7:4] Port Function Select Register (PMUX_P1_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P1[7:4] Port Function Select Register (PMUX_P1_47)	0x300803 (8 bits)	D7–6	CFP17[1:0]	P17 port function select	CFP17[1:0]	Function	0x0	R/W	Write-protected * When TRACE/MISC_TRACE register = 0 (default)
					0x3	#NAND_RD			
					0x2	REMC_O			
					0x1	#CE5			
					0x0	P17			
					–	DPCO *			
		D5–4	CFP16[1:0]	P16 port function select	CFP16[1:0]	Function	0x0	R/W	
					0x3	#NAND_WR			
					0x2	CMU_CLK			
					0x1	#CE4			
					0x0	P16			
					–	DST1 *			
		D3–2	CFP15[1:0]	P15 port function select	CFP15[1:0]	Function	0x0	R/W	
					0x3	#HIF_WAIT			
					0x2	PWM_L1			
					0x1	I2S_MCLKO			
					0x0	P15			
					–	DST0 *			
		D1–0	CFP14[1:0]	P14 port function select	CFP14[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D7			
					0x2	PWM_H1			
					0x1	I2S_SCLKO			
					0x0	P14			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFP17[1:0]: P17 Port Function Select Bits

0x3 (R/W): #NAND_RD (CARD)
0x2 (R/W): REMC_O (REMC)
0x1 (R/W): #CE5 (SRAMC)
0x0 (R/W): P17 (GPIO) (default)

The P17 pin functions as the DPCO pin for debugging when TRACE/MISC_TRACE register = 0 (default).

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

0x3 (R/W): #NAND_WR (CARD)
0x2 (R/W): CMU_CLK (CMU)
0x1 (R/W): #CE4 (SRAMC)
0x0 (R/W): P16 (GPIO) (default)

The P16 pin functions as the DST1 pin for debugging when TRACE/MISC_TRACE register = 0 (default).

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

0x3 (R/W): #HIF_WAIT (HIF)
0x2 (R/W): PWM_L1 (T16P Ch.1)
0x1 (R/W): I2S_MCLKO (I2S)
0x0 (R/W): P15 (GPIO) (default)

The P15 pin functions as the DST0 pin for debugging when TRACE/MISC_TRACE register = 0 (default).

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

0x3 (R/W): HIF_D7 (HIF)
0x2 (R/W): PWM_H1 (T16P Ch.1)
0x1 (R/W): I2S_SCLKO (I2S)
0x0 (R/W): P14 (GPIO) (default)

P2[1:0] Port Function Select Register (PMUX_P2_01)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P2[1:0] Port Function Select Register (PMUX_P2_01)	0x300804 (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3–2	CFP21[1:0]	P21 port function select	CFP21[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P21			
					0x0	SDCLK			
		D1–0	CFP20[1:0]	P20 port function select	CFP20[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P20			
					0x0	SDCKE			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:4] Reserved

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P21 (GPIO)
0x0 (R/W): SDCLK (SDRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P20 (GPIO)
0x0 (R/W): SDCKE (SDRAMC) (default)

P3[3:0] Port Function Select Register (PMUX_P3_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P3[3:0] Port Function Select Register (PMUX_P3_03)	0x300806 (8 bits)	D7–6	CFP33[1:0]	P33 port function select	CFP33[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_A3			
					0x2	T16A_ATMB_3			
					0x1	I2S_MCLKI			
					0x0	P33			
		D5–4	CFP32[1:0]	P32 port function select	CFP32[1:0]	Function	0x0	R/W	
					0x3	HIF_A2			
					0x2	T16A_ATMA_3			
					0x1	I2S_SCLKI			
					0x0	P32			
		D3–2	CFP31[1:0]	P31 port function select	CFP31[1:0]	Function	0x0	R/W	
					0x3	HIF_A1			
					0x2	T16A_ATMB_2			
					0x1	I2S_WSI			
					0x0	P31			
		D1–0	CFP30[1:0]	P30 port function select	CFP30[1:0]	Function	0x0	R/W	
					0x3	HIF_A0			
					0x2	T16A_ATMA_2			
					0x1	I2S_SDI			
					0x0	P30			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): HIF_A3 (HIF)
0x2 (R/W): T16A_ATMB_3 (T16A6 Ch.3)
0x1 (R/W): I2S_MCLKI (I2S)
0x0 (R/W): P33 (GPIO) (default)

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

0x3 (R/W): HIF_A2 (HIF)
0x2 (R/W): T16A_ATMA_3 (T16A6 Ch.3)
0x1 (R/W): I2S_SCLKI (I2S)
0x0 (R/W): P32 (GPIO) (default)

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

0x3 (R/W): HIF_A1 (HIF)
0x2 (R/W): T16A_ATMB_2 (T16A6 Ch.2)
0x1 (R/W): I2S_WSI (I2S)
0x0 (R/W): P31 (GPIO) (default)

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

0x3 (R/W): HIF_A0 (HIF)
0x2 (R/W): T16A_ATMA_2 (T16A6 Ch.2)
0x1 (R/W): I2S_SDI (I2S)
0x0 (R/W): P30 (GPIO) (default)

P3[6:4] Port Function Select Register (PMUX_P3_46)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P3[6:4] Port Function Select Register (PMUX_P3_46)	0x300807 (8 bits)	D7–6	–	reserved	–		–	–	0 when being read.
		D5–4	CFP36[1:0]	P36 port function select	CFP36[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P36			
					0x0	DST2			
		D3–2	CFP35[1:0]	P35 port function select	CFP35[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P35			
					0x0	DSIO			
		D1–0	CFP34[1:0]	P34 port function select	CFP34[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P34			
					0x0	DCLK			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] Reserved

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P36 (GPIO)
0x0 (R/W): DST2 (DBG) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P35 (GPIO)
0x0 (R/W): DSIO (DBG) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P34 (GPIO)
0x0 (R/W): DCLK (DBG) (default)

P4[2:0] Port Function Select Register (PMUX_P4_02)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P4[2:0] Port Function Select Register (PMUX_P4_02)	0x300808 (8 bits)	D7–6	–	reserved	–		–	–	0 when being read.
		D5–4	CFP42[1:0]	P42 port function select	CFP42[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	TFT_CTL3			
					0x1	P42			
					0x0	A23			
		D3–2	CFP41[1:0]	P41 port function select	CFP41[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	TFT_CTL2			
					0x1	P41			
					0x0	A22			
		D1–0	CFP40[1:0]	P40 port function select	CFP40[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P40			
					0x0	A21			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] Reserved

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

0x3 (R/W): Reserved
0x2 (R/W): TFT_CTL3 (LCDC)
0x1 (R/W): P42 (GPIO)
0x0 (R/W): A23 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): TFT_CTL2 (LCDC)
0x1 (R/W): P41 (GPIO)
0x0 (R/W): A22 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P40 (GPIO)
0x0 (R/W): A21 (SRAMC) (default)

P5[3:0] Port Function Select Register (PMUX_P5_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P5[3:0] Port Function Select Register (PMUX_P5_03)	0x30080a (8 bits)	D7–6	CFP53[1:0]	P53 port function select	CFP53[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P53			
					0x0	#CE10			
		D5–4	CFP52[1:0]	P52 port function select	CFP52[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P52			
					0x0	#CE9			
		D3–2	CFP51[1:0]	P51 port function select	CFP51[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#HIF_INT			
					0x1	P51			
					0x0	#CE8			
		D1–0	CFP50[1:0]	P50 port function select	CFP50[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#SDCS			
					0x1	P50			
					0x0	#CE7			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFP53[1:0]: P53 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P53 (GPIO)
0x0 (R/W): #CE10 (SRAMC) (default)

D[5:4] CFP52[1:0]: P52 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P52 (GPIO)
0x0 (R/W): #CE9 (SRAMC) (default)

D[3:2] CFP51[1:0]: P51 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): #HIF_INT (HIF)
0x1 (R/W): P51 (GPIO)
0x0 (R/W): #CE8 (SRAMC) (default)

D[1:0] CFP50[1:0]: P50 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): #SDCS (SDRAMC)
0x1 (R/W): P50 (GPIO)
0x0 (R/W): #CE7 (SRAMC) (default)

P5[6:4] Port Function Select Register (PMUX_P5_46)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P5[6:4] Port Function Select Register (PMUX_P5_46)	0x30080b (8 bits)	D7–6	–	reserved	–		–	–	0 when being read.
		D5–4	CFP56[1:0]	P56 port function select	CFP56[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P56			
					0x0	#WRH/#BSH			
		D3–2	CFP55[1:0]	P55 port function select	CFP55[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P55			
					0x0	#WRL			
		D1–0	CFP54[1:0]	P54 port function select	CFP54[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P54			
					0x0	#RD			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] Reserved

D[5:4] CFP56[1:0]: P56 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P56 (GPIO)
0x0 (R/W): #WRH/#BSH (SRAMC) (default)

D[3:2] CFP55[1:0]: P55 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P55 (GPIO)
0x0 (R/W): #WRL (SRAMC) (default)

D[1:0] CFP54[1:0]: P54 Port Function Select Bits
0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): P54 (GPIO)
0x0 (R/W): #RD (SRAMC) (default)

P60 Port Function Select Register (PMUX_P6_0)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P60 Port Function Select Register (PMUX_P6_0)	0x30080c (8 bits)	D7–2	–	reserved	–		–	–	0 when being read.
		D1–0	CFP60[1:0]	P60 port function select	CFP60[1:0]	Function	0x0	R/W	Write-protected
					0x3	#WDT_NMI			
					0x2	WDT_CLK			
					0x1	#WAIT			
					0x0	P60			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:2] Reserved

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

0x3 (R/W): #WDT_NMI (WDT)
0x2 (R/W): WDT_CLK (WDT)
0x1 (R/W): #WAIT (SRAMC)
0x0 (R/W): P60 (GPIO) (default)

P7[3:0] Port Function Select Register (PMUX_P7_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P7[3:0] Port Function Select Register (PMUX_P7_03)	0x30080e (8 bits)	D7–6	CFP73[1:0]	P73 port function select	CFP73[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	I2S_SCLKI			
					0x1	AIN3			
					0x0	P73			
		D5–4	CFP72[1:0]	P72 port function select	CFP72[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL1			
					0x1	AIN2			
					0x0	P72			
		D3–2	CFP71[1:0]	P71 port function select	CFP71[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	I2S_WSI			
					0x1	AIN1			
					0x0	P71			
		D1–0	CFP70[1:0]	P70 port function select	CFP70[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	AIN0			
					0x0	P70			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): Reserved
0x2 (R/W): I2S_SCLKI (I2S)
0x1 (R/W): AIN3 (ADC10)
0x0 (R/W): P73 (GPIO) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): EXCL1 (T16A6 Ch. 2/3, T16P Ch.1)
0x1 (R/W): AIN2 (ADC10)
0x0 (R/W): P72 (GPIO) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): I2S_WSI (I2S)
0x1 (R/W): AIN1 (ADC10)
0x0 (R/W): P71 (GPIO) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): EXCL0 (T16A6 Ch.0/1, T16P Ch.0)
0x1 (R/W): AIN0 (ADC10)
0x0 (R/W): P70 (GPIO) (default)

P7[7:4] Port Function Select Register (PMUX_P7_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P7[7:4] Port Function Select Register (PMUX_P7_47)	**0x30080f** (8 bits)	D7–6	**CFP77[1:0]**	P77 port function select	CFP77[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	AIN7			
					0x0	P77			
		D5–4	**CFP76[1:0]**	P76 port function select	CFP76[1:0]	Function	0x0	R/W	
					0x3	#ADTRIG			
					0x2	#WAIT			
					0x1	AIN6			
					0x0	P76			
		D3–2	**CFP75[1:0]**	P75 port function select	CFP75[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	I2S_MCLKI			
					0x1	AIN5			
					0x0	P75			
		D1–0	**CFP74[1:0]**	P74 port function select	CFP74[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	WDT_EXCL			
					0x1	AIN4			
					0x0	P74			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): AIN7 (ADC10)
0x0 (R/W): P77 (GPIO) (default)

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

0x3 (R/W): #ADTRIG (ADC10)
0x2 (R/W): #WAIT (SRAMC)
0x1 (R/W): AIN6 (ADC10)
0x0 (R/W): P76 (GPIO) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): I2S_MCLKI (I2S)
0x1 (R/W): AIN5 (ADC10)
0x0 (R/W): P75 (GPIO) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): WDT_EXCL (WDT)
0x1 (R/W): AIN4 (ADC10)
0x0 (R/W): P74 (GPIO) (default)

P8[3:0] Port Function Select Register (PMUX_P8_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P8[3:0] Port Function Select Register (PMUX_P8_03)	**0x300810** (8 bits)	D7–6	**CFP83[1:0]**	P83 port function select	CFP83[1:0]	Function	0x0	R/W	Write-protected
					0x3	MMCD3			
					0x2	USIL_CK			
					0x1	FPDRDY			
					0x0	P83			
		D5–4	**CFP82[1:0]**	P82 port function select	CFP82[1:0]	Function	0x0	R/W	
					0x3	MMCD2			
					0x2	USIL_CS			
					0x1	FPSHIFT			
					0x0	P82			
		D3–2	**CFP81[1:0]**	P81 port function select	CFP81[1:0]	Function	0x0	R/W	
					0x3	MMCD1			
					0x2	USIL_DO			
					0x1	FPLINE			
					0x0	P81			
		D1–0	**CFP80[1:0]**	P80 port function select	CFP80[1:0]	Function	0x0	R/W	
					0x3	MMCD0			
					0x2	USIL_DI			
					0x1	FPFRAME			
					0x0	P80			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): MMCD3 (SD_MMC)
0x2 (R/W): USIL_CK (USIL)
0x1 (R/W): FPDRDY (LCDC)
0x0 (R/W): P83 (GPIO) (default)

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

0x3 (R/W): MMCD2 (SD_MMC)
0x2 (R/W): USIL_CS (USIL)
0x1 (R/W): FPSHIFT (LCDC)
0x0 (R/W): P82 (GPIO) (default)

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

0x3 (R/W): MMCD1 (SD_MMC)
0x2 (R/W): USIL_DO (USIL)
0x1 (R/W): FPLINE (LCDC)
0x0 (R/W): P81 (GPIO) (default)

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

0x3 (R/W): MMCD0 (SD_MMC)
0x2 (R/W): USIL_DI (USIL)
0x1 (R/W): FPFRAME (LCDC)
0x0 (R/W): P80 (GPIO) (default)

P9[3:0] Port Function Select Register (PMUX_P9_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P9[3:0] Port Function Select Register (PMUX_P9_03)	**0x300812** (8 bits)	D7–6	**CFP93[1:0]**	P93 port function select	CFP93[1:0]	Function	0x0	R/W	Write-protected
					0x3	#NAND_RD			
					0x2	USIL_LCD_D3			
					0x1	FPDAT3			
					0x0	P93			
		D5–4	**CFP92[1:0]**	P92 port function select	CFP92[1:0]	Function	0x0	R/W	
					0x3	#NAND_WR			
					0x2	USIL_LCD_D2			
					0x1	FPDAT2			
					0x0	P92			
		D3–2	**CFP91[1:0]**	P91 port function select	CFP91[1:0]	Function	0x0	R/W	
					0x3	MMCCLK			
					0x2	USIL_LCD_D1			
					0x1	FPDAT1			
					0x0	P91			
		D1–0	**CFP90[1:0]**	P90 port function select	CFP90[1:0]	Function	0x0	R/W	
					0x3	MMCCMD			
					0x2	USIL_LCD_D0			
					0x1	FPDAT0			
					0x0	P90			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): #NAND_RD (CARD)
0x2 (R/W): USIL_LCD_D3 (USIL)
0x1 (R/W): FPDAT3 (LCDC)
0x0 (R/W): P93 (GPIO) (default)

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

0x3 (R/W): #NAND_WR (CARD)
0x2 (R/W): USIL_LCD_D2 (USIL)
0x1 (R/W): FPDAT2 (LCDC)
0x0 (R/W): P92 (GPIO) (default)

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

0x3 (R/W): MMCCLK (SD_MMC)
0x2 (R/W): USIL_LCD_D1 (USIL)
0x1 (R/W): FPDAT1 (LCDC)
0x0 (R/W): P91 (GPIO) (default)

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

0x3 (R/W): MMCCMD (SD_MMC)
0x2 (R/W): USIL_LCD_D0 (USIL)
0x1 (R/W): FPDAT0 (LCDC)
0x0 (R/W): P90 (GPIO) (default)

P9[7:4] Port Function Select Register (PMUX_P9_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P9[7:4] Port Function Select Register (PMUX_P9_47)	**0x300813** (8 bits)	D7–6	**CFP97[1:0]**	P97 port function select	CFP97[1:0]	Function	0x0	R/W	Write-protected
					0x3	USI_CK2			
					0x2	USIL_LCD_D7			
					0x1	FPDAT7			
					0x0	P97			
		D5–4	**CFP96[1:0]**	P96 port function select	CFP96[1:0]	Function	0x0	R/W	
					0x3	USI_CS2			
					0x2	USIL_LCD_D6			
					0x1	FPDAT6			
					0x0	P96			
		D3–2	**CFP95[1:0]**	P95 port function select	CFP95[1:0]	Function	0x0	R/W	
					0x3	USI_DO2			
					0x2	USIL_LCD_D5			
					0x1	FPDAT5			
					0x0	P95			
		D1–0	**CFP94[1:0]**	P94 port function select	CFP94[1:0]	Function	0x0	R/W	
					0x3	USI_DI2			
					0x2	USIL_LCD_D4			
					0x1	FPDAT4			
					0x0	P94			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): USI_CK2 (USI Ch.2)
0x2 (R/W): USIL_LCD_D7 (USIL)
0x1 (R/W): FPDAT7 (LCDC)
0x0 (R/W): P97 (GPIO) (default)

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

0x3 (R/W): USI_CS2 (USI Ch.2)
0x2 (R/W): USIL_LCD_D6 (USIL)
0x1 (R/W): FPDAT6 (LCDC)
0x0 (R/W): P96 (GPIO) (default)

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

0x3 (R/W): USI_DO2 (USI Ch.2)
0x2 (R/W): USIL_LCD_D5 (USIL)
0x1 (R/W): FPDAT5 (LCDC)
0x0 (R/W): P95 (GPIO) (default)

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

0x3 (R/W): USI_DI2 (USI Ch.2)
0x2 (R/W): USIL_LCD_D4 (USIL)
0x1 (R/W): FPDAT4 (LCDC)
0x0 (R/W): P94 (GPIO) (default)

PA[3:0] Port Function Select Register (PMUX_PA_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PA[3:0] Port Function Select Register (PMUX_PA_03)	**0x300814** (8 bits)	D7–6	**CFPA3[1:0]**	PA3 port function select	CFPA3[1:0]	Function	0x0	R/W	Write-protected
					0x3	FPDAT19			
					0x2	MMCD3			
					0x1	HIF_D11			
					0x0	PA3			
		D5–4	**CFPA2[1:0]**	PA2 port function select	CFPA2[1:0]	Function	0x0	R/W	
					0x3	FPDAT18			
					0x2	MMCD2			
					0x1	HIF_D10			
					0x0	PA2			
		D3–2	**CFPA1[1:0]**	PA1 port function select	CFPA1[1:0]	Function	0x0	R/W	
					0x3	FPDAT17			
					0x2	MMCD1			
					0x1	HIF_D9			
					0x0	PA1			
		D1–0	**CFPA0[1:0]**	PA0 port function select	CFPA0[1:0]	Function	0x0	R/W	
					0x3	FPDAT16			
					0x2	MMCD0			
					0x1	HIF_D8			
					0x0	PA0			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): FPDAT19 (LCDC)
0x2 (R/W): MMCD3 (SD_MMC)
0x1 (R/W): HIF_D11 (HIF)
0x0 (R/W): PA3 (GPIO) (default)

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

0x3 (R/W): FPDAT18 (LCDC)
0x2 (R/W): MMCD2 (SD_MMC)
0x1 (R/W): HIF_D10 (HIF)
0x0 (R/W): PA2 (GPIO) (default)

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

0x3 (R/W): FPDAT17 (LCDC)
0x2 (R/W): MMCD1 (SD_MMC)
0x1 (R/W): HIF_D9 (HIF)
0x0 (R/W): PA1 (GPIO) (default)

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

0x3 (R/W): FPDAT16 (LCDC)
0x2 (R/W): MMCD0 (SD_MMC)
0x1 (R/W): HIF_D8 (HIF)
0x0 (R/W): PA0 (GPIO) (default)

PA[7:4] Port Function Select Register (PMUX_PA_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PA[7:4] Port Function Select Register (PMUX_PA_47)	**0x300815** (8 bits)	D7–6	**CFPA7[1:0]**	PA7 port function select	CFPA7[1:0]	Function	0x0	R/W	Write-protected
					0x3	FPDAT23			
					0x2	T16A_ATMB_0			
					0x1	HIF_D15			
					0x0	PA7			
		D5–4	**CFPA6[1:0]**	PA6 port function select	CFPA6[1:0]	Function	0x0	R/W	
					0x3	FPDAT22			
					0x2	T16A_ATMA_0			
					0x1	HIF_D14			
					0x0	PA6			
		D3–2	**CFPA5[1:0]**	PA5 port function select	CFPA5[1:0]	Function	0x0	R/W	
					0x3	FPDAT21			
					0x2	MMCCLK			
					0x1	HIF_D13			
					0x0	PA5			
		D1–0	**CFPA4[1:0]**	PA4 port function select	CFPA4[1:0]	Function	0x0	R/W	
					0x3	FPDAT20			
					0x2	MMCCMD			
					0x1	HIF_D12			
					0x0	PA4			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFPA7[1:0]: PA7 Port Function Select Bits

0x3 (R/W): FPDAT23 (LCDC)
0x2 (R/W): T16A_ATMB_0 (T16A6 Ch.0)
0x1 (R/W): HIF_D15 (HIF)
0x0 (R/W): PA7 (GPIO) (default)

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

0x3 (R/W): FPDAT22 (LCDC)
0x2 (R/W): T16A_ATMA_0 (T16A6 Ch.0)
0x1 (R/W): HIF_D14 (HIF)
0x0 (R/W): PA6 (GPIO) (default)

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

0x3 (R/W): FPDAT21 (LCDC)
0x2 (R/W): MMCCLK (SD_MMC)
0x1 (R/W): HIF_D13 (HIF)
0x0 (R/W): PA5 (GPIO) (default)

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

0x3 (R/W): FPDAT20 (LCDC)
0x2 (R/W): MMCCMD (SD_MMC)
0x1 (R/W): HIF_D12 (HIF)
0x0 (R/W): PA4 (GPIO) (default)

PB[3:0] Port Function Select Register (PMUX_PB_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PB[3:0] Port Function Select Register (PMUX_PB_03)	**0x300816** (8 bits)	D7–6	**CFPB3[1:0]**	PB3 port function select	CFPB3[1:0]	Function	0x0	R/W	Write-protected
					0x3	MMCD3			
					0x2	REMC_I			
					0x1	FPDAT11			
					0x0	PB3			
		D5–4	**CFPB2[1:0]**	PB2 port function select	CFPB2[1:0]	Function	0x0	R/W	
					0x3	MMCD2			
					0x2	REMC_O			
					0x1	FPDAT10			
					0x0	PB2			
		D3–2	**CFPB1[1:0]**	PB1 port function select	CFPB1[1:0]	Function	0x0	R/W	
					0x3	MMCD1			
					0x2	#NAND_RD			
					0x1	FPDAT9			
					0x0	PB1			
		D1–0	**CFPB0[1:0]**	PB0 port function select	CFPB0[1:0]	Function	0x0	R/W	
					0x3	MMCD0			
					0x2	#NAND_WR			
					0x1	FPDAT8			
					0x0	PB0			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): MMCD3 (SD_MMC)
0x2 (R/W): REMC_I (REMC)
0x1 (R/W): FPDAT11 (LCDC)
0x0 (R/W): PB3 (GPIO) (default)

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

0x3 (R/W): MMCD2 (SD_MMC)
0x2 (R/W): REMC_O (REMC)
0x1 (R/W): FPDAT10 (LCDC)
0x0 (R/W): PB2 (GPIO) (default)

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

0x3 (R/W): MMCD1 (SD_MMC)
0x2 (R/W): #NAND_RD (CARD)
0x1 (R/W): FPDAT9 (LCDC)
0x0 (R/W): PB1 (GPIO) (default)

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

0x3 (R/W): MMCD0 (SD_MMC)
0x2 (R/W): #NAND_WR (CARD)
0x1 (R/W): FPDAT8 (LCDC)
0x0 (R/W): PB0 (GPIO) (default)

PB[7:4] Port Function Select Register (PMUX_PB_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PB[7:4] Port Function Select Register (PMUX_PB_47)	**0x300817** (8 bits)	D7–6	**CFPB7[1:0]**	PB7 port function select	CFPB7[1:0]	Function	0x0	R/W	Write-protected
					0x3	T16A_ATMB_1			
					0x2	USI_CK2			
					0x1	FPDAT15			
					0x0	PB7			
		D5–4	**CFPB6[1:0]**	PB6 port function select	CFPB6[1:0]	Function	0x0	R/W	
					0x3	T16A_ATMA_1			
					0x2	USI_CS2			
					0x1	FPDAT14			
					0x0	PB6			
		D3–2	**CFPB5[1:0]**	PB5 port function select	CFPB5[1:0]	Function	0x0	R/W	
					0x3	MMCCLK			
					0x2	USI_DO2			
					0x1	FPDAT13			
					0x0	PB5			
		D1–0	**CFPB4[1:0]**	PB4 port function select	CFPB4[1:0]	Function	0x0	R/W	
					0x3	MMCCMD			
					0x2	USI_DI2			
					0x1	FPDAT12			
					0x0	PB4			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): T16A_ATMB_1 (T16A6 Ch.1)
0x2 (R/W): USI_CK2 (USI Ch.2)
0x1 (R/W): FPDAT15 (LCDC)
0x0 (R/W): PB7 (GPIO) (default)

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

0x3 (R/W): T16A_ATMA_1 (T16A6 Ch.1)
0x2 (R/W): USI_CS2 (USI Ch.2)
0x1 (R/W): FPDAT14 (LCDC)
0x0 (R/W): PB6 (GPIO) (default)

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

0x3 (R/W): MMCCLK (SD_MMC)
0x2 (R/W): USI_DO2 (USI Ch.2)
0x1 (R/W): FPDAT13 (LCDC)
0x0 (R/W): PB5 (GPIO) (default)

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

0x3 (R/W): MMCCMD (SD_MMC)
0x2 (R/W): USI_DI2 (USI Ch.2)
0x1 (R/W): FPDAT12 (LCDC)
0x0 (R/W): PB4 (GPIO) (default)

PC[3:0] Port Function Select Register (PMUX_PC_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PC[3:0] Port Function Select Register (PMUX_PC_03)	**0x300818** (8 bits)	D7–6	**CFPC3[1:0]**	PC3 port function select	CFPC3[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	PC3			
					0x0	D11			
		D5–4	**CFPC2[1:0]**	PC2 port function select	CFPC2[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC2			
					0x0	D10			
		D3–2	**CFPC1[1:0]**	PC1 port function select	CFPC1[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC1			
					0x0	D9			
		D1–0	**CFPC0[1:0]**	PC0 port function select	CFPC0[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC0			
					0x0	D8			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC3 (GPIO)
0x0 (R/W): D11 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC2 (GPIO)
0x0 (R/W): D10 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC1 (GPIO)
0x0 (R/W): D9 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC0 (GPIO)
0x0 (R/W): D8 (SRAMC) (default)

PC[7:4] Port Function Select Register (PMUX_PC_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PC[7:4] Port Function Select Register (PMUX_PC_47)	**0x300819** (8 bits)	D7–6	**CFPC7[1:0]**	PC7 port function select	CFPC7[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	PC7			
					0x0	D15			
		D5–4	**CFPC6[1:0]**	PC6 port function select	CFPC6[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC6			
					0x0	D14			
		D3–2	**CFPC5[1:0]**	PC5 port function select	CFPC5[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC5			
					0x0	D13			
		D1–0	**CFPC4[1:0]**	PC4 port function select	CFPC4[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC4			
					0x0	D12			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC7 (GPIO)
0x0 (R/W): D15 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC6 (GPIO)
0x0 (R/W): D14 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC5 (GPIO)
0x0 (R/W): D13 (SRAMC) (default)

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

0x3 (R/W): Reserved
0x2 (R/W): Reserved
0x1 (R/W): PC4 (GPIO)
0x0 (R/W): D12 (SRAMC) (default)

PD[3:0] Port Function Select Register (PMUX_PD_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PD[3:0] Port Function Select Register (PMUX_PD_03)	**0x30081a** (8 bits)	D7–6	**CFPD3[1:0]**	PD3 port function select	CFPD3[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	USI_CK0			
					0x1	FPDAT19			
					0x0	PD3			
		D5–4	**CFPD2[1:0]**	PD2 port function select	CFPD2[1:0]	Function	0x0	R/W	
					0x3	SCLK			
					0x2	USI_CS0			
					0x1	FPDAT18			
					0x0	PD2			
		D3–2	**CFPD1[1:0]**	PD1 port function select	CFPD1[1:0]	Function	0x0	R/W	
					0x3	SOUT			
					0x2	USI_DO0			
					0x1	FPDAT17			
					0x0	PD1			
		D1–0	**CFPD0[1:0]**	PD0 port function select	CFPD0[1:0]	Function	0x0	R/W	
					0x3	SIN			
					0x2	USI_DI0			
					0x1	FPDAT16			
					0x0	PD0			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFPD3[1:0]: PD3 Port Function Select Bits

0x3 (R/W): Reserved
0x2 (R/W): USI_CK0 (USI Ch.0)
0x1 (R/W): FPDAT19 (LCDC)
0x0 (R/W): PD3 (GPIO) (default)

D[5:4] CFPD2[1:0]: PD2 Port Function Select Bits

0x3 (R/W): SCLK (UART)
0x2 (R/W): USI_CS0 (USI Ch.0)
0x1 (R/W): FPDAT18 (LCDC)
0x0 (R/W): PD2 (GPIO) (default)

D[3:2] CFPD1[1:0]: PD1 Port Function Select Bits

0x3 (R/W): SOUT (UART)
0x2 (R/W): USI_DO0 (USI Ch.0)
0x1 (R/W): FPDAT17 (LCDC)
0x0 (R/W): PD1 (GPIO) (default)

D[1:0] CFPD0[1:0]: PD0 Port Function Select Bits

0x3 (R/W): SIN (UART)
0x2 (R/W): USI_DI0 (USI Ch.0)
0x1 (R/W): FPDAT16 (LCDC)
0x0 (R/W): PD0 (GPIO) (default)

PD[7:4] Port Function Select Register (PMUX_PD_47)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PD[7:4] Port Function Select Register (PMUX_PD_47)	**0x30081b** (8 bits)	D7–6	**CFPD7[1:0]**	PD7 port function select	CFPD7[1:0]	Function	0x0	R/W	Write-protected
					0x3	T16A_ATMB_0			
					0x2	USI_CK1			
					0x1	FPDAT23			
					0x0	PD7			
		D5–4	**CFPD6[1:0]**	PD6 port function select	CFPD6[1:0]	Function	0x0	R/W	
					0x3	T16A_ATMA_0			
					0x2	USI_CS1			
					0x1	FPDAT22			
					0x0	PD6			
		D3–2	**CFPD5[1:0]**	PD5 port function select	CFPD5[1:0]	Function	0x0	R/W	
					0x3	#NAND_RD			
					0x2	USI_DO1			
					0x1	FPDAT21			
					0x0	PD5			
		D1–0	**CFPD4[1:0]**	PD4 port function select	CFPD4[1:0]	Function	0x0	R/W	
					0x3	#NAND_WR			
					0x2	USI_DI1			
					0x1	FPDAT20			
					0x0	PD4			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFPD7[1:0]: PD7 Port Function Select Bits

0x3 (R/W): T16A_ATMB_0 (T16A6 Ch.0)
0x2 (R/W): USI_CK1 (USI Ch.1)
0x1 (R/W): FPDAT23 (LCDC)
0x0 (R/W): PD7 (GPIO) (default)

D[5:4] CFPD6[1:0]: PD6 Port Function Select Bits

0x3 (R/W): T16A_ATMA_0 (T16A6 Ch.0)
0x2 (R/W): USI_CS1 (USI Ch.1)
0x1 (R/W): FPDAT22 (LCDC)
0x0 (R/W): PD6 (GPIO) (default)

D[3:2] CFPD5[1:0]: PD5 Port Function Select Bits

0x3 (R/W): #NAND_RD (CARD)
0x2 (R/W): USI_DO1 (USI Ch.1)
0x1 (R/W): FPDAT21 (LCDC)
0x0 (R/W): PD5 (GPIO) (default)

D[1:0] CFPD4[1:0]: PD4 Port Function Select Bits

0x3 (R/W): #NAND_WR (CARD)
0x2 (R/W): USI_DI1 (USI Ch.1)
0x1 (R/W): FPDAT20 (LCDC)
0x0 (R/W): PD4 (GPIO) (default)

PE[3:0] Port Function Select Register (PMUX_PE_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PE[3:0] Port Function Select Register (PMUX_PE_03)	0x30081c (8 bits)	D7–6	CFPE3[1:0]	PE3 port function select	CFPE3[1:0]	Function	0x0	R/W	Write-protected
					0x3	USI_CK2			
					0x2	T16A_ATMB_3			
					0x1	MMCD3			
					0x0	PE3			
		D5–4	CFPE2[1:0]	PE2 port function select	CFPE2[1:0]	Function	0x0	R/W	
					0x3	USI_CS2			
					0x2	T16A_ATMA_3			
					0x1	MMCD2			
					0x0	PE2			
		D3–2	CFPE1[1:0]	PE1 port function select	CFPE1[1:0]	Function	0x0	R/W	
					0x3	USI_DO2			
					0x2	T16A_ATMB_2			
					0x1	MMCD1			
					0x0	PE1			
		D1–0	CFPE0[1:0]	PE0 port function select	CFPE0[1:0]	Function	0x0	R/W	
					0x3	USI_DI2			
					0x2	T16A_ATMA_2			
					0x1	MMCD0			
					0x0	PE0			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFPE3[1:0]: PE3 Port Function Select Bits

0x3 (R/W): USI_CK2 (USI Ch.2)
0x2 (R/W): T16A_ATMB_3 (T16A6 Ch.3)
0x1 (R/W): MMCD3 (SD_MMC)
0x0 (R/W): PE3 (GPIO) (default)

D[5:4] CFPE2[1:0]: PE2 Port Function Select Bits

0x3 (R/W): USI_CS2 (USI Ch.2)
0x2 (R/W): T16A_ATMA_3 (T16A6 Ch.3)
0x1 (R/W): MMCD2 (SD_MMC)
0x0 (R/W): PE2 (GPIO) (default)

D[3:2] CFPE1[1:0]: PE1 Port Function Select Bits

0x3 (R/W): USI_DO2 (USI Ch.2)
0x2 (R/W): T16A_ATMB_2 (T16A6 Ch.2)
0x1 (R/W): MMCD1 (SD_MMC)
0x0 (R/W): PE1 (GPIO) (default)

D[1:0] CFPE0[1:0]: PE0 Port Function Select Bits

0x3 (R/W): USI_DI2 (USI Ch.2)
0x2 (R/W): T16A_ATMA_2 (T16A6 Ch.2)
0x1 (R/W): MMCD0 (SD_MMC)
0x0 (R/W): PE0 (GPIO) (default)

PE[5:4] Port Function Select Register (PMUX_PE_45)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PE[5:4] Port Function Select Register (PMUX_PE_45)	0x30081d (8 bits)	D7–4	–	reserved	–		–	–	0 when being read.
		D3–2	CFPE5[1:0]	PE5 port function select	CFPE5[1:0]	Function	0x0	R/W	Write-protected
					0x3	REMC_I			
					0x2	T16A_ATMB_1			
					0x1	MMCCLK			
					0x0	PE5			
		D1–0	CFPE4[1:0]	PE4 port function select	CFPE4[1:0]	Function	0x0	R/W	
					0x3	REMC_O			
					0x2	T16A_ATMA_1			
					0x1	MMCCMD			
					0x0	PE4			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:4] Reserved

D[3:2] CFPE5[1:0]: PE5 Port Function Select Bits

0x3 (R/W): REMC_I (REMC)
0x2 (R/W): T16A_ATMB_1 (T16A6 Ch.1)
0x1 (R/W): MMCCLK (SD_MMC)
0x0 (R/W): PE5 (GPIO) (default)

D[1:0] CFPE4[1:0]: PE4 Port Function Select Bits

0x3 (R/W): REMC_O (REMC)
0x2 (R/W): T16A_ATMA_1 (T16A6 Ch.1)
0x1 (R/W): MMCCMD (SD_MMC)
0x0 (R/W): PE4 (GPIO) (default)

PF[3:0] Port Function Select Register (PMUX_PF_03)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PF[3:0] Port Function Select Register (PMUX_PF_03)	0x30081e (8 bits)	D7–6	CFPF3[1:0]	PF3 port function select	CFPF3[1:0]	Function	0x0	R/W	Write-protected
					0x3	TFT_CTL3			
					0x2	PWM_L1			
					0x1	I2S_MCLKO			
					0x0	PF3			
		D5–4	CFPF2[1:0]	PF2 port function select	CFPF2[1:0]	Function	0x0	R/W	
					0x3	TFT_CTL2			
					0x2	PWM_H1			
					0x1	I2S_SCLKO			
					0x0	PF2			
		D3–2	CFPF1[1:0]	PF1 port function select	CFPF1[1:0]	Function	0x0	R/W	
					0x3	TFT_CTL1			
					0x2	PWM_L0			
					0x1	I2S_WSO			
					0x0	PF1			
		D1–0	CFPF0[1:0]	PF0 port function select	CFPF0[1:0]	Function	0x0	R/W	
					0x3	TFT_CTL0			
					0x2	PWM_H0			
					0x1	I2S_SDO			
					0x0	PF0			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:6] CFPF3[1:0]: PF3 Port Function Select Bits

0x3 (R/W): TFT_CTL3 (LCDC)
0x2 (R/W): PWM_L1 (T16P Ch.1)
0x1 (R/W): I2S_MCLKO (I2S)
0x0 (R/W): PF3 (GPIO) (default)

D[5:4] CFPF2[1:0]: PF2 Port Function Select Bits

0x3 (R/W): TFT_CTL2 (LCDC)
0x2 (R/W): PWM_H1 (T16P Ch.1)
0x1 (R/W): I2S_SCLKO (I2S)
0x0 (R/W): PF2 (GPIO) (default)

D[3:2] CFPF1[1:0]: PF1 Port Function Select Bits

0x3 (R/W): TFT_CTL1 (LCDC)
0x2 (R/W): PWM_L0 (T16P Ch.0)
0x1 (R/W): I2S_WSO (I2S)
0x0 (R/W): PF1 (GPIO) (default)

D[1:0] CFPF0[1:0]: PF0 Port Function Select Bits

0x3 (R/W): TFT_CTL0 (LCDC)
0x2 (R/W): PWM_H0 (T16P Ch.0)
0x1 (R/W): I2S_SDO (I2S)
0x0 (R/W): PF0 (GPIO) (default)

PF4 Port Function Select Register (PMUX_PF_4)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PF4 Port Function Select Register (PMUX_PF_4)	0x30081f (8 bits)	D7–2	–	reserved	–		–	–	0 when being read.
		D1–0	CFPF4[1:0]	PF4 port function select	CFPF4[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	#ADTRIG			
					0x1	I2S_SDI			
					0x0	PF4			

The GPIO pins are shared with the peripheral module pins. This register is used to select how the pins are used.

D[7:2] Reserved

D[1:0] CFPF4[1:0]: PF4 Port Function Select Bits

0x3 (R/W): Reserved
0x2 (R/W): #ADTRIG
0x1 (R/W): I2S_SDI (I2S)
0x0 (R/W): PF4 (GPIO) (default)

P*x* Port Pull-up Control Registers (GPIO_P*x*_PUP)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
P*x* Port Pull-up Control Register (GPIO_P*x*_PUP)	0x300820 | 0x30082f (8 bits)	D7–0	PUP*x*[7:0]	P*x*[7:0] port pull-up enable	1	Enable	0	Disable	*	R/W	Write-protected

Notes: • The PUP*xy* bits for unavailable pins are read only bits from which 0 is always read out.

• The PC[7:0] ports does not include a pull-up resistor.

D[7:0] PUP*x*[7:0]: P*x*[7:0] Port Pull-Up Enable Bits

Enables or disables the pull-up resistor for the I/O port pin.
1 (R/W): Enabled
0 (R/W): Disabled

PUP*xy* is the pull-up enable bit that corresponds directly to P*xy* port. Setting to 1 enables the pull-up resistor so that the port pin will be pulled up when the port is set to input mode.
When the port is in output mode, the port pin is not pulled up even if PUP*xy* is set to 1.
The pull-up register is disabled when PUP*xy* is set to 0.
This control is also effective when the port is used for a peripheral module function.
The table below shows the initial pull-up settings.

Table 26.8.9 Initial Pull-Up Status

Port	P*x*0	P*x*1	P*x*2	P*x*3	P*x*4	P*x*5	P*x*6	P*x*7
P0	–	–	–	–	–	–	–	–
P1	–	–	–	–	–	–	–	–
P2	–	–						
P3	–	–	–	–	–	Enabled	–	
P4	–	–	–					
P5	–	–	–	–	–	–	–	
P6	Enabled							
P7	–	–	–	–	–	–	Enabled	–
P8	–	–	–	–				
P9	–	–	–	–	–	–	–	–
PA	–	–	–	–	–	–	–	–
PB	–	–	–	–	–	–	–	–
PC	No pull-up resistor included (bus hold latch)							
PD	–	–	–	–	–	–	–	–
PE	–	–	–	–	–	–	–	–
PF	–	–	–	–	–	–	–	–

–: Disabled

Bus Drive Control Register (GPIO_BUS_DRV)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Bus Drive Control Register (GPIO_BUS_DRV)	0x300830 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	LDRVDB	D[15:0] low drive	1 Low drive 0 Normal output	0	R/W	Write-protected
		D0	LDRVAD	A[25:0] low drive		0	R/W	

D[7:2] Reserved

D1 LDRVDB: D[15:0] Low Drive Bit

Drives the data bus signals forcibly low.
1 (R/W): Low drive
0 (R/W): Normal output (default)

When LDRVDB is set to 1, the D[15:0] signals are forcibly driven low. When it is set to 0, the signals are controlled by the SRAMC/SDRAMC normally.

D0 LDRVAD: A[25:0] Low Drive Bit

Drives the address bus signals forcibly low.
1 (R/W): Low drive
0 (R/W): Normal output (default)

When LDRVAD is set to 1, the A[25:0] signals are forcibly driven low. When it is set to 0, the signals are controlled by the SRAMC/SDRAMC normally.

Port Noise Filter Control Register (GPIO_FILTER)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Port Noise Filter Control Register (GPIO_FILTER)	0x30083e (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	ANFEN	Input port noise filter enable	1 Enable 0 Disable	0	R/W	Write-protected

D[7:1] Reserved

D0 ANFEN: Input Port Noise Filter Enable Bit

Enables or disables the noise filters for peripheral input ports.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Setting 1 to ANFEN enables the noise filters to remove noise on the signals input from the ports shown below.

USI: USI_DI0, USI_DI1, USI_DI2, USI_CS0, USI_CS1, USI_CS2, USI_CK0, USI_CK1, USI_CK2
USIL: USIL_DI, USIL_CS, USIL_CK
UART: SIN, SCLK
REMC: REMC_I
I2S: I2S_SDI, I2S_WSI, I2S_SCLKI, I2S_MCLKI
T16A6: EXCL0, EXCL1, T16A_ATMA_0, T16A_ATMA_1, T16A_ATMA_2, T16A_ATMA_3, T16A_ATMB_0, T16A_ATMB_1, T16A_ATMB_2, T16A_ATMB_3
ADC10: #ADTRIG
WDT: WDT_EXCL
SD_MMC: MMCD[3:0], MMCCMD
GPIO: Ports used for FPT0–FPT7 port input interrupts and FPK0–FPK1 key input interrupts (See note below.)

When ANFEN is set to 0 (default), the input signals bypass the noise filters.

Notes: • These noise filters cannot be enabled individually.

• The noise filters are not effective if these ports are used as general-purpose input port. However, the noise filters for the general-purpose input ports that are selected as FPT interrupt ports (FPT0 to FPT7) or FPK interrupt ports (FPK0 and FPK1) are effective.

GPIO/PMUX Write Protect Register (GPIO_PROTECT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
GPIO/PMUX Write Protect Register (GPIO_ PROTECT)	**0x30083f** (8 bits)	D7–0	**PPROT[7:0]**	GPIO/PMUX register protect flag	Writing 10010110 (0x96) removes the write protection of the GPIO and PMUX registers (0x300800–0x300830 and 0x30083e). Writing another value set the write protection.	0x0	R/W	

D[7:0] PPROT[7:0]: GPIO/PMUX Register Protect Flag Bits

Enables or disables write protection of the GPIO and PMUX registers (0x300800–0x300830 and 0x30083e).

0x96 (R/W): Disable write protection

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

Before altering any GPIO/PMUX register within addresses 0x300800–0x300830 and 0x30083e, write data 0x96 to PPROT[7:0] to disable write protection. If PPROT[7:0] is set to other than 0x96, even if an attempt is made to alter any GPIO/PMUX register by executing a write instruction, the content of the register will not be altered even though the instruction may have been executed without a problem. Once PPROT[7:0] is set to 0x96, the GPIO/PMUX registers can be rewritten any number of times until being reset to other than 0x96. When rewriting the GPIO/PMUX registers has finished, PPROT[7:0] should be set to other than 0x96 to prevent accidental writing to the GPIO/PMUX registers.

27 A/D Converter (ADC10)

27.1 ADC10 Module Overview

The S1C33L27 incorporates an A/D converter with the following features:

- Conversion method: Successive approximation type
- Resolution: 10 bits
- Input channels: Max. 8 channels (Chip and PFBGA12U-180 package)
  Max. 4 channels (TQFP24/QFP20-144pin and TQFP15-128pin package)
- A/D conversion clock: Max. 2 MHz
- Sampling rate: Max. 100 ksps
- Analog input voltage range: Vss to AVDD
- Sampling & hold circuit included
- Supports two conversion modes: One-time conversion mode
  (for single channel or multi-channels)
  Continuous conversion mode
  (for single channel or multi-channels, terminated with software)
- Supports three conversion triggers: Software trigger
  External trigger (input from the #ADTRIG pin)
  Fine mode 8-bit timer Ch.4 underflow trigger
- The conversion results can be read as 16-bit data with the 10-bit converted data aligned to left or right.
- Two types of interrupts can be generated: Conversion completion interrupt
  Conversion data overwrite error interrupt

Figure 27.1.1 shows the A/D converter configuration.

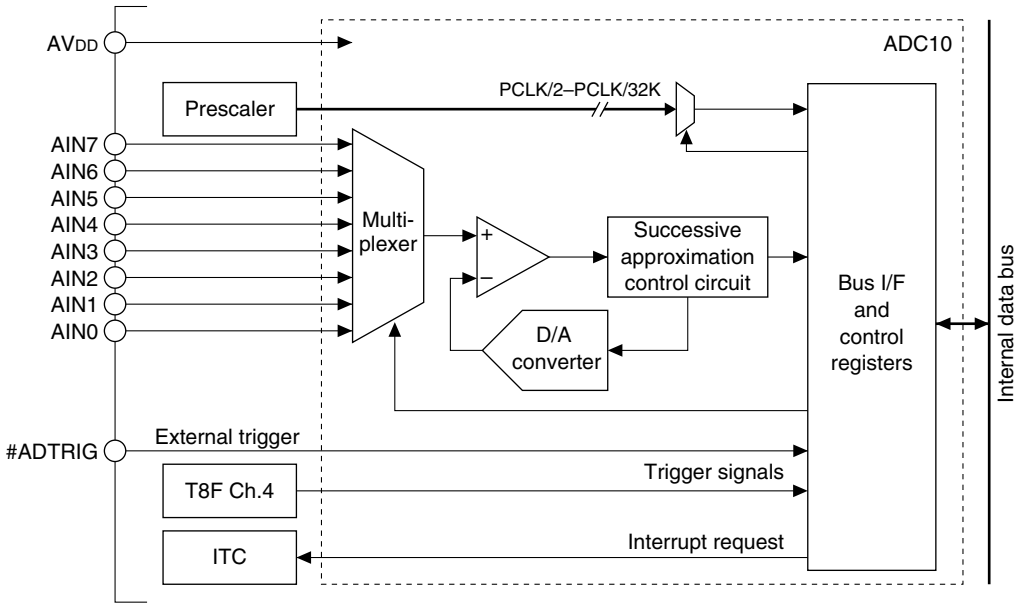


Figure 27.1.1 A/D Converter Configuration

27.2 ADC10 Input Pins

Table 27.2.1 lists the A/D converter input pins.

Table 27.2.1 List of A/D Converter Input Pins

Pin name	I/O	Qty	Function
AIN[7:0] *1 AIN0/2/4/6 *2	I	8 *1 4 *2	Analog signal input pins Input the analog signals to be A/D converted. The analog input voltage AV_{IN} must be within the range of $V_{SS} \leq AV_{IN} \leq AV_{DD}$. *1: AIN0 (Ch.0) to AIN7 (Ch.7) (Chip and PFBGA12U-180 package) *2: AIN0 (Ch.0), AIN2 (Ch.2), AIN4 (Ch.4), AIN6 (Ch.6) (TQFP24/QFP20-144pin and TQFP15-128pin package)
#ADTRIG	I	1	External trigger input pin Input a trigger signal to start A/D conversion from an external source.
AV_{DD}	–	1	Analog power-supply pin Always supply the HV_{DD} voltage even if the A/D converter is not used.

The A/D converter input pins (AIN[7:0], #ADTRIG) are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as A/D converter input pins.
For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

Note: The ADC10 converts the digital values input to the P70 to P77 pins as analog inputs even if these ports are configured as general-purpose input ports by the port function select bits.

27.3 A/D Converter Settings

Make the following settings before starting A/D conversion.
(1) Set the analog input pins. See Section 27.2.
(2) Set the A/D conversion clock.
(3) Select the A/D conversion start and end channels.
(4) Select the A/D conversion mode.
(5) Select the A/D conversion trigger source.
(6) Set the sampling time.
(7) Select the conversion result storing mode.
(8) When using A/D converter interrupts, set interrupt conditions. See Section 27.5.

Note: Make sure the A/D converter is disabled (ADEN/ADC10_CTL register = 0) before changing the above settings. Changing the settings while the A/D converter is enabled may cause a malfunction.

27.3.1 A/D Conversion Clock Setting

To use the A/D converter, the clocks used in the A/D converter must be supplied by turning on the peripheral module clock (PCLK) output from the clock management unit (CMU) and the PCLK division clocks output from the Prescaler (PSC Ch.1). For more information on clock control, see the "Clock Management Unit (CMU)" and "Prescaler (PSC)" chapters.
The A/D conversion clock can be selected from the 15 PCLK division clocks supplied by the Prescaler. Use ADDF[3:0]/ADC10_CLK register for this selection as shown in Table 27.3.1.1.

Notes: • For the A/D conversion clock frequency range that can be used for this A/D converter, see "A/D Converter Characteristics" in the "Electrical Characteristics" chapter.

• 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 may cause the A/D converter to operate erratically.

Table 27.3.1.1 A/D Conversion Clock (PCLK Division Ratio) Selections

ADDF[3:0]	Division Ratio
0xf	Reserved
0xe	1/32768
0xd	1/16384
0xc	1/8192
0xb	1/4096
0xa	1/2048
0x9	1/1024
0x8	1/512
0x7	1/256
0x6	1/128
0x5	1/64
0x4	1/32
0x3	1/16
0x2	1/8
0x1	1/4
0x0	1/2

(Default: 0x0)

27.3.2 Selecting A/D 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 ADCS[2:0]/ADC10_TRG register and ADCE[2:0]/ADC10_TRG register, respectively.

Table 27.3.2.1 Relationship between ADCS/ADCE and Input Channels

ADCS[2:0]/ADCE[2:0]	Channel selected
0x7	AIN7
0x6	AIN6
0x5	AIN5
0x4	AIN4
0x3	AIN3
0x2	AIN2
0x1	AIN1
0x0	AIN0

(Default: 0x0)

Examples: • One A/D conversion sequence in the chip and PFBGA12U-180 package models
ADCS[2:0] = 0, ADCE[2:0] = 0
Converted only in AIN0
ADCS[2:0] = 0, ADCE[2:0] = 3
Converted in the following order: AIN0→AIN1→AIN2→AIN3
ADCS[2:0] = 4, ADCE[2:0] = 1
Converted in the following order: AIN4→AIN5→AIN6→AIN7→AIN0→AIN1

• One A/D conversion sequence in the TQFP24/QFP20-144pin and TQFP15-128pin package models
ADCS[2:0] = 0, ADCE[2:0] = 0
Converted only in AIN0
ADCS[2:0] = 0, ADCE[2:0] = 3
Converted in the following order: AIN0→(AIN1)→AIN2→(AIN3)
ADCS[2:0] = 4, ADCE[2:0] = 1
Converted in the following order: AIN4→(AIN5)→AIN6→(AIN7)→AIN0→(AIN1)

Note: The control circuits in the A/D converter supports up to eight channels regardless of the package type selected, and it performs A/D conversion if an analog input channel (AIN1/3/5/7) that does not exist in the TQFP24/QFP20-144pin and TQFP15-128pin packages is specified. In this case, the results that will be stored to ADD[15:0]/ADC10_ADD register is 0x0.

27.3.3 A/D Conversion Mode Setting

The A/D converter provides two conversion modes that can be selected using ADMS/ADC10_TRG register: one-time conversion mode and continuous conversion mode.

1. One-time conversion mode (ADMS = 0)

The A/D converter performs A/D conversion for all analog inputs within the range from the start channel specified by ADCS[2:0]/ADC10_TRG register to the end channel specified by the ADCE[2:0]/ADC10_TRG register once and then stops automatically.

2. Continuous conversion mode (ADMS = 1)

The A/D converter repeatedly performs A/D conversion for the channels in the range specified by ADCS[2:0] and ADCE[2:0] until stopped with software.

At initial reset, the A/D converter is set to one-time conversion mode.

27.3.4 Trigger Selection

Select a trigger source to start A/D conversion from among the three types listed in Table 27.3.4.1 using ADTS[1:0]/ ADC10_TRG register.

Table 27.3.4.1 Trigger Selection

ADTS[1:0]	Trigger source
0x3	External trigger (#ADTRIG)
0x2	Reserved
0x1	T8F Ch.4
0x0	Software trigger

(Default: 0x0)

1. External trigger (#ADTRIG)

The signal input to the #ADTRIG pin is used as a trigger. To use this trigger source, the I/O port pin must be configured for the #ADTRIG input using the port function select bit (see the "I/O Ports (GPIO)" chapter). An A/D conversion starts when a Low level of the #ADTRIG signal is detected.

Note: When using an external trigger to start A/D conversion, ensure to maintain the Low period of the trigger signal input to the #ADTRIG pin for two or more C33 PE Core operating clock cycles.

2. T8F Ch.4

The underflow signal of the fine mode 8-bit timer (T8F) Ch.4 is used as a trigger. Since T8F underflow cycle can be programmed with flexibility, this trigger source is effective when periodic A/D conversions are required. For more information on timer settings, see the "Fine Mode 8-bit Timers (T8F)" chapter.

3. Software trigger

Writing 1 to ADCTL/ADC10_CTL register with software serves as a trigger to start A/D conversion.

27.3.5 Sampling Time Setting

The analog signal input sampling time in this A/D converter can be configured to eight steps (two to nine conversion clock cycles) using ADST[2:0]/ADC10_TRG register.
However, do not alter ADST[2:0] from the default value (0x7: nine cycles) in the S1C33L27.

27.3.6 Setting Conversion Result Storing Mode

The A/D converter loads the 10-bit conversion results into ADD[15:0]/ADC10_ADD register (16-bit register) after an A/D conversion has completed. At this time, the 10-bit conversion results are aligned in the 16-bit register according to the conversion result storing mode set with STMD/ADC10_TRG register either as the high-order 10 bits (left justify mode) or the low-order 10 bits (right justify mode). The remaining six bits are all set to 0.

ADD bit	15	...	10	9	...	6	5	...	0
Left justify mode (STMD = 1)	(MSB)	10-bit conversion results				(LSB)	0	...	0
Right justify mode (STMD = 0)	0	...	0	(MSB)	10-bit conversion results				(LSB)

Figure 27.3.6.1 Conversion Data Alignment

27.4 A/D Conversion Control and Operations

The A/D converter should be controlled in the sequence shown below.
1. Activate the A/D converter.
2. Start A/D conversion.
3. Read the A/D conversion results.
4. Terminate A/D conversion.

27.4.1 Activating A/D Converter

After the settings described in Section 27.3 have been completed, write 1 to ADEN/ADC10_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 up the A/D converter again, or when the A/D converter is not used, ADEN must be set to 0.

27.4.2 Starting A/D conversion

The A/D converter starts A/D conversion when a trigger is input while ADEN is 1. When software trigger is selected, an A/D conversion starts by writing 1 to ADCTL/ADC10_CTL register.
The A/D converter accepts triggers from only the trigger source selected by ADTS[1:0]/ADC10_TRG register.
Once a trigger is input, the A/D converter starts sampling of the analog input signal and A/D conversion beginning with the conversion start channel selected by ADCS[2:0]/ADC10_TRG register.
The software trigger bit ADCTL functions as an A/D conversion status bit that goes 1 while A/D conversion is underway even if it has started by another trigger source. The channel in which conversion is underway can be identified by reading ADICH[2:0]/ADC10_CTL register.

27.4.3 Reading A/D conversion results

Upon completion of the A/D conversion in the start channel, the A/D converter loads the conversion results into ADD[15:0]/ADC10_ADD register and sets the conversion completion flag ADCF/ADC10_CTL register. If multiple channels are specified using ADCS[2:0]/ADC10_TRG register and ADCE[2:0]/ADC10_TRG register, the A/D converter continues A/D conversions in the subsequent channels.
The results of A/D conversion are stored in ADD[15:0] each time conversion in one channel is completed. At the same time, a conversion completion interrupt can be generated, enabling to read out the converted data. If no conversion completion interrupt is used, read the conversion results from ADD[15:0] after confirming that ADCF is set to 1 indicating completion of conversion. ADCF is reset to 0 when ADD[15:0] is read.
When a single channel or multiple channels are being converted continuously, the conversion results must be read out from ADD[15:0] before the following conversion has completed. If the A/D conversion currently underway is completed while ADCF is set to 1 (before reading the previous conversion results), ADD[15:0] is overwritten and the overwrite error flag ADOWE/ADC10_CTL register is set to 1. At this time, a conversion data overwrite error interrupt can be generated. After the conversion results are read from ADD[15:0], ADOWE should be read to check whether the read data is valid or not. Or enable conversion data overwrite error interrupts and perform error handling using the interrupt. Once ADOWE is set, it will not be reset until software writes 1. Since ADCF is also set simultaneously with ADOWE, read out the converted data to reset ADCF.

Note: Occurrence of an overwrite error does not stop continuous conversion.

27.4.4 Terminating A/D Conversion

One-time conversion mode (ADMS = 0)

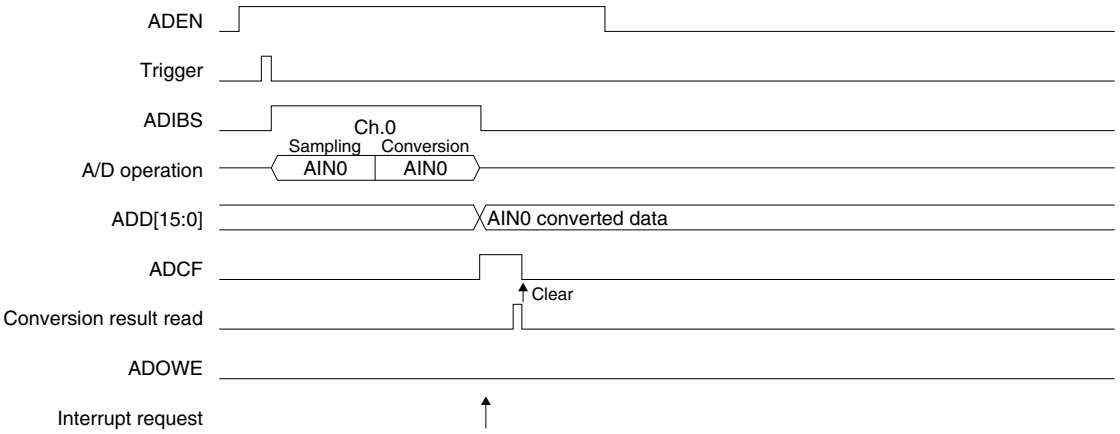
In one-time mode, the A/D converter performs A/D conversion within the channel range successively beginning with the conversion start channel specified by ADCS[2:0]/ADC10_TRG register and terminates once the conversion end channel specified by ADCE[2:0]/ADC10_TRG register has been completed. ADCTL/ADC10_CTL register is reset to 0 upon completion of the conversion sequence.

Continuous conversion mode (ADMS = 1)

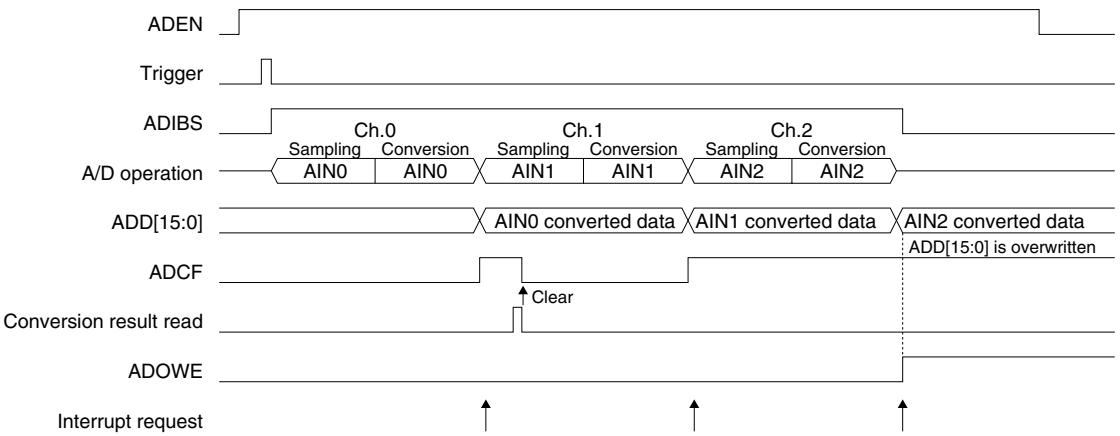
In continuous conversion mode, the A/D converter repeatedly performs A/D conversion from the conversion start channel to the conversion end channel. The hardware does not stop the conversion sequence. To stop A/D conversion, write 0 to ADCTL. Since the conversion sequence is forcibly terminated, the results of the conversion then underway cannot be obtained.

27.4.5 Timing Charts

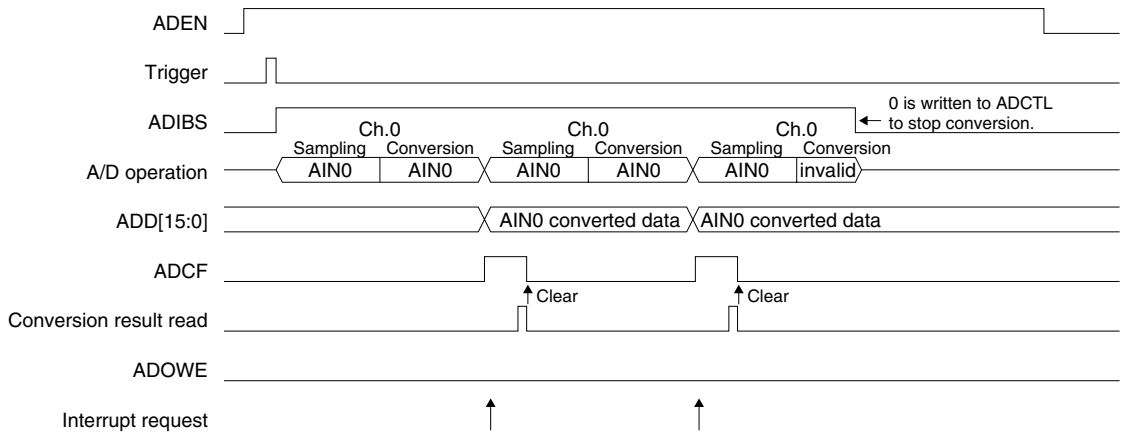
Figure 27.4.5.1 shows the operations of the A/D converter.



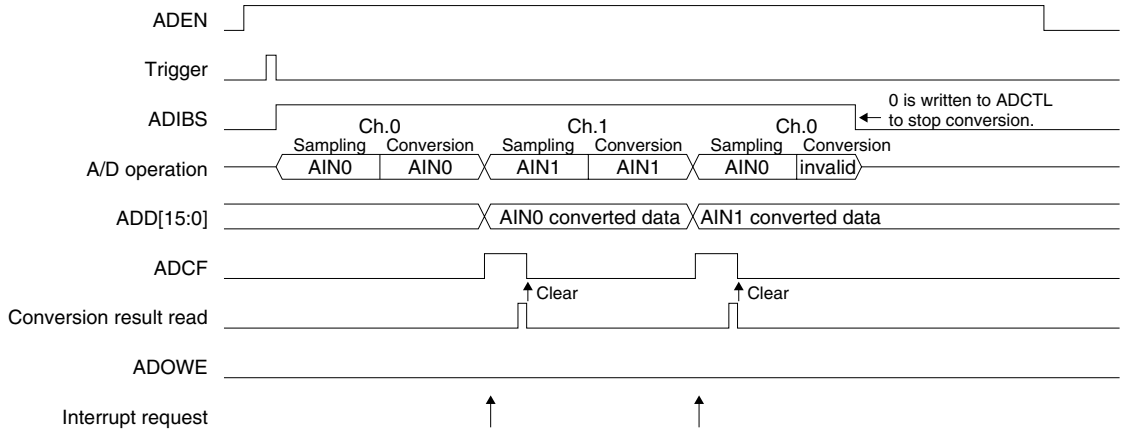
(1) Single channel (AIN0) one-time conversion mode (ADCS = 0, ADCE = 0, ADMS = 0)



(2) Multi-channel (AIN0–AIN2) one-time conversion mode (ADCS = 0, ADCE = 2, ADMS = 0)



(3) Single channel (AIN0) continuous conversion mode (ADCS = 0, ADCE = 0, ADMS = 1)



(4) Multi-channel (AIN0–AIN1) continuous conversion mode (ADCS = 0, ADCE = 1, ADMS = 1)

Figure 27.4.5.1 A/D Converter Operations

27.5 A/D Converter Interrupts and DMA

The A/D converter includes a function for generating the following two different types of interrupts.

- Conversion completion interrupt
- Conversion data overwrite error interrupt

The A/D converter outputs one interrupt signal shared by the two above interrupt causes to the interrupt controller (ITC). Inspect the status flag to determine the interrupt cause occurred.

Conversion completion interrupt

To use this interrupt, set ADCIE/ADC10_CTL register to 1. If ADCIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

When A/D conversion in a channel has completed, the A/D converter sets ADCF/ADC10_CTL register to 1, indicating that the converted data can be read out. If conversion completion interrupts are enabled (ADCIE = 1), an interrupt request is sent simultaneously to the ITC.

An interrupt occurs if other interrupt conditions are met.

You can inspect ADCF in the ADC10 interrupt handler routine to determine whether the ADC10 interrupt is attributable to a completion of conversion. If ADCF is 1, the converted data can be read out from ADD[15:0]/ ADC10_ADD register by the interrupt handler routine. The interrupt cause ADCF is reset to 0 by reading ADD[15:0] and this interrupt will not be generated until the subsequent conversion has completed.

Conversion data overwrite error interrupt

To use this interrupt, set ADOIE/ADC10_CTL register to 1. If ADOIE is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.

If the following A/D conversion has completed when ADD[15:0] has not been read (ADCF = 1), the A/D converter sets ADOWE/ADC10_CTL register to 1, indicating that ADD[15:0] is overwritten. If conversion data overwrite error interrupts are enabled (ADOIE = 1), an interrupt request is sent simultaneously to the ITC.

An interrupt occurs if other interrupt conditions are met.

You can inspect ADOWE in the ADC10 interrupt handler routine to determine whether the ADC10 interrupt is attributable to an overwrite error. If ADOWE is 1, perform error handling by the interrupt handler routine. The interrupt cause ADOWE is reset to 0 by writing 1.

For more information on interrupt processing, see the "Interrupt Controller (ITC)" chapter.

DMA Transfer

The A/D converter can invoke a DMA when A/D conversion in a channel has completed. This allows continuous data transfer via the DMAC between memory and the conversion result register (ADC10_ADD). Two DMAC channels (Ch.0 and Ch.7) are available for A/D converter. For more information on DMA transfer, see the "DMA Controller (DMAC)" chapter.

27.6 Control Register Details

Table 27.6.1 List of A/D Converter Registers

Address	Register name		Function
0x301300	ADC10_ADD	A/D Conversion Result Register	A/D converted data
0x301302	ADC10_TRG	A/D Trigger/Channel Select Register	Sets start/end channels and conversion mode.
0x301304	ADC10_CTL	A/D Control/Status Register	Controls A/D converter and indicates conversion status.
0x301306	ADC10_CLK	A/D Clock Control Register	Controls A/D converter clock.

The A/D converter registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

A/D Conversion Result Register (ADC10_ADD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Conversion Result Register (ADC10_ADD)	**0x301300** (16 bits)	D15–0	**ADD[15:0]**	A/D converted data ADD[9:0] are effective when STMD = 0 (ADD[15:10] = 0) ADD[15:6] are effective when STMD = 1 (ADD[5:0] = 0)	0x0 to 0x3ff	0x0	R	

D[15:0] ADD[15:0]: A/D Converted Data Bits

The A/D conversion results are stored. (Default: 0x0)

The data alignment in this 16-bit register (conversion result storing mode) can be selected using the STMD/ADC10_TRG register.

ADD bit	15	...	10	9	...	6	5	...	0
Left justify mode (STMD = 1)	(MSB)	10-bit conversion results				(LSB)	0	...	0
Right justify mode (STMD = 0)	0	...	0	(MSB)	10-bit conversion results				(LSB)

Figure 27.6.1 Conversion Data Alignment

This register is a read-only, so writing to this register is ignored.

A/D Trigger/Channel Select Register (ADC10_TRG)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Trigger/ Channel Select Register (ADC10_TRG)	0x301302 (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13–11	ADCE[2:0]	End channel select	0x0 to 0x7	0x0	R/W	
		D10–8	ADCS[2:0]	Start channel select	0x0 to 0x7	0x0	R/W	
		D7	STMD	Conversion result storing mode	1 ADD[15:6] 0 ADD[9:0]	0	R/W	
		D6	ADMS	Conversion mode select	1 Continuous 0 Single	0	R/W	
		D5–4	ADTS[1:0]	Conversion trigger select	ADTS[1:0] Trigger 0x3 #ADTRIG pin 0x2 reserved 0x1 T8F Ch.4 0x0 Software	0x0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	ADST[2:0]	Sampling time setting	ADST[2:0] Sampling time 0x7 9 clocks 0x6 8 clocks 0x5 7 clocks 0x4 6 clocks 0x3 5 clocks 0x2 4 clocks 0x1 3 clocks 0x0 2 clocks	0x7	R/W	

D[15:14] Reserved

D[13:11] ADCE[2:0]: End Channel Select Bits

Sets the conversion end channel with a channel number from 0 to 7. (Default: 0x0 = AIN0)
Analog inputs can be A/D-converted continuously from the channel set by ADCS[2:0] to the channel set by ADCE[2:0] in one A/D conversion. If only one channel is to be A/D converted, set the same channel number in both ADCS[2:0] and ADCE[2:0].

Table 27.6.2 Relationship between ADCS/ADCE and Input Channels

ADCS[2:0]/ADCE[2:0]	Channel selected
0x7	AIN7
0x6	AIN6
0x5	AIN5
0x4	AIN4
0x3	AIN3
0x2	AIN2
0x1	AIN1
0x0	AIN0

(Default: 0x0)

D[10:8] ADCS[2:0]: Start Channel Select Bits

Sets the conversion start channel with a channel number from 0 to 7. (Default: 0x0 = AIN0)

D7 STMD: Conversion Result Storing Mode Bit

Selects the data alignment when the conversion results are loaded into ADD[15:0].
1 (R/W): Left justify mode (10-bit conversion results → ADD[15:6], ADD[5:0] = 0)
0 (R/W): Right justify mode (10-bit conversion results → ADD[9:0], ADD[15:10] = 0) (default)

D6 ADMS: Conversion Mode Select Bit

Selects an A/D conversion mode.
1 (R/W): Continuous conversion mode
0 (R/W): One-time conversion mode (default)

Writing 1 to ADMS sets the A/D converter to continuous conversion mode. In this mode, A/D conversions in the range of the channels selected by ADCS[2:0] and ADCE[2:0] are executed continuously until stopped with software. When ADMS is 0, the A/D converter operates in one-time conversion mode. In this mode, A/D conversion is terminated after all inputs in the range of the channels selected by ADCS[2:0] and ADCE[2:0] have been converted once.

D[5:4] ADTS[1:0]: Conversion Trigger Select Bits

Selects a trigger source to start A/D conversion.

Table 27.6.3 Trigger Selection

ADTS[1:0]	Trigger source
0x3	External trigger (#ADTRIG)
0x2	Reserved
0x1	T8F Ch.4
0x0	Software trigger

(Default: 0x0)

When an external trigger is used, the #ADTRIG pin must be configured in advance using the port function select bit (see the "I/O Ports (GPIO)" chapter). A/D conversion is started when the #ADTRIG signal goes Low. When T8F Ch.4 is used, since its underflow signal serves as a trigger, set the underflow cycle and other conditions for the timer.

D3 Reserved

D[2:0] ADST[2:0]: Sampling Time Setting Bits

Sets the analog input sampling time.

Table 27.6.4 Sampling Time Settings

ADST[2:0]	Sampling time (in conversion clock cycles)
0x7	9 cycles
0x6	8 cycles
0x5	7 cycles
0x4	6 cycles
0x3	5 cycles
0x2	4 cycles
0x1	3 cycles
0x0	2 cycles

(Default: 0x7)

Note: Leave the ADST[2:0] default setting (nine cycles) unchanged to ensure conversion accuracy.

A/D Control/Status Register (ADC10_CTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Control/ Status Register (ADC10_CTL)	0x301304 (16 bits)	D15	–	reserved	–	–	–	0 when being read.
		D14–12	ADICH[2:0]	Conversion channel indicator	0x0 to 0x7	0x0	R	
		D11	–	reserved	–	–	–	0 when being read.
		D10	ADIBS	ADC10 status	1 Busy 0 Idle	0	R	
		D9	ADOWE	Overwrite error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D8	ADCF	Conversion completion flag	1 Completed 0 Run/Stand-by	0	R	Reset when ADC10_ADD is read.
		D7–6	–	reserved	–	–	–	0 when being read.
		D5	ADOIE	Overwrite error interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	ADCIE	Conversion completion int. enable	1 Enable 0 Disable	0	R/W	
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	ADCTL	A/D conversion control	1 Start 0 Stop	0	R/W	
		D0	ADEN	ADC10 enable	1 Enable 0 Disable	0	R/W	

D15 Reserved

D[14:12] ADICH[2:0]: Conversion Channel Indicator 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.

D11 Reserved

D10 ADIBS: ADC10 Status Bit

Indicates the A/D converter status.

1 (R): Being converted
0 (R): Conversion completed/standby (default)

ADIBS is set to 1 at the input trigger signal edge (at the beginning of sampling) and is reset to 0 upon completion of conversion (when ADCTL is set to 0).

D9 ADOWE: Overwrite Error Flag Bit

Indicates that the converted results in ADD[15:0]/ADC10_ADD register have been overwritten before reading.

1 (R): Overwrite error (cause of interrupt has occurred)
0 (R): Normal (cause of interrupt has not occurred) (default)
1 (W): Flag is reset
0 (W): Ignored

When a single channel or multiple channels are being converted continuously, ADD[15:0] is overwritten and ADOWE is set to 1 if the A/D conversion currently underway is completed while ADCF is set to 1 (before reading the previous conversion results). After the conversion results are read from ADD[15:0], ADOWE should be read to check whether the read data is valid or not.
ADOWE is a cause of ADC10 interrupt. When ADOWE is set to 1, a conversion data overwrite error interrupt request is output to the ITC if ADOIE has been set to 1 (interrupt enabled). An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied. ADOWE is reset by writing 1.

D8 ADCF: Conversion Completion Flag Bit

Indicates that A/D conversion has been completed.

1 (R): Conversion completed (cause of interrupt has occurred)
0 (R): Being converted/standby (cause of interrupt has not occurred) (default)

ADCF is set to 1 when A/D conversion is completed, and the converted data is loaded into ADD[15:0]/ADC10_ADD register.
ADCF is a cause of ADC10 interrupt. When ADCF is set to 1, a conversion completion interrupt request is output to the ITC if ADCIE has been set to 1 (interrupt enabled). An interrupt is generated if the ITC and C33 PE Core interrupt conditions are satisfied. ADCF is reset to 0 by reading ADD[15:0].
An overwrite error occurs if the next A/D conversion is completed while ADCF is set (see ADOWE above), ADCF must be reset by reading ADD[15:0] before an overwrite occurs. When an overwrite error occurs, ADCF is also set due to completion of conversion.

D[7:6] Reserved

D5 ADOIE: Overwrite Error Interrupt Enable Bit

Enables or disables interrupts caused by occurrences of conversion data overwrite errors.

1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting ADOIE to 1 enables conversion data overwrite error interrupt requests to the ITC; setting to 0 disables interrupts.

D4 ADCIE: Conversion Completion Interrupt Enable Bit

Enables or disables interrupts caused by completion of conversion.

1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

Setting ADCIE to 1 enables conversion data overwrite error interrupt requests to the ITC; setting to 0 disables interrupts.

D[3:2] Reserved

D1 ADCTL: A/D Conversion Control Bit

Controls A/D conversion.

1 (W): Software trigger
0 (W): Stop A/D conversion
1 (R): Being converted
0 (R): Conversion completed/standby (default)

Write 1 to ADCTL to start A/D conversion by a software trigger. If any other trigger is used, ADCTL is automatically set to 1 by the hardware.

ADCTL remains set while A/D conversion is underway. In one-time conversion mode, upon completion of A/D conversion in the specified channels, ADCTL is reset to 0 and the A/D conversion circuit stops operating. To stop A/D conversion during operation in continuous conversion mode, reset ADCTL by writing 0.
When ADEN is 0 (A/D conversion disabled), ADCTL is fixed to 0, with no trigger accepted.

D0 ADEN: ADC10 Enable Bit

Enables or disables the A/D converter operations.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Writing 1 to ADEN enables the A/D converter, meaning it is ready to start A/D conversion (i.e., ready to accept a trigger).
When ADEN is 0, the A/D converter is disabled, meaning it is unable to accept a trigger.
Before setting the modes, start/end channels, or other A/D converter conditions, be sure to reset ADEN to 0. This helps to prevent the A/D converter from operating erratically.

A/D Clock Control Register (ADC10_CLK)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
A/D Clock Control Register (ADC10_CLK)	0x301306 (16 bits)	D15–4	–	reserved	–		–	–	0 when being read.
		D3–0	ADDF[3:0]	A/D converter clock division ratio select	ADDF[3:0]	Division ratio	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
					0xf	reserved			
					0xe	1/32768			
					0xd	1/16384			
					0xc	1/8192			
					0xb	1/4096			
					0xa	1/2048			
					0x9	1/1024			
					0x8	1/512			
					0x7	1/256			
					0x6	1/128			
					0x5	1/64			
					0x4	1/32			
					0x3	1/16			
					0x2	1/8			
					0x1	1/4			
					0x0	1/2			

D[15:4] Reserved

D[3:0] ADDF[3:0]: A/D Converter Clock Division Ratio Select Bits

Selects the A/D converter clock (PCLK division ratio).

Table 27.6.5 A/D Conversion Clock (PCLK Division Ratio) Selections

ADDF[3:0]	Division Ratio
0xf	Reserved
0xe	1/32768
0xd	1/16384
0xc	1/8192
0xb	1/4096
0xa	1/2048
0x9	1/1024
0x8	1/512
0x7	1/256
0x6	1/128
0x5	1/64
0x4	1/32
0x3	1/16
0x2	1/8
0x1	1/4
0x0	1/2

(Default: 0x0)

Note: The A/D converter uses the prescaler output as the source clock, the prescaler must be run in advance.

28 LCD Controller (LCDC)

28.1 LCDC Module Overview

The S1C33L27 has a built-in LCD controller (LCDC) that supports 4/8-bit monochrome and color LCD panels, and 12/16/24-bit TFT panels. Also the S1C33L27 contains a 32K-byte VRAM (IVRAM) allowing a 320 × 240-dot 4-color/gray scale screen (2-bpp mode) to be displayed. Furthermore, the bus controller (SRAMC, SDRAMC) allows the LCDC to access the external SDRAM/SRAM as a VRAM, thus a 320 × 240-dot screen with 24-bpp color depth (16M colors) can be displayed. The LCDC provides support for Picture-in-Picture Plus (a variable size overlay window). The LCDC can use both the IVRAM and external VRAM, this makes it possible to manages the main and sub (PIP) window display data in different memories.
The features of the LCDC are described below.

Internal bus interface and VRAM

- The UMA (Unified Memory Architecture) method using the Bus Arbiter and IVRAM interface is implemented. This method allows the LCDC to access SDRAM/SRAM (external VRAM) while the CPU is accessing an internal circuit, or to access IVRAM (internal VRAM) while the CPU is accessing another circuit.
- The 32K-byte internal VRAM (IVRAM) is mapped at addresses 0xc0000 to 0xc7fff.
- The external VRAM map is configurable.
- The frame interrupt signal is output to the ITC.

Display support

- 4- or 8-bit monochrome LCD interface
- 4- or 8-bit color LCD interface
- Single-panel, single-drive passive displays
- 12-, 16- or 24-bit TFT interface
- Typical resolutions
  320 × 240 (24-bpp mode, external VRAM is required) bpp = bits per pixel
  320 × 240 (2-bpp mode)
  * Note that the panel width must be a multiple of (16 ÷ bpp).

Display modes

- Due to frame rate modulation, gray scale display is possible in up to 16 shades of gray when a monochrome passive LCD panel is used.
  Two-shade display in 1-bpp mode
  Four-shade display in 2-bpp mode
  16-shade display in 4-bpp mode
- A maximum of 4K colors can be simultaneously displayed on a color passive LCD panel.
  Two-color display in 1-bpp mode
  Four-color display in 2-bpp mode
  16-color display in 4-bpp mode
  256-color display in 8-bpp mode
  4K-color display in 12- and 16-bpp mode
- A maximum of 16M colors can be simultaneously displayed on a TFT panel.
  Two-color display in 1-bpp mode
  Four-color display in 2-bpp mode
  16-color display in 4-bpp mode
  256-color display in 8-bpp mode
  4K-color display in 12-bpp mode
  64K-color display in 16-bpp mode
  16M-color display in 24-bpp mode

Display features

- Picture-in-Picture Plus to display a variable size window overlaid over background image.
- Virtual display function to handle images with a different resolution from the LCD panel (any area in the virtual screen can be displayed on the LCD.)

Clock

- LCDC_CLK for the LCD controller is generated in the CMU by dividing the OSC3 clock by 1 to 32.
- Different clock paths are provided for the AHB bus interface (for accessing the VRAM), SAPB interface (for accessing the control registers), and LCDC_CLK, and each clock supply can be controlled individually in the CMU. This makes it possible to reduce current consumption by disabling unnecessary clocks.

Power save

- Software power-save mode
- Blank display

28.2 Block Diagram

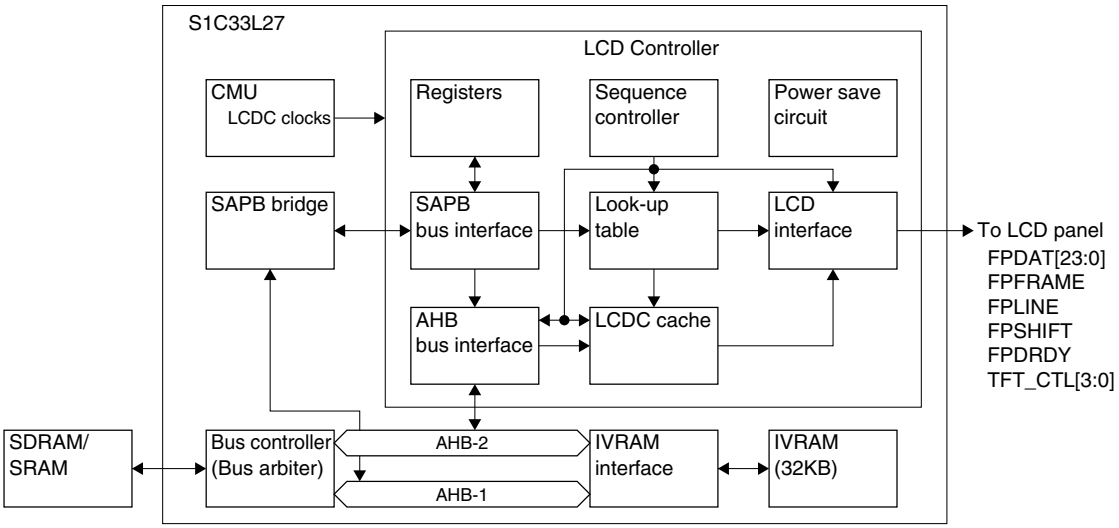


Figure 28.2.1 LCD Controller Block Diagram

SAPB bus interface

The C33 PE Core accesses the LCDC registers and look-up table through this interface.

AHB bus interface

The LCDC access the VRAM through this interface.

LCDC cache

This consists of two 32-byte FIFOs used as a display data cache for sending display data to the LCD panel.

Sequence controller

The sequence controller controls data flow from the AHB bus interface to the LCD interface through the look-up table. It also generates display data memory addresses for refreshing display.

Look-up table

In monochrome mode, the look-up table included in the LCDC is used.
The color mode does not support look-up table.

LCD interface

The LCD interface performs frame rate modulation for passive LCD panels. It also formats display data and generates the timing control signals for various LCD panels.

Power save circuit

This circuit controls the power save mode in the LCDC.

28.3 LCDC Output Pins

Table 28.3.1 lists the output pins of the LCD controller. Table 28.3.2 shows the pin configurations classified by type of LCD panel.

Table 28.3.1 LCDC Output Pins

Pin name	I/O	Qty	Function
FPDAT[23:0]	O	24	LCD display data outputs
FPFRAME	O	1	LCD frame clock output
FPLINE	O	1	LCD line clock output
FPSHIFT	O	1	LCD shift clock output
FPDRDY	O	1	LCD DRDY/MOD signal output
TFT_CTL[3:0]	O	4	TFT interface control signal outputs

The LCDC output pins are shared with I/O ports and are initially set as general purpose I/O port pins. The pin functions must be switched using the port function select bits to use the general purpose I/O port pins as LCDC output pins. For detailed information on pin function switching, see the "I/O Ports (GPIO)" chapter.

Table 28.3.2 Pin Configurations by LCD Panel Type

Pin name	Monochrome passive panel		Color passive panel			TFT panel
	4-bit data width	8-bit data width	4-bit data width	8-bit data width format 1	8-bit data width format 2	– *2
FPFRAME	FPFRAME					SPS
FPLINE	FPLINE					LP
FPSHIFT	FPSHIFT					DCLK
FPDRDY	MOD			FPSHIFT2	MOD	DEN
FPDAT0	– *1	D0	– *1	D0	D0	B3
FPDAT1	– *1	D1	– *1	D1	D1	B4
FPDAT2	– *1	D2	– *1	D2	D2	B5
FPDAT3	– *1	D3	– *1	D3	D3	B6
FPDAT4	D0	D4	D0	D4	D4	B7
FPDAT5	D1	D5	D1	D5	D5	G2
FPDAT6	D2	D6	D2	D6	D6	G3
FPDAT7	D3	D7	D3	D7	D7	G4
FPDAT8	– *1					G5
FPDAT9	– *1					G6
FPDAT10	– *1					G7
FPDAT11	– *1					R3
FPDAT12	– *1					R4
FPDAT13	– *1					R5
FPDAT14	– *1					R6
FPDAT15	– *1					R7
FPDAT16	– *1					B0
FPDAT17	– *1					B1
FPDAT18	– *1					B2
FPDAT19	– *1					G0
FPDAT20	– *1					G1
FPDAT21	– *1					R0
FPDAT22	– *1					R1
FPDAT23	– *1					R2
TFT_CTL0	– *1					PS
TFT_CTL1	– *1					CLS

Pin name	Monochrome passive panel		Color passive panel			TFT panel
	4-bit data width	8-bit data width	4-bit data width	8-bit data width format 1	8-bit data width format 2	– *2
TFT_CTL2	– *1					REV
TFT_CTL3	– *1					SPL

*1 These pins can be used for other peripheral functions.

*2 Since the LCDC supports maximum 24-bit data width TFT panels, no LCD panel data width configuration is required.
When using a 12-bit data width TFT panel, connect it to the R7–R4, G7–G4, and B7–B4 pins.
When using a 16-bit data width TFT panel, connect it to the R7–R3, G7–G2, and B7–B3 pins.
When using a 18-bit data width TFT panel, connect it to the R7–R2, G7–G2, and B7–B2 pins.

Table 28.3.3 LCD Panel Configurations

LCD Panel	PANELSEL *	COLOR *	DWD[1:0] *	BPP[2:0] *
Monochrome passive panel	0 (STN)	0 (Mono)	0 (4-bit data width) 1 (8-bit data width) 3 (8-bit data width)	0 (1 bpp) 1 (2 bpp) 2 (4 bpp)
Color passive panel	0 (STN)	1 (Color)	0 (4-bit data width) 1 (8-bit data width format 1) 3 (8-bit data width format 2)	0 (1 bpp) 1 (2 bpp) 2 (4 bpp) 3 (8 bpp) 4 (12 bpp) 5 (16 bpp)
TFT panel	1 (TFT)	1 (Color)	–	0 (1 bpp) 1 (2 bpp) 2 (4 bpp) 3 (8 bpp) 4 (12 bpp) 5 (16 bpp) 6 (24 bpp)

* The PANELSEL, COLOR, DWD[1:0], and BPP[2:0] control bits are assigned in the LCDC_DISPMOD register.

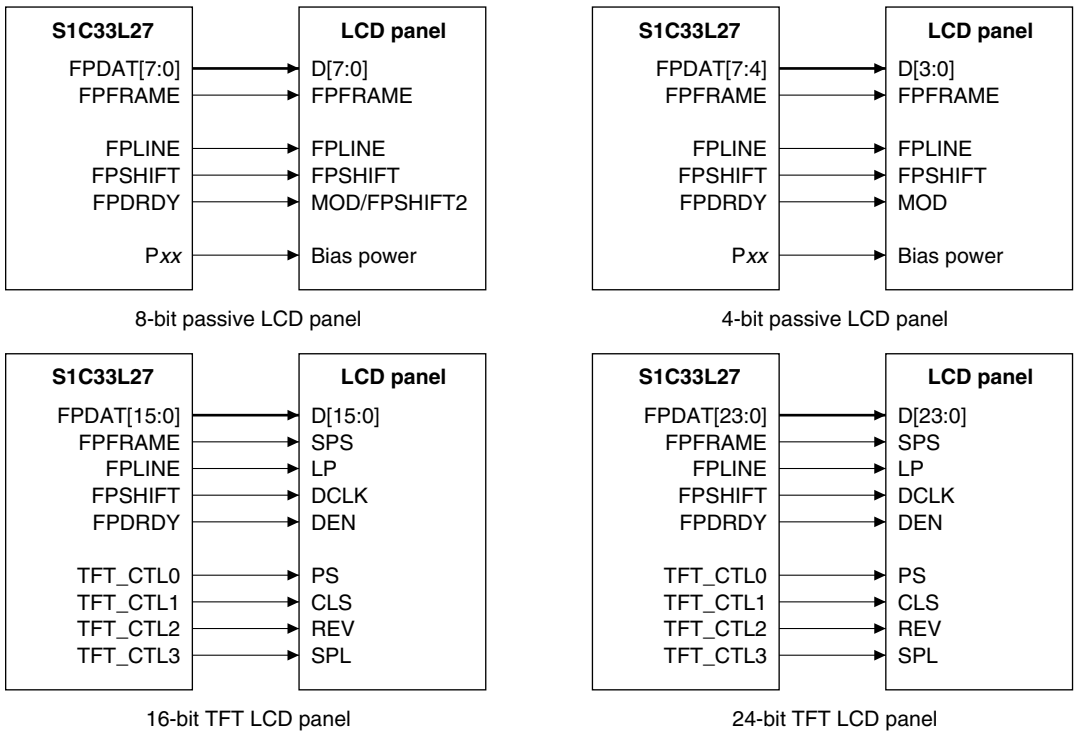


Figure 28.3.1 Typical LCD-Panel Connections

28.4 System Settings

28.4.1 Configuration of Display Data Memory (VRAM)

The S1C33L27 has a built-in 32K-byte display data memory (IVRAM). This memory allows the software to configure as a VRAM by locating at 0xc0000 to 0xc7fff in Area 3, a general-purpose RAM by locating in Area 0 or a calculator RAM. Setting RAMCFG[2:0]/MISC_RAMCFG register to 0x0 or 0x1 configures the IVRAM as a VRAM in Area 3. At initial reset, this memory is located in Area 3 as a VRAM allowing LCDC to access directly. For more information on the RAM configuration, see the "Misc Register (MISC)" chapter.

The LCDC can use an external SDRAM/SRAM as a VRAM in addition to IVRAM (the external SDRAM/SRAM can store general-purpose data as well as display data). There is no special configuration procedure for use of the external VRAM. Furthermore, both the external SDRAM/SRAM and IVRAM can be used as a VRAM simultaneously. The LCDC handles two screen data for the main window and the sub-window overlaid over the main window to support "Picture-in-Picture Plus." Also the LCDC can switch the display by selecting a screen from two or more display data prepared in the VRAM. Since the display start memory address is specified using a register, display data can be stored in any location (but it must be a word boundary address) in the memory.
Figure 28.4.1.1 shows memory usage examples.

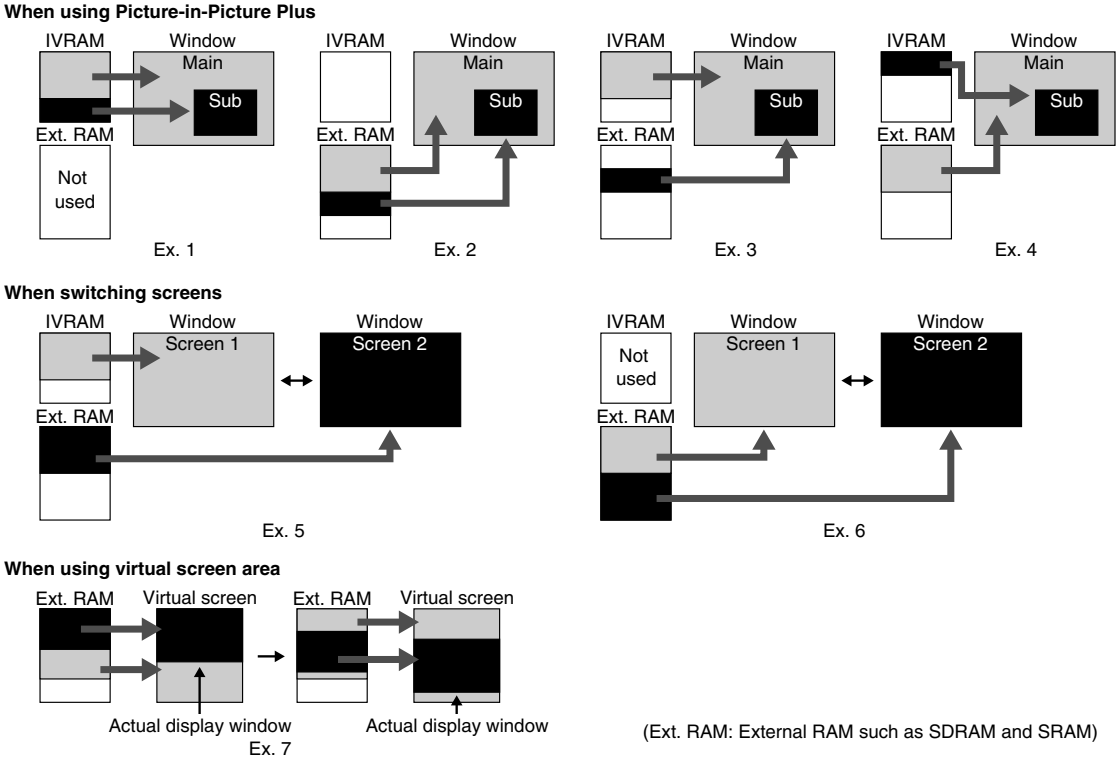


Figure 28.4.1.1 Memory Usage Examples

The memory size required for a screen depends on the virtual screen size and bpp mode. It can be expressed by the following equation:

Virtual screen size = H_PIXEL × V_PIXEL × bpp / 32 [words]
(The fractional portion of the number must be rounded up.)
H_PIXEL: Number of horizontal pixels
V_PIXEL: Number of vertical pixels
bpp: Number of bits per pixel (1, 2, 4, 8, 12, 16, 24)

For example, a 320 × 240-pixel screen displayed in 64K colors (16-bpp mode) needs a 38,400 words (150K bytes) of memory area.

28.4.2 Setting the LCDC Clock

The LCDC operates with the LCDC_CLK, BCLK, LCDC_AHB_CLK (PCLK), and LCDC_SAPB_CLK (PCLK) clocks supplied from the CMU. For controlling the clocks, see the "Clock Management Unit (CMU)" chapter.

LCDC_CLK

This is the LCDC operating clock (pixel clock) generated by dividing the OSC3 clock. The frequency divider generates 32 kinds of clocks from OSC3/1 to OSC3/32. Select a division ratio according to the frame rate using LCLKDIV[4:0]/CMU_LCLKDIV register.

$$\text{Frame rate} = \frac{f_{LCDC\_CLK}}{HT \times VT} [\text{Hz}]$$

f_{LCDC_CLK}: LCDC_CLK frequency
HT: Horizontal total period (horizontal panel size + horizontal non-display period) [pixels]
VT: Vertical total period (vertical panel size + vertical non-display period) [lines]

Table 28.4.2.1 LCDC Clock (OSC3 Division Ratio) Selections

LCLKDIV[4:0]	Division ratio (OSC3/n)	LCLKDIV[4:0]	Division ratio (OSC3/n)
0x1f	1/32	0xf	1/16
0x1e	1/31	0xe	1/15
0x1d	1/30	0xd	1/14
0x1c	1/29	0xc	1/13
0x1b	1/28	0xb	1/12
0x1a	1/27	0xa	1/11
0x19	1/26	0x9	1/10
0x18	1/25	0x8	1/9
0x17	1/24	0x7	1/8
0x16	1/23	0x6	1/7
0x15	1/22	0x5	1/6
0x14	1/21	0x4	1/5
0x13	1/20	0x3	1/4
0x12	1/19	0x2	1/3
0x11	1/18	0x1	1/2
0x10	1/17	0x0	1/1

(Default: 0x7)

LCDC_CKE/CMU_CLKCTL2 register is used for clock supply control (default: off). Before using the LCDC, set LCDC_CKE to 1.

Note: Disable LCDC_CLK supply (LCDC_CKE = 0) when changing the clock division ratio using LCLKDIV[4:0] or before executing the slp instruction.

BCLK and LCDC_AHB_CLK (PCLK)

These clocks are required for the LCDC to access the VRAM.
By default, BCLK is supplied from the CMU. It can be stopped in HALT mode using BCLK_CKE/CMU_CLKCTL2 register.
LCDC_AHB_CLK is stopped by default. Before using the LCDC, set LCDC_AHB_CKE/CMU_CLKCTL2 register to 1 to supply the clock from the CMU.

LCDC_SAPB_CLK (PCLK)

This clock is required for accessing the LCDC registers. LCDC_SAPB_CLK is stopped by default. Before accessing the LCDC registers, set LCDC_SAPB_CKE/CMU_CLKCTL2 register to 1 to supply the clock from the CMU.

28.5 Setting the LCD Panel

28.5.1 Types of Panels

The LCD controller supports the following types of LCD panels.

- 4- or 8-bit single monochrome passive LCD panels
- 4- or 8-bit single color passive LCD panels
- 12-, 16- or 24-bit TFT LCD panels

Dual panels are not supported.
The type of LCD panel used must be set in the LCD controller in advance, using the control bits described below.

Selecting between STN and TFT

Use PANELSEL/LCDC_DISPMOD register to select the type of LCD panel, either STN or TFT.
PANELSEL = 1: TFT panel selected
PANELSEL = 0: STN panel selected (default)

Selecting between color and monochrome

Use COLOR/LCDC_DISPMOD register to select the type of LCD panel, either color or monochrome.
COLOR = 1: Color panel selected
COLOR = 0: Monochrome panel selected (default)

Selecting the STN panel data width

Use DWD[1:0]/LCDC_DISPMOD register to select the data width and format for STN panels.

Table 28.5.1.1 Data Width Selection of STN Panels

COLOR	DWD[1:0]	LCD panel
1	0x3	Color single 8-bit passive LCD panel (format 2)
	0x2	Reserved
	0x1	Color single 8-bit passive LCD panel (format 1)
	0x0	Color single 4-bit passive LCD panel
0	0x3	Monochrome single 8-bit passive LCD panel
	0x2	Reserved
	0x1	Monochrome single 8-bit passive LCD panel
	0x0	Monochrome single 4-bit passive LCD panel

The data width for a TFT LCD panel is determined according to the display (bpp) mode selected.

28.5.2 STN Panel Timing Parameters

The STN panel timing parameters shown in Figures below can be set using the LCDC control registers.

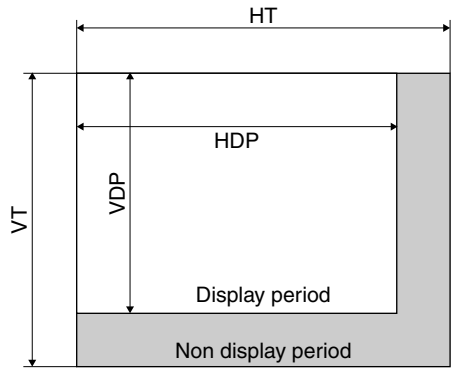
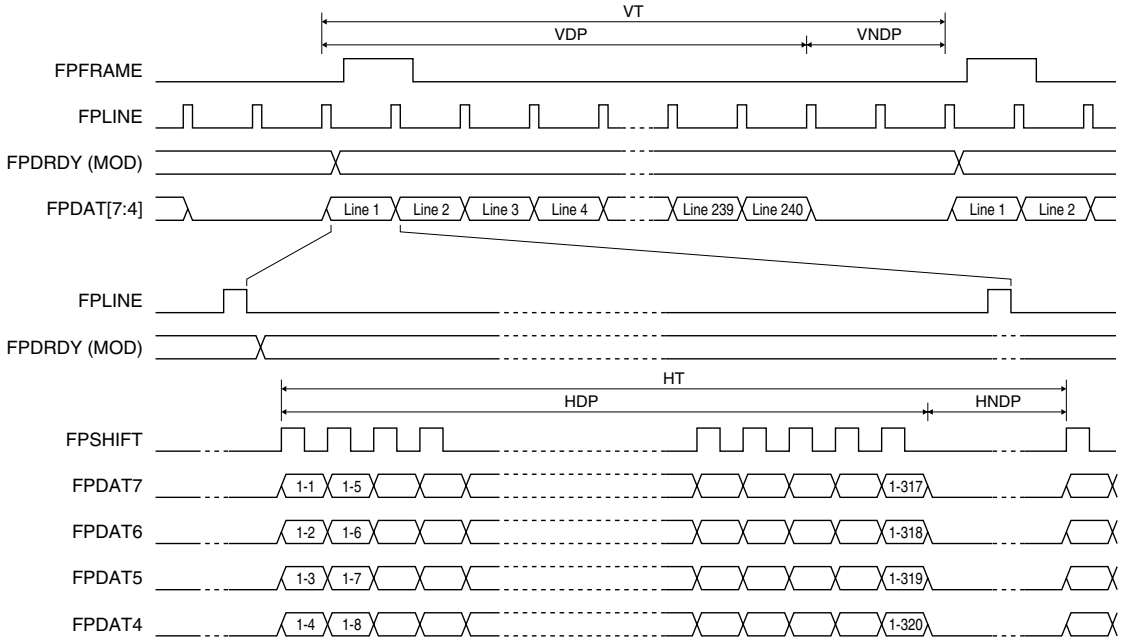
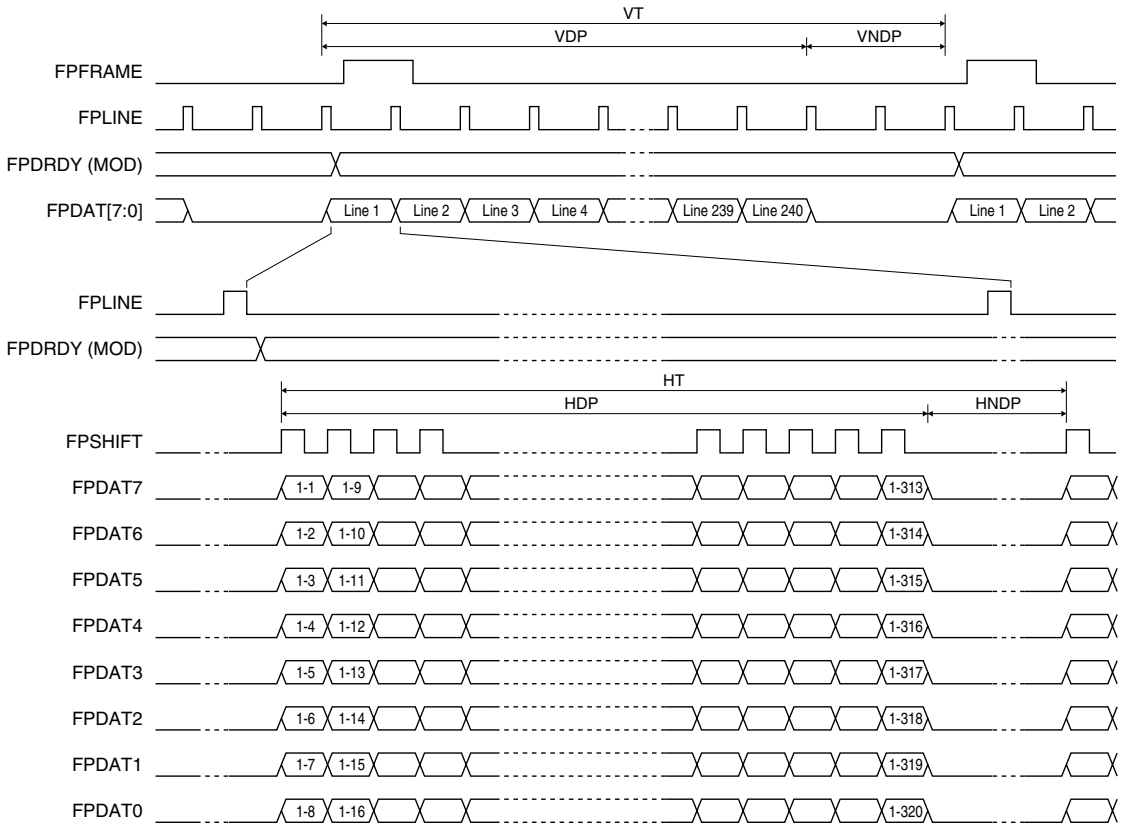


Figure 28.5.2.1 STN Panel Timing Parameters



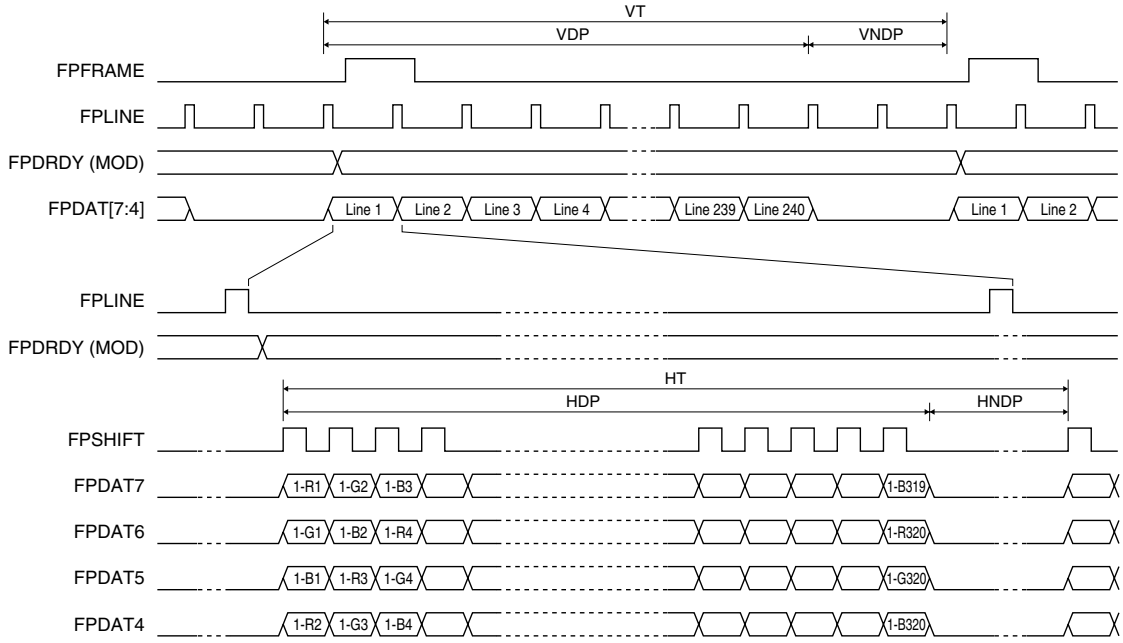
* Diagram drawn with 2 FPLINE vertical blank period
  Example timing for a 320 × 240 panel
  For this timing diagram FPSHIFT_MSK is set to 1

Figure 28.5.2.2 4-bit Single Monochrome Panel Timing Chart (Example)



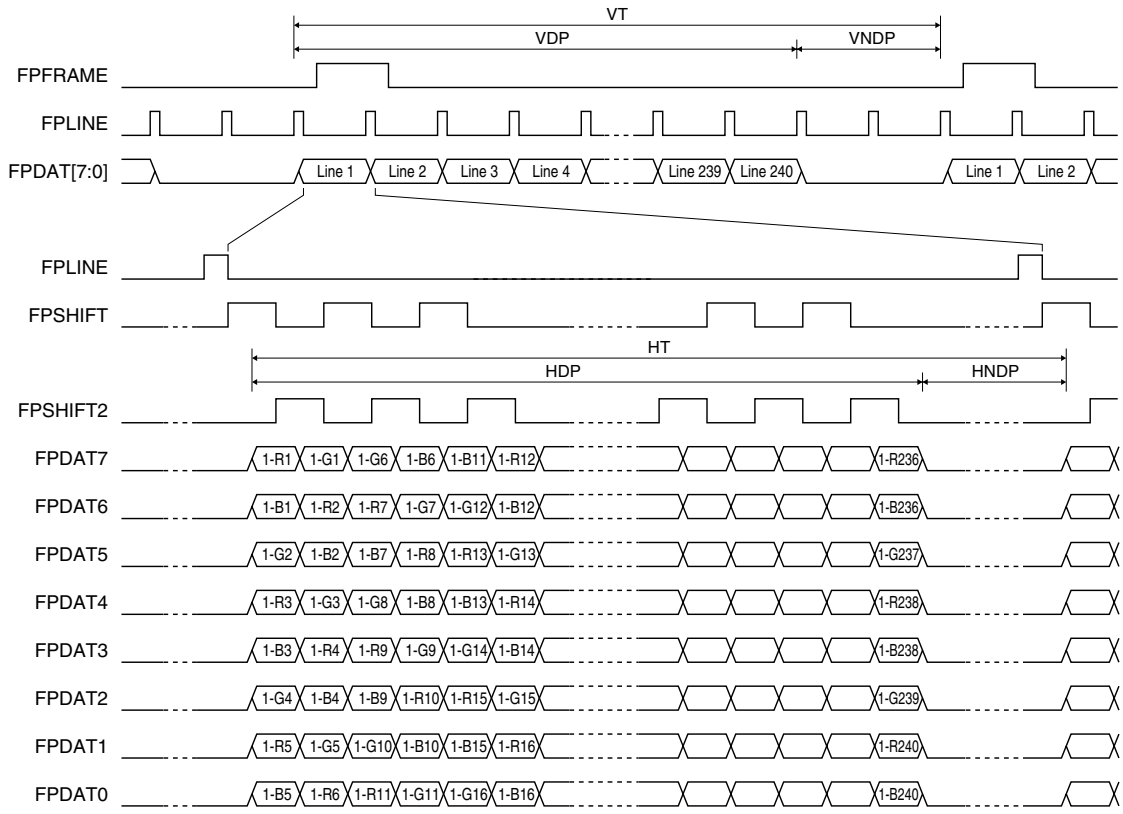
* Diagram drawn with 2 FPLINE vertical blank period
  Example timing for a 320 × 240 panel
  For this timing diagram FPSHIFT_MSK is set to 1

Figure 28.5.2.3 8-bit Single Monochrome Panel Timing Chart (Example)



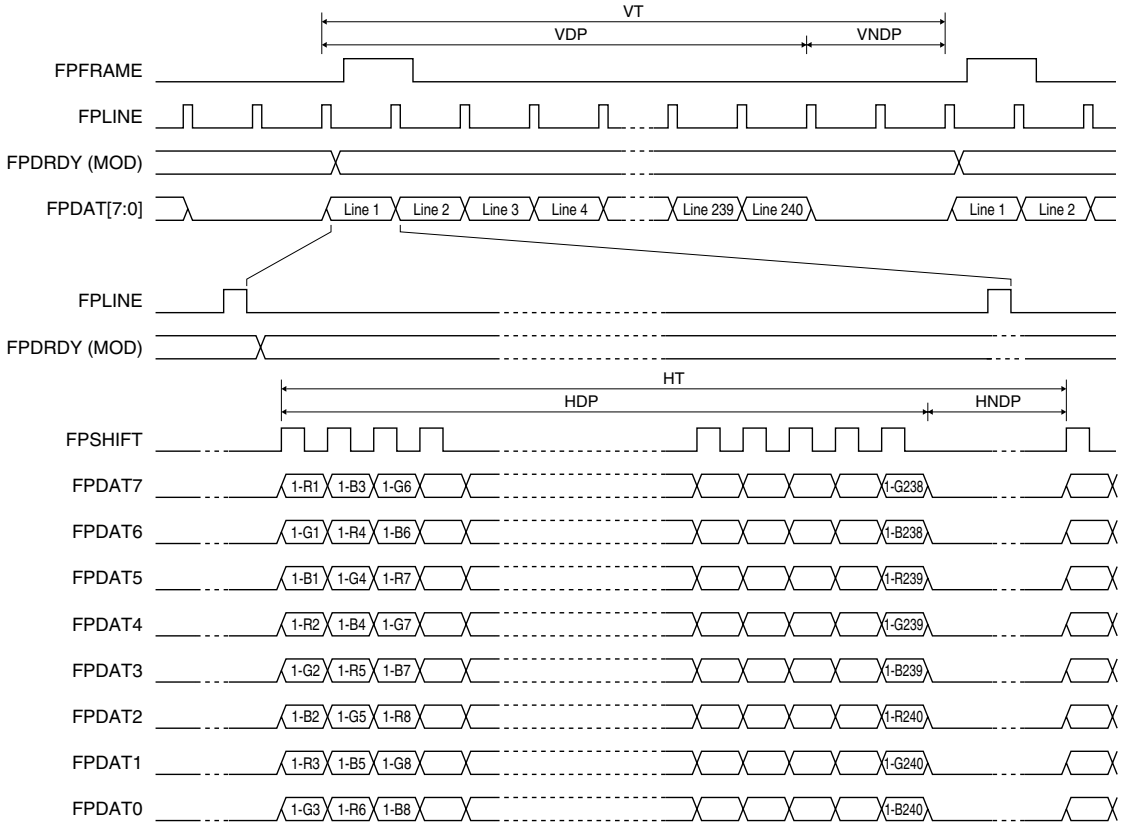
* Diagram drawn with 2 FPLINE vertical blank period
  Example timing for a 320 × 240 panel

Figure 28.5.2.4 4-bit Single Color Panel Timing Chart (Example)



* Diagram drawn with 2 FPLINE vertical blank period
  Example timing for a 320 × 240 panel

Figure 28.5.2.5 8-bit Single Color Panel (Format 1) Timing Chart (Example)



∗ Diagram drawn with 2 FPLINE vertical blank period
Example timing for a 320 × 240 panel

Figure 28.5.2.6 8-bit Single Color Panel (Format 2) Timing Chart (Example)

HT: Horizontal total period

Use HTCNT[6:0]/LCDC_HDISP register to set the horizontal total period.

$HT = (HTCNT[6:0] + 1) \times 8$ [Ts] Ts: Pixel clock (LCDC_CLK) period

HTCNT[6:0] must be programmed such that the following condition is met:
$HTCNT[6:0] \geq HDPCNT[6:0] + 3$

Note: HT should be determined so that the horizontal non-display period (HNDP = HT - HDP) will be longer than the time required when the LCDC accesses eight words in the VRAM.

HDP: Horizontal display period

Use HDPCNT[6:0]/LCDC_HDISP register to set the horizontal display period (= horizontal panel resolution).

$HDP = (HDPCNT[6:0] + 1) \times 8$ [Ts]

HDPCNT[6:0] must be programmed such that the following condition is met:
$HDP \geq 16$ $(HDPCNT[6:0] \geq 1)$

VT: Vertical total period

Use VTCNT[9:0]/LCDC_VDISP register to set the vertical total period.

$VT = VTCNT[9:0] + 1$ [lines]

VDP: Vertical display period

Use VDPCNT[9:0]/LCDC_VDISP register to set the vertical display period (= vertical panel resolution).

$VDP = VDPCNT[9:0] + 1$ [lines]

VDPCNT[9:0] must be programmed such that the following condition is met:
$VT \geq VDP + 1$

FPSHIFT mask for monochrome LCD panel

When color passive panel is selected (COLOR/LCDC_DISPMOD register = 1), the FPSHIFT clock is output during the horizontal display period (HDP) and it stops during the horizontal non-display period (HNDP) as shown in Figures 28.5.2.2 to 28.5.2.6.

When monochrome passive panel is selected (COLOR = 0), the FPSHIFT clock does not stop even in the horizontal non-display period by the default setting. To stop the FPSHIFT clock during the horizontal non-display period, set FPSHIFT_MSK/LCDC_DISPMOD register to 1.

Note: When using an STN panel, the registers for setting the TFT timing parameters must be set to 0x0.

28.5.3 TFT Panel Timing Parameters

The TFT panel timing parameters shown in Figures below can be set using the LCDC control registers.

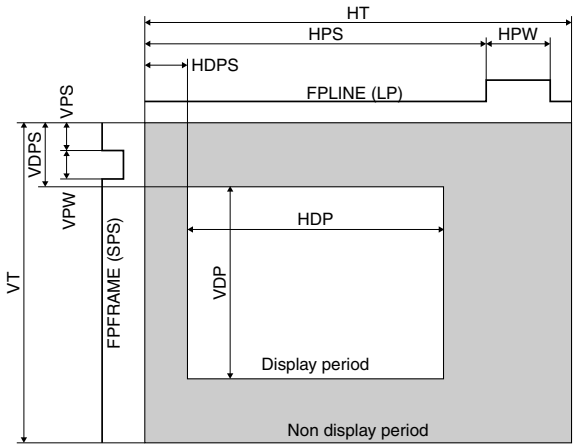
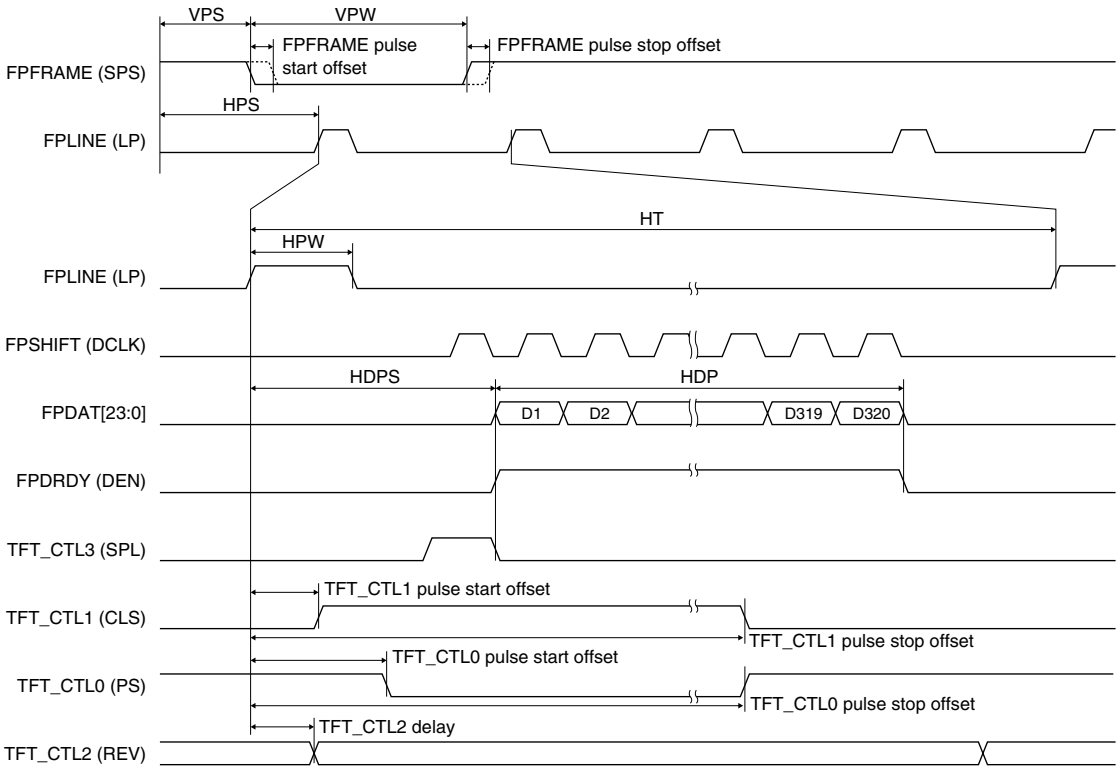
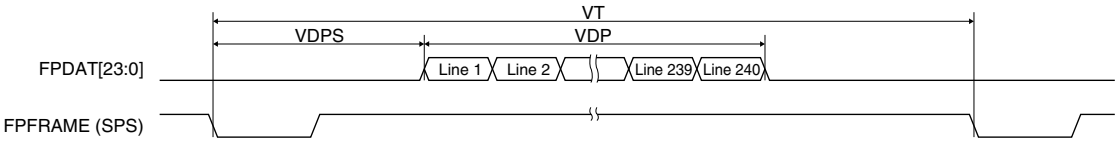


Figure 28.5.3.1 TFT Panel Timing Parameters



Horizontal timing



Vertical timing

Figure 28.5.3.2 TFT Panel Timing Chart

HT: Horizontal total period

Use HTCNT[6:0]/LCDC_HDISP register to set the horizontal total period.

HT = (HTCNT[6:0] + 1) × 8 [Ts] (Ts: Pixel clock period)

HTCNT[6:0] must be programmed such that the following conditions are met:

HTCNT[6:0] ≥ HDPCNT[6:0] + 3

HT > HDP + HDPS

Note: HT should be determined so that the horizontal non-display period (HNDP = HT - HDP) will be longer than the time required when the LCDC accesses eight words in the VRAM.

HDP: Horizontal display period

Use HDPCNT[6:0]/LCDC_HDISP register to set the horizontal display period (= horizontal panel resolution).

HDP = (HDPCNT[6:0] + 1) × 8 [Ts]

HDPCNT[6:0] must be programmed such that the following condition is met:

HDP ≥ 16 (HDPCNT[6:0] ≥ 1)

HDPS: Horizontal display period start position

Use HDPSCNT[9:0]/LCDC_HDPS register to set the horizontal display period start position for the TFT panel.

HDPS = HDPCNT[9:0] + 1 [Ts]

HDPSCNT[9:0] must be programmed such that the following condition is met:

HT > HDP + HDPS

HPS: Horizontal sync pulse start position

Use FPLINE_ST[9:0]/LCDC_FPLINE register to set the horizontal sync pulse (FPLINE or LP) start position for the TFT panel.

HPS = FPLINE_ST[9:0] + 1 [Ts]

HPW: Horizontal sync pulse width

Use FPLINE_WD[6:0]/LCDC_FPLINE register to set the horizontal sync pulse width for the TFT panel.

HPW = FPLINE_WD[6:0] + 1 [Ts]

Horizontal sync pulse polarity

Use FPLINE_POL/LCDC_FPLINE register to set the horizontal sync pulse polarity for the TFT panel.

FPLINE_POL = 1: Active high

FPLINE_POL = 0: Active low (default)

VT: Vertical total period

Use VTCNT[9:0]/LCDC_VDISP register to set the vertical total period.

VT = VTCNT[9:0] + 1 [lines]

VTCNT[9:0] must be programmed such that the following condition is met:

VT > VDP + VDPS

VDP: Vertical display period

Use VDPCNT[9:0]/LCDC_VDISP register to set the vertical display period (= vertical panel resolution).

VDP = VDPCNT[9:0] + 1 [lines]

VDPCNT[9:0] must be programmed such that the following condition is met:
VT ≥ VDP + 1

VDPS: Vertical display period start position

Use VDPSCNT[9:0]/LCDC_VDPS register to set the vertical display period start position for the TFT panel.

VDPS = VDPSCNT[9:0] [lines]

VDPSCNT[9:0] must be programmed such that the following condition is met:
VT > VDP + VDPS

VPS: Vertical sync pulse start position

Use FPFRAME_ST[9:0]/LCDC_FPFR register to set the vertical sync pulse (FPFRAME or SPS) start position for the TFT panel.

VPS = FPFRAME_ST[9:0] [lines] = FPFRAME_ST[9:0] × HT [Ts]

VPW: Vertical sync pulse width

Use FPFRAME_WD[6:0]/LCDC_FPFR register to set the vertical sync pulse width for the TFT panel.

VPW = FPFRAME_WD[6:0] + 1 [lines] = (FPFRAME_WD[6:0] + 1) × HT [Ts]

Vertical sync pulse polarity

Use FPFRAME_POL/LCDC_FPFR register to set the vertical sync pulse polarity for the TFT panel.

FPFRAME_POL = 1: Active High
FPFRAME_POL = 0: Active low (default)

Vertical sync pulse offset

The vertical sync pulse position and width that are basically set in line units can be adjusted in pixel clock units.

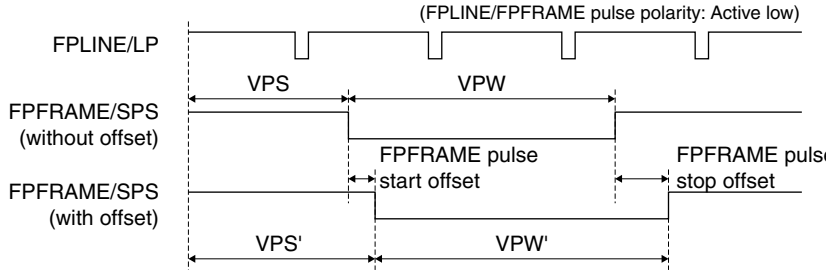


Figure 28.5.3.3 Vertical Sync Pulse Offset

Use FPFRAME_STOFS[9:0]/LCDC_FPFROFS register and FPFRAME_STPOFS[9:0]/LCDC_FPFROFS register to adjust the vertical sync pulse start and stop positions.

VPS' = FPFRAME_ST[9:0] × HT + FPFRAME_STOFS[9:0] [Ts]
VPW' = (FPFRAME_WD[6:0] + 1) × HT - FPFRAME_STOFS[9:0] + FPFRAME_STPOFS[9:0] [Ts]

FPSHIFT (DCLK) signal

The FPSHIFT (DCLK) signal polarity for TFT panels can be selected using FPSHIFT_POL/LCDC_TFTSO register.
When TFT panel is selected (PANELSEL/LCDC_DISPMOD register = 1), the FPSHIFT (DCLK) clock does not stop even in the horizontal non-display period by the default setting. To stop the FPSHIFT clock during the horizontal non-display period, set FPSHIFT_MSK/LCDC_DISPMOD register to 1.

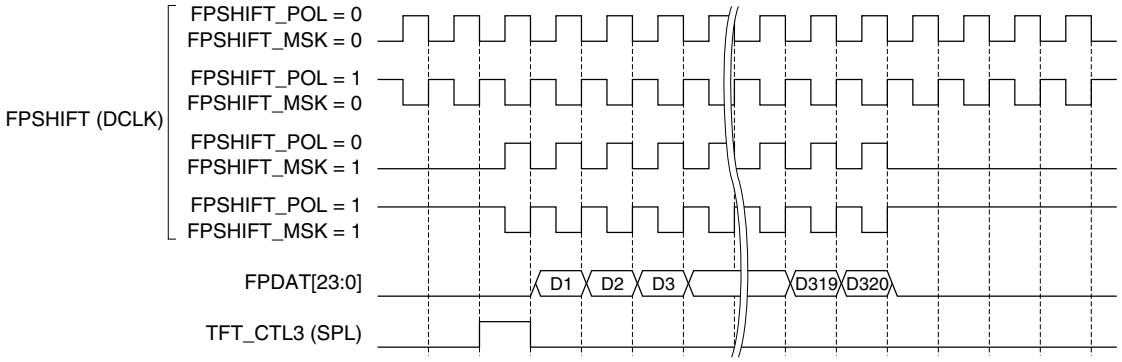


Figure 28.5.3.4 FPSHIFT (DCLK) Variations

TFT_CTL1 (CLS) pulse start/stop offset

The TFT_CTL1 (CLS) pulse position and width can be specified in pixel clock cycles. Use CTL1ST[9:0]/LCDC_TFT_CTL1 register to set the pulse start position and CTL1STP[9:0]/LCDC_TFT_CTL1 register to set the pulse stop position. These values should be specified an offset from the FPLINE pulse start position.

By setting this register, the TFT_CTL1 pulse width is set to CTL1STP[9:0] - CTL1ST[9:0] + 1 [Ts].

To program the TFT_CTL1 pulse, CTL1CTL/LCDC_TFTSO register and CTLCNT_RUN/LCDC_TFTSO register must be set to 1.

When CTL1CTL is set to 0 (default), the TFT_CTL1 pulse is toggled at the FPLINE pulse start edge.

The TFT_CTL1 and TFT_CTL0 signals can be swapped using CTL01SWAP/LCDC_TFTSO register.

TFT_CTL1 pin: CLS output (CTL01SWAP = 0), PS output (CTL01SWAP = 1)

TFT_CTL0 pin: PS output (CTL01SWAP = 0), CLS output (CTL01SWAP = 1)

TFT_CTL0 (PS) pulse start/stop offset

The TFT_CTL0 (PS) pulse position and width can be specified in pixel clock cycles. Use CTL0ST[9:0]/LCDC_TFT_CTL0 register to set the pulse start position and CTL0STP[9:0]/LCDC_TFT_CTL0 register to set the pulse stop position. These values should be specified an offset from the FPLINE pulse start position.

By setting this register, the TFT_CTL0 pulse width is set to CTL0STP[9:0] - CTL0ST[9:0] + 1 [Ts].

To program the TFT_CTL0 pulse, CTLCNT_RUN must be set to 1.

The TFT_CTL1 and TFT_CTL0 signals can be swapped using CTL01SWAP.

TFT_CTL2 (REV) delay

Use CTL2DLY[9:0]/LCDC_TFT_CTL2 register to set the TFT_CTL2 toggle edge delay time from the FPLINE pulse start edge. To program the TFT_CTL2 delay time, CTLCNT_RUN must be set to 1.

28.5.4 Display Modes

By setting the LCDC_DISPMOD register, a display mode (color/mono, use of LUT) and a bpp mode (color depth or gray levels) can be selected according to the LCD panel used as shown in Table 28.5.4.1.

Table 28.5.4.1 LCD Panel Type and Display Mode

Panel	LCDC_DISPMOD register			Display mode		Available bpp mode						
	PANELSEL	COLOR	LUTPASS	Color/Mono	LUT	1	2	4	8	12	16	24
TFT panel	1	1	1	Color	Bypass	✓	✓	✓	✓	✓	✓	✓
CSTN panel	0	1	1	Color	Bypass	✓	✓	✓	✓	✓	✓	
MSTN panel	0	0	1	Monochrome	Bypass	✓	✓	✓				
	0	0	0	Monochrome	MLUT	✓	✓	✓				

Table 28.5.4.2 lists the number of displayable colors/gray levels in different display modes and bpp modes.

Table 28.5.4.2 Displayable Colors/Gray Levels

bpp mode	Color mode	Monochrome mode	
	LUT bypassed	LUT bypassed	MLUT used
1 bpp	Black & White	Black & White	2 gray levels (entries) out of 16 gray levels
2 bpp	4 gray levels	4 gray levels	4 gray levels (entries) out of 16 gray levels
4 bpp	16 gray levels	16 gray levels	16 gray levels (entries) out of 16 gray levels
8 bpp	256 colors (R: 3 bits, G: 3 bits, B: 2 bits)		
12 bpp	4,096 colors (R: 4 bits, G: 4 bits, B: 4 bits)		
16 bpp	65,536 colors *1 (R: 5 bits, G: 6 bits, B: 5 bits)		
24 bpp	16,777,216 colors *2 (R: 8 bits, G: 8 bits, B: 8 bits)		

*1: Limited to 4,096 colors in CSTN panels.
*2: Cannot be used in CSTN panels.

The bpp mode is set using BPP[2:0]/LCDC_DISPMOD register.

Table 28.5.4.3 Bpp Mode Settings

BPP[2:0]	bpp mode
0x7	Reserved
0x6	24 bpp
0x5	16 bpp
0x4	12 bpp
0x3	8 bpp
0x2	4 bpp
0x1	2 bpp
0x0	1 bpp

(Default: 0x0)

To display the number of colors shown above, VRAM data must be configured to the appropriate format with or without the look-up table.
For detailed information on the VRAM data format and look-up tables, see Sections 28.5.5 to 28.5.7.

28.5.5 VRAM Data Format

This section describes the VRAM data format in each bpp mode.
Pixel data is placed in the VRAM beginning with the pixel at the upper-left corner of the screen. The subsequent pixels in the first line follow the first pixel and then the second to last line data follow. Each byte begins with the MSB and byte data are aligned in little-endian format. Color pixel data begins with Red bits, then Green and Blue bits follow.

1-bpp mode (black & white or 2 gray levels)

In 1-bpp mode, each bit in the VRAM corresponds to one pixel. The bit data represents the brightness (0 or 1) of the pixel when the look-up table is bypassed or an LUT entry number (0 or 1) when the look-up table is used.

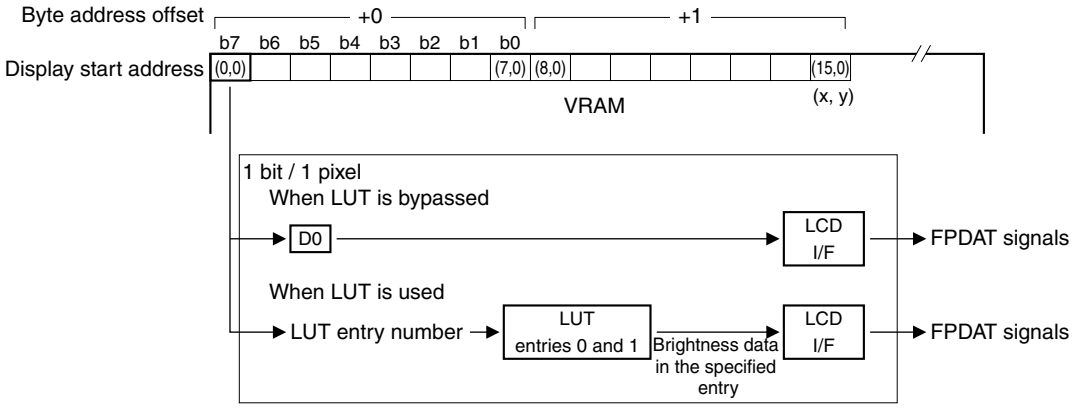
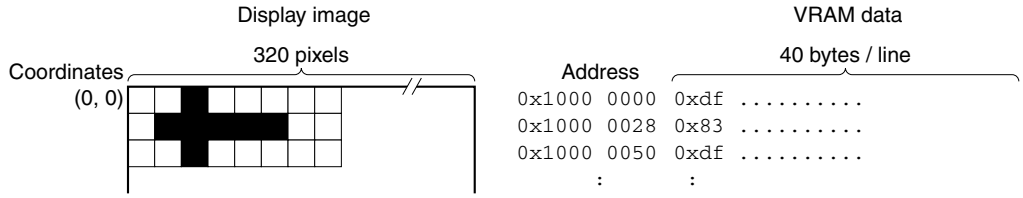


Figure 28.5.5.1 VRAM Data Format in 1-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.

Figure 28.5.5.2 Example of VRAM Data in 1-bpp Mode

2-bpp mode (4 gray levels)

In 2-bpp mode, each 2-bit data in the VRAM corresponds to one pixel. The 2-bit data represents the brightness (0 to 3) of the pixel when the look-up table is bypassed or an LUT entry number (0 to 3) when the look-up table is used.

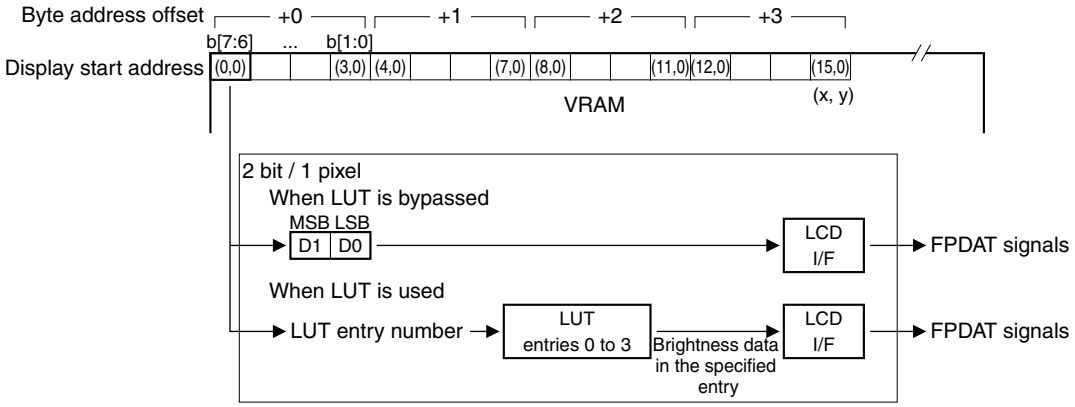
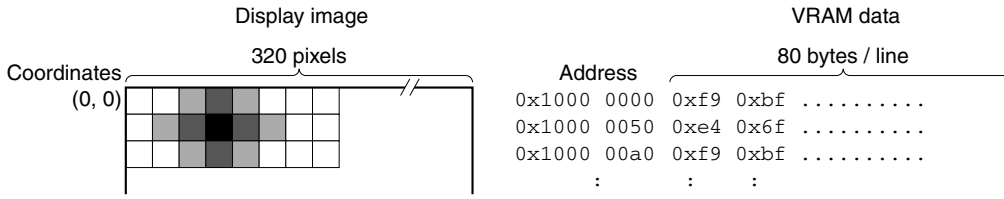


Figure 28.5.5.3 VRAM Data Format in 2-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.
Figure 28.5.5.4 Example of VRAM Data in 2-bpp Mode

4-bpp mode (16 gray levels)

In 4-bpp mode, each 4-bit data in the VRAM corresponds to one pixel. The 4-bit data represents the brightness (0 to 15) of the pixel when the look-up table is bypassed or an LUT entry number (0 to 15) when the look-up table is used.

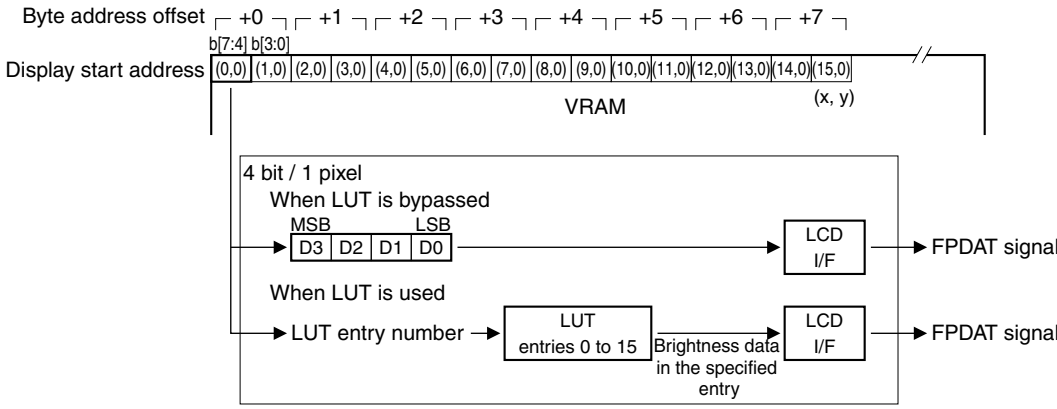
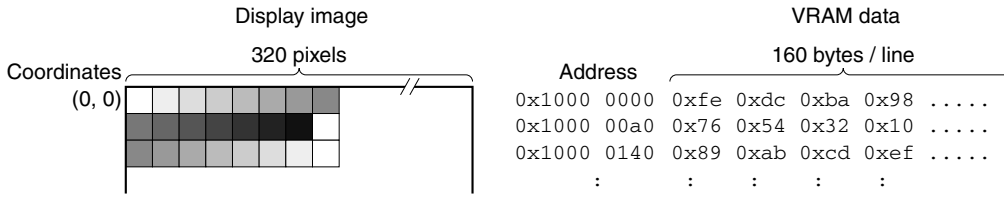


Figure 28.5.5.5 VRAM Data Format in 4-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.
Figure 28.5.5.6 Example of VRAM Data in 4-bpp Mode

8-bpp mode (256 colors)

In 8-bpp mode, each byte in the VRAM corresponds to one pixel. The bit[7:5] (3 bits), bit[4:2] (3 bits), and bit[1:0] (2 bits) in each byte represent the Red, Green, and Blue intensities, respectively, of the pixel. This mode does not support the look-up table.

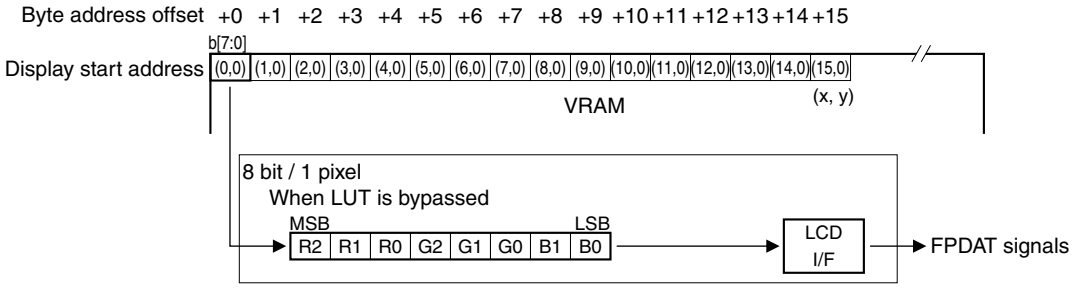
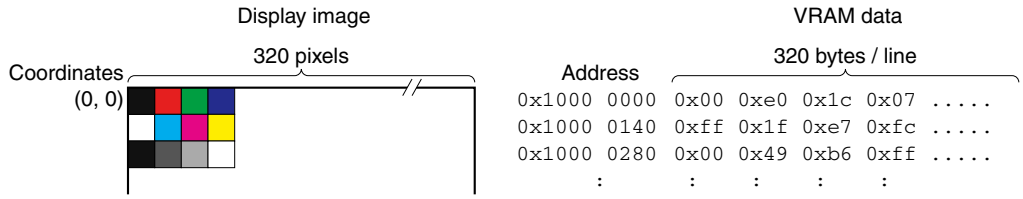


Figure 28.5.5.7 VRAM Data Format in 8-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.

Figure 28.5.5.8 Example of VRAM Data in 8-bpp Mode

12-bpp mode (4K colors)

In 12-bpp mode, each 12-bit data in the VRAM corresponds to one pixel. Display data must be manipulated in 3-byte units that consist of an even and odd X coordinate pixel pair. The bit[11:8] (4 bits), bit[7:4] (4 bits), and bit[3:0] (4 bits) in each 12-bit data represent the Red, Green, and Blue intensities, respectively, of the pixel. This mode does not support the look-up table.

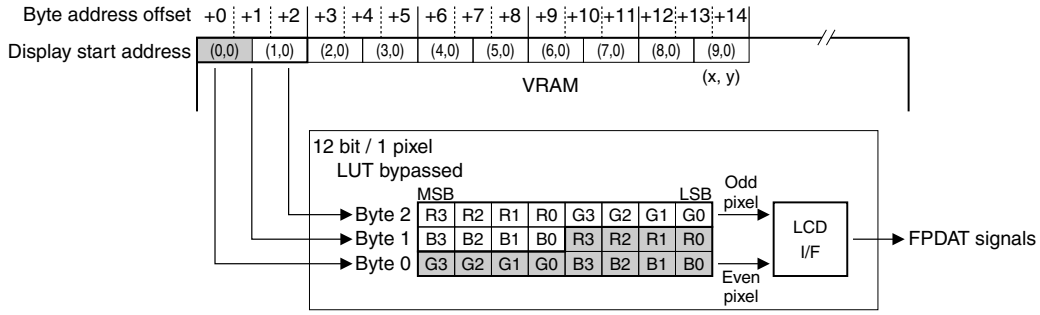
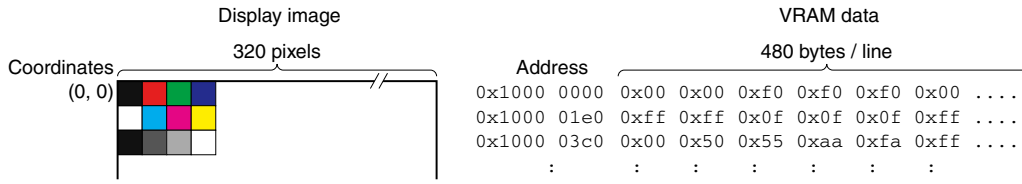


Figure 28.5.5.9 VRAM Data Format in 12-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.

Figure 28.5.5.10 Example of VRAM Data in 12-bpp Mode

16-bpp mode (64K colors for TFT, 4K colors for CSTN)

In 16-bpp mode, each 2-byte data in the VRAM corresponds to one pixel. The bit[15:11] (5 bits), bit[10:5] (6 bits), and bit[4:0] (5 bits) in each 16-bit data represent the Red, Green, and Blue intensities, respectively, of the pixel. This mode does not support the look-up table.

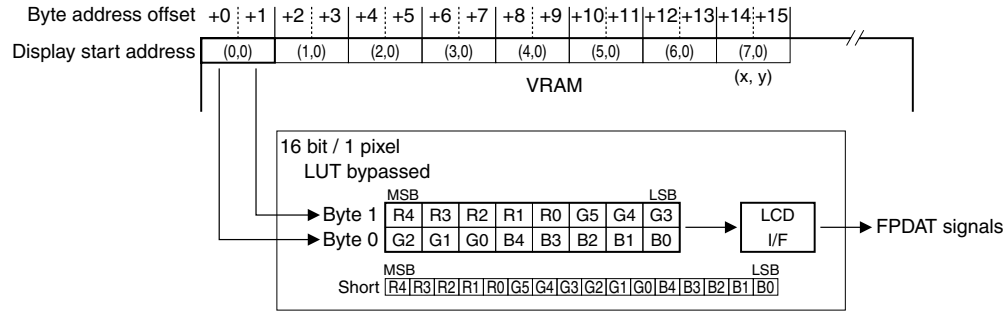
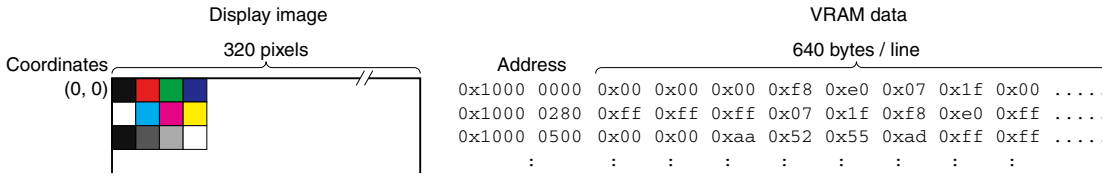


Figure 28.5.5.11 VRAM Data Format in 16-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.

Figure 28.5.5.12 Example of VRAM Data in 16-bpp Mode

Note: The LCDC supports up to 4K colors for CSTN panels even if 16-bpp mode is selected.

24-bpp mode (16M colors for TFT)

In 24-bpp mode, each 3-byte data in the VRAM corresponds to one pixel. The bit[23:16] (8 bits), bit[15:8] (8 bits), and bit[7:0] (8 bits) in each 24-bit data represent the Red, Green, and Blue intensities, respectively, of the pixel. This mode does not support the look-up table.

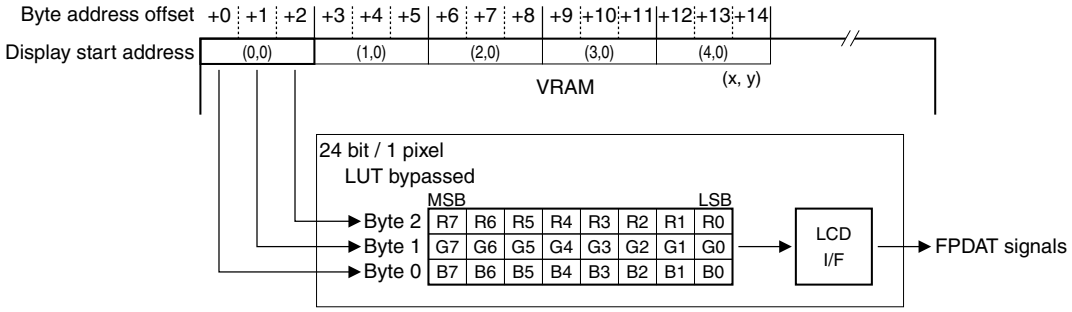
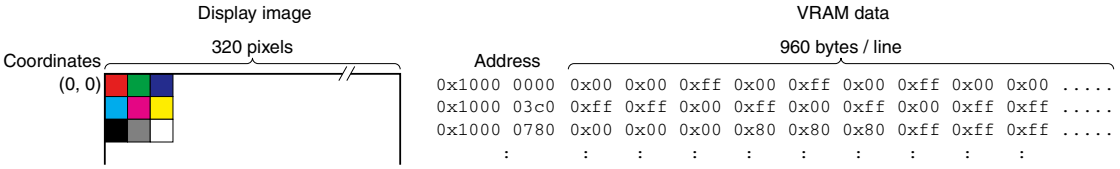


Figure 28.5.5.13 VRAM Data Format in 24-bpp Mode

Example

VRAM start address: 0x10000000
Screen width: 320 pixels
LUT: Bypassed
LCD characteristics: Data = 0 → Low LCD brightness



Note) Display may be inverted depending on the LCD panel used.

Figure 28.5.5.14 Example of VRAM Data in 24-bpp Mode

28.5.6 LUT Bypass Mode

In LUT bypass mode (LUTPASS/LCDC_DISPMOD register = 1), VRAM data are converted directly into the FPDAT signals. This mode always displays gray scale images in 1/2/4-bpp mode or color images in 8/12/16/24-bpp mode regardless of the COLOR/LCDC_DISPMOD register setting.

LUT bypass mode for TFT panel

When a TFT panel is used, make the following settings:

- Enable the LUT bypass function (LUTPASS/LCDC_DISPMOD register = 1).
- Select color mode (COLOR/LCDC_DISPMOD register = 1).

In LUT bypass mode, the FPDAT signals are generated directly from pixel data in the VRAM.
The following shows the relationship between pixel data and FPDAT signals in each bpp mode:

Notes: • When a TFT panel is used, be sure to set the LCDC into LUT bypass mode.

• The signal levels described in this section assume that SWINV/LCDC_DISPMOD register is set to 0. They will be inverted if SWINV is set to 1 (software inverse video enabled).

1-bpp mode (TFT panel, LUT bypassed)

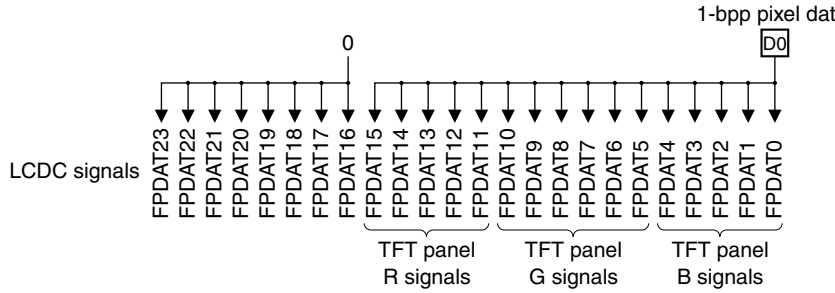


Figure 28.5.6.1 FPDAT Signals in LUT Bypass Mode (TFT panel, 1-bpp mode)

Table 28.5.6.1 Relationship between 1-bpp Pixel Data and FPDAT Signals

Pixel data	FPDAT[15:0] signals	FPDAT[23:16] signals
1	High (1)	Low (0)
0	Low (0)	Low (0)

2-bpp mode (TFT panel, LUT bypassed)

To ensure a uniform brightness, D1 and D0 are connected to the TFT panel RGB signals repeatedly.

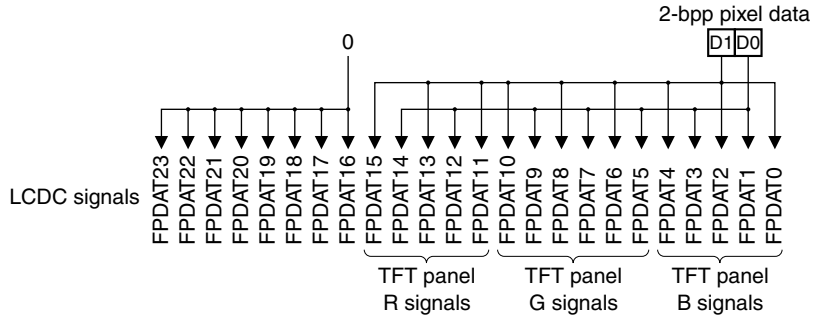


Figure 28.5.6.2 FPDAT Signals in LUT Bypass Mode (TFT panel, 2-bpp mode)

Table 28.5.6.2 Relationship between 2-bpp Pixel Data and FPDAT Signals

Pixel data	FPDAT0/2/4/6/8/10/11/13/15 signals	FPDAT1/3/5/7/9/12/14 signals	FPDAT[23:16] signals
0x3	High (1)	High (1)	Low (0)
0x2	High (1)	Low (0)	Low (0)
0x1	Low (0)	High (1)	Low (0)
0x0	Low (0)	Low (0)	Low (0)

4-bpp mode (TFT panel, LUT bypassed)

To ensure a uniform brightness, D3 to D0 are connected to the TFT panel RGB signals repeatedly.

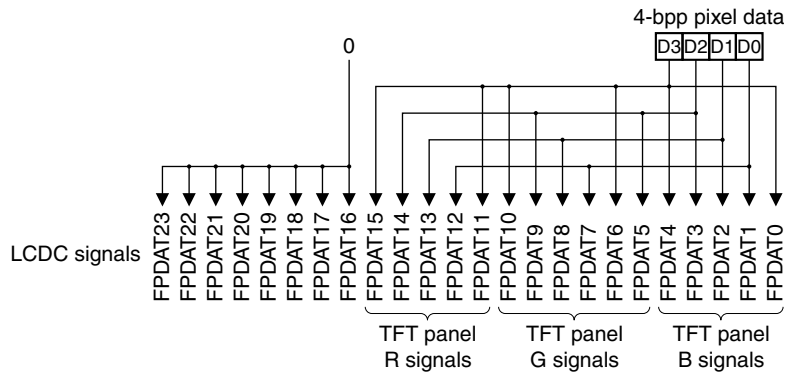


Figure 28.5.6.3 FPDAT Signals in LUT Bypass Mode (TFT panel, 4-bpp mode)

Table 28.5.6.3 Relationship between 4-bpp Pixel Data and FPDAT Signals

Pixel data	FPDAT0/4/6/10 /11/15 signals	FPDAT3/5/9/14 signals	FPDAT2/8/13 signals	FPDAT1/7/12 signals	FPDAT[23:16] signals
0xf	High (1)	High (1)	High (1)	High (1)	Low (0)
0xe	High (1)	High (1)	High (1)	Low (0)	Low (0)
0xd	High (1)	High (1)	Low (0)	High (1)	Low (0)
0xc	High (1)	High (1)	Low (0)	Low (0)	Low (0)
0xb	High (1)	Low (0)	High (1)	High (1)	Low (0)
0xa	High (1)	Low (0)	High (1)	Low (0)	Low (0)
0x9	High (1)	Low (0)	Low (0)	High (1)	Low (0)
0x8	High (1)	Low (0)	Low (0)	Low (0)	Low (0)
0x7	Low (0)	High (1)	High (1)	High (1)	Low (0)
0x6	Low (0)	High (1)	High (1)	Low (0)	Low (0)
0x5	Low (0)	High (1)	Low (0)	High (1)	Low (0)
0x4	Low (0)	High (1)	Low (0)	Low (0)	Low (0)
0x3	Low (0)	Low (0)	High (1)	High (1)	Low (0)
0x2	Low (0)	Low (0)	High (1)	Low (0)	Low (0)
0x1	Low (0)	Low (0)	Low (0)	High (1)	Low (0)
0x0	Low (0)	Low (0)	Low (0)	Low (0)	Low (0)

8-bpp mode (TFT panel, LUT bypassed)

To ensure a uniform brightness, R2–R0, G2–G0, and B1–B0 are connected to the TFT panel RGB signals repeatedly.

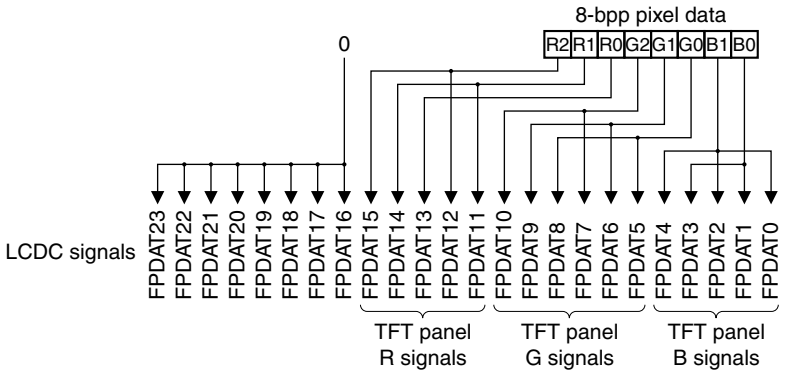


Figure 28.5.6.4 FPDAT Signals in LUT Bypass Mode (TFT panel, 8-bpp mode)

When a pixel data is 0x59 (R = 0x2, G = 0x6, B = 0x1), for example, the FPDAT signals will be configured as follows:

FPDAT15 = Low (0)
FPDAT14 = High (1)
FPDAT13 = Low (0)
FPDAT12 = Low (0)
FPDAT11 = High (1)
FPDAT10 = High (1)
FPDAT9 = High (1)
FPDAT8 = Low (0)
FPDAT7 = High (1)
FPDAT6 = High (1)
FPDAT5 = Low (0)
FPDAT4 = Low (0)
FPDAT3 = High (1)
FPDAT2 = Low (0)
FPDAT1 = High (1)
FPDAT0 = Low (0)

FPDAT[23:16] = Low (0)

12-bpp mode (TFT panel, LUT bypassed)

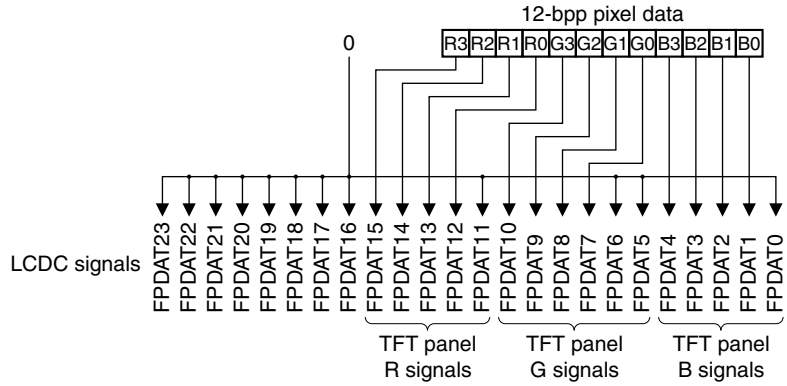


Figure 28.5.6.5 FPDAT Signals in LUT Bypass Mode (TFT panel, 12-bpp mode)

When a pixel data is 0xc51 (R = 0xb, G = 0x5, B = 0x1), for example, the FPDAT signals will be configured as follows:

FPDAT15 = High (1)
FPDAT14 = High (1)
FPDAT13 = Low (0)
FPDAT12 = Low (0)
FPDAT11 = Low (0)
FPDAT10 = Low (0)
FPDAT9 = High (1)
FPDAT8 = Low (0)
FPDAT7 = High (1)
FPDAT6 = Low (0)
FPDAT5 = Low (0)
FPDAT4 = Low (0)
FPDAT3 = Low (0)
FPDAT2 = Low (0)
FPDAT1 = High (1)
FPDAT0 = Low (0)

FPDAT[23:16] = Low (0)

16-bpp mode (TFT panel, LUT bypassed)

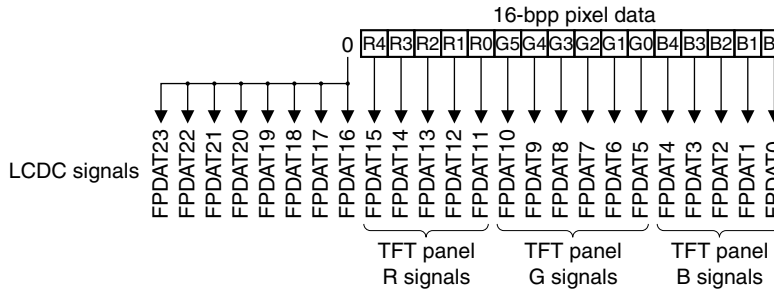


Figure 28.5.6.6 FPDAT Signals in LUT Bypass Mode (TFT panel, 16-bpp mode)

When a pixel data is 0x5b37 (R = 0xb, G = 0x19, B = 0x17), for example, the FPDAT signals will be configured as follows:

FPDAT15 = Low (0)
FPDAT14 = High (1)
FPDAT13 = Low (0)
FPDAT12 = High (1)
FPDAT11 = High (1)
FPDAT10 = Low (0)
FPDAT9 = High (1)
FPDAT8 = High (1)
FPDAT7 = Low (0)
FPDAT6 = Low (0)
FPDAT5 = High (1)
FPDAT4 = High (1)
FPDAT3 = Low (0)
FPDAT2 = High (1)
FPDAT1 = High (1)
FPDAT0 = High (1)

FPDAT[23:16] = Low (0)

24-bpp mode (TFT panel, LUT bypassed)

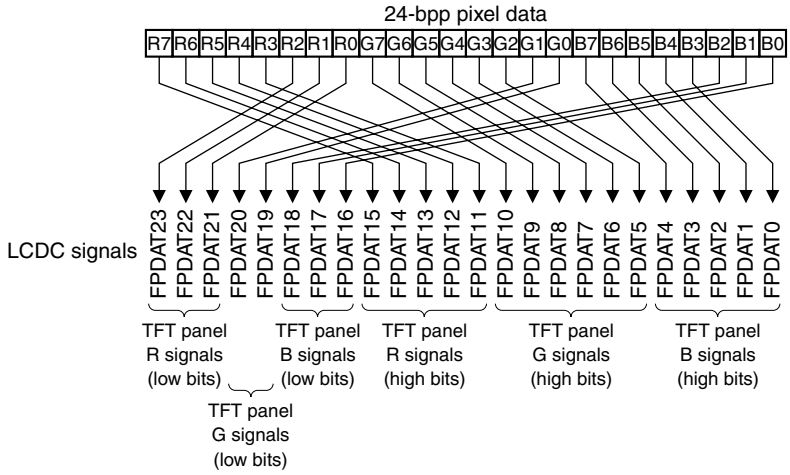


Figure 28.5.6.7 FPDAT Signals in LUT Bypass Mode (TFT panel, 24-bpp mode)

When a pixel data is 0x123456 (R = 0x12, G = 0x34, B = 0x56), for example, the FPDAT signals will be configured as follows:

FPDAT15 = Low (0)	FPDAT10 = Low (0)	FPDAT4 = Low (0)
FPDAT14 = Low (0)	FPDAT9 = Low (0)	FPDAT3 = High (1)
FPDAT13 = Low (0)	FPDAT8 = High (1)	FPDAT2 = Low (0)
FPDAT12 = High (1)	FPDAT7 = High (1)	FPDAT1 = High (1)
FPDAT11 = Low (0)	FPDAT6 = Low (0)	FPDAT0 = Low (0)
FPDAT23 = Low (0)	FPDAT5 = High (1)	FPDAT18 = High (1)
FPDAT22 = High (1)	FPDAT20 = Low (0)	FPDAT17 = High (1)
FPDAT21 = Low (0)	FPDAT19 = Low (0)	FPDAT16 = Low (0)

LUT bypass mode for CSTN panel

When a CSTN panel is used, LUT bypass mode is effective if the conditions shown below are all met.

- The LUT bypass function is enabled (LUTPASS/LCDC_DISPMOD register = 1).
- Color mode is selected (COLOR/LCDC_DISPMOD register = 1).
- 1, 2, 4, 8, 12, or 16-bpp mode is selected (BPP[2:0]/LCDC_DISPMOD register = 0x0 to 0x5).

In LUT bypass mode, pixel data in the VRAM specify a halftone to generate the FPDAT signals.
The following shows the relationship between pixel data and FPDAT signals in each bpp mode:

Notes: • When a CSTN panel is used, be sure to set the LCDC into LUT bypass mode.

• CSTN panels turn the pixels on and off periodically to generate halftones. The FPDAT signals do not represent colors directly in contrast to TFT panels.

1-bpp mode (CSTN panel, LUT bypassed)

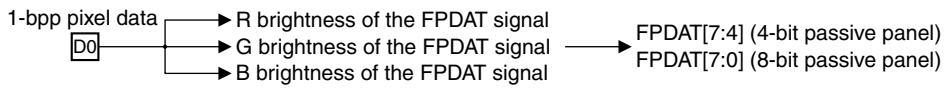


Figure 28.5.6.8 FPDAT Signals in LUT Bypass Mode (CSTN panel, 1-bpp mode)

2-bpp mode (CSTN panel, LUT bypassed)

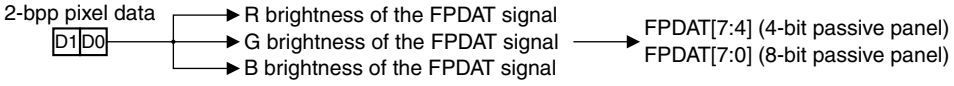


Figure 28.5.6.9 FPDAT Signals in LUT Bypass Mode (CSTN panel, 2-bpp mode)

4-bpp mode (CSTN panel, LUT bypassed)

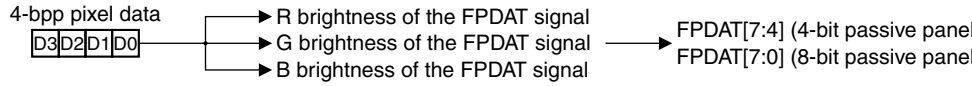


Figure 28.5.6.10 FPDAT Signals in LUT Bypass Mode (CSTN panel, 4-bpp mode)

8-bpp mode (CSTN panel, LUT bypassed)

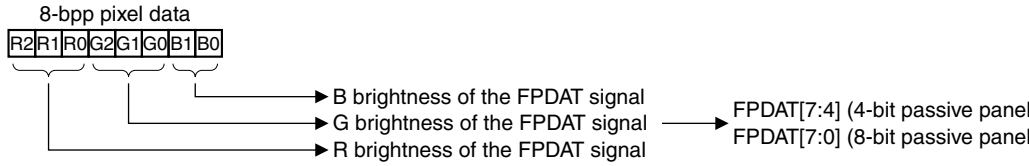


Figure 28.5.6.11 FPDAT Signals in LUT Bypass Mode (CSTN panel, 8-bpp mode)

12-bpp mode (CSTN panel, LUT bypassed)

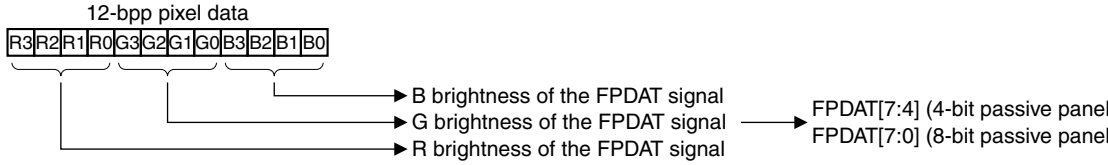


Figure 28.5.6.12 FPDAT Signals in LUT Bypass Mode (CSTN panel, 12-bpp mode)

16-bpp mode (CSTN panel, LUT bypassed)

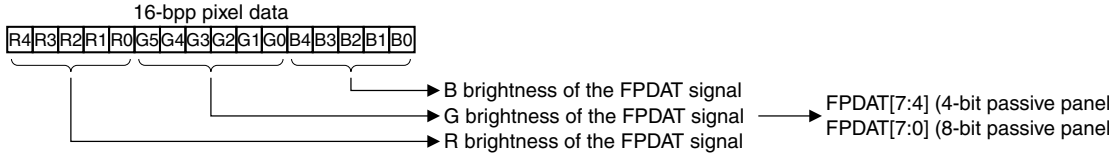


Figure 28.5.6.13 FPDAT Signals in LUT Bypass Mode (CSTN panel, 16-bpp mode)

LUT bypass mode for MSTN panel

When an MSTN panel is used, LUT bypass mode is effective if the conditions shown below are all met.

- The LUT bypass function is enabled (LUTPASS/LCDC_DISPMOD register = 1).
- Monochrome mode is selected (COLOR/LCDC_DISPMOD register = 0).
- 1, 2, or 4-bpp mode is selected (BPP[2:0]/LCDC_DISPMOD register = 0x0 to 0x2).

In LUT bypass mode, pixel data in the VRAM specify a gray level to generate the FPDAT signals.
The following shows the relationship between pixel data and FPDAT signals in each bpp mode:

Note: An FPDAT signal is assigned to a pixel and 4 or 8 pixel data are sent to the MSTN panel at the same time. MSTN panels turn the pixels on and off periodically to generate gray levels. The FPDAT signals do not represent gray levels directly in contrast to TFT panels.

1-bpp mode (MSTN panel, LUT bypassed)

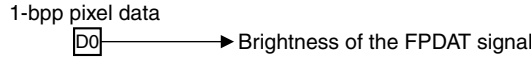


Figure 28.5.6.14 FPDAT Signals in LUT Bypass Mode (MSTN panel, 1-bpp mode)

2-bpp mode (MSTN panel, LUT bypassed)

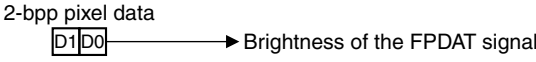


Figure 28.5.6.15 FPDAT Signals in LUT Bypass Mode (MSTN panel, 2-bpp mode)

4-bpp mode (MSTN panel, LUT bypassed)

Figure 28.5.6.16 FPDAT Signals in LUT Bypass Mode (MSTN panel, 4-bpp mode)

28.5.7 Look-up Table

When the look-up table is enabled, a pixel data in the VRAM is used as an index to the look-up table, so that FPDAT signals are generated based on the gray level stored in the LUT entry indicated by the VRAM data, before being output to the LCD panel.
The S1C33L27 includes the monochrome look-up table MLUT.

Monochrome look-up table (MLUT)

The LCDC uses MLUT to convert VRAM data into gray scale data if the conditions shown below are all met.

- The LUT bypass function is disabled (LUTPASS/LCDC_DISPMOD register = 0).
- Monochrome mode is selected (COLOR/LCDC_DISPMOD register = 0).
- 1, 2, or 4-bpp mode is selected (BPP[2:0]/LCDC_DISPMOD register = 0x0 to 0x2).

The monochrome look-up table consists of 4 bits × 16 entries as shown in the figure below. The software can set a gray level out of 16 levels in each entry. The entries to be used depend on the bpp mode selected.

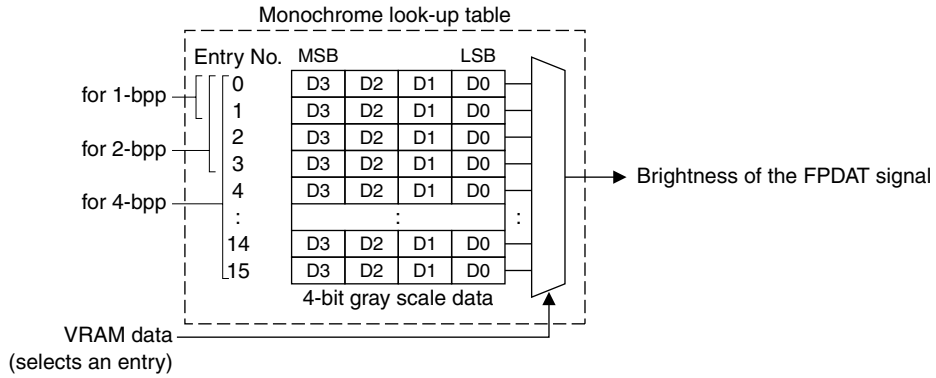


Figure 28.5.7.1 Monochrome Look-up Table Configuration

Note: An FPDAT signal is assigned to a pixel and 4 or 8 pixel data are sent to the MSTN panel at the same time. MSTN panels turn the pixels on and off periodically to generate gray levels. The FPDAT signals do not represent gray levels directly in contrast to TFT panels.

Setting data to the monochrome look-up table

Use the monochrome look-up table data registers for writing and reading 4-bit gray scale data to/from the look-up table. The monochrome look-up table data registers are mapped to addresses 0x302090 and 0x302094.

Table 28.5.7.1 Monochrome Look-up Table Registers

LUT entry	LUT bit/register
0	MLUT0[3:0]/LCDC_MLUT0
1	MLUT1[3:0]/LCDC_MLUT0
2	MLUT2[3:0]/LCDC_MLUT0
3	MLUT3[3:0]/LCDC_MLUT0
4	MLUT4[3:0]/LCDC_MLUT0
5	MLUT5[3:0]/LCDC_MLUT0
6	MLUT6[3:0]/LCDC_MLUT0
7	MLUT7[3:0]/LCDC_MLUT0
8	MLUT8[3:0]/LCDC_MLUT1
9	MLUT9[3:0]/LCDC_MLUT1
10	MLUT10[3:0]/LCDC_MLUT1
11	MLUT11[3:0]/LCDC_MLUT1
12	MLUT12[3:0]/LCDC_MLUT1
13	MLUT13[3:0]/LCDC_MLUT1
14	MLUT14[3:0]/LCDC_MLUT1
15	MLUT15[3:0]/LCDC_MLUT1

In addition settings shown above, the LCDC reload function can be used to replace the monochrome look-up table data by the values prepared in the memory. For more information on the LCDC reload function, see Section 28.9.

28.5.8 Frame Rates

The frame rate is calculated from the LCD panel's horizontal and vertical total periods, and pixel clock frequency, as shown below.

$$\text{Frame rate} = \frac{f_{LCDC\_CLK}}{HT \times VT}$$

f_{LCDC_CLK}: Pixel clock frequency
f_{LCDC_CLK} = OSC3/1 to OSC3/32 (Hz) see Section 28.4.2, "Setting the LCDC Clock."

HT: Horizontal total period
HT = (HTCNT[6:0] + 1) × 8 (Ts) where Ts = pixel clock period

VT: Vertical total period
VT = VTCNT[9:0] + 1 (lines)

28.5.9 Other Settings

MOD rate

The period during which the MOD signal is switched can be set using the MOD[5:0]/LCDC_MODR register.
MOD = 0x0: MOD signal switched at a period of the FPFRAME signal (default)
MOD = other than 0x0: Switched at a period of MOD + 1 FPLINE pulses

Repeating of the FRM pattern

This setup item is provided for EL panels. Whether the frame-rate modulation pattern is to be repeated every 0x40000 frames (counted by the internal frame counter) can be set using FRMRPT/LCDC_DISPMOD register.
FRMRPT = 1: FRM pattern repeated (for EL panel)
FRMRPT = 0: FRM pattern not repeated (default)

28.6 Display Control

28.6.1 Controlling LCD Power Up/Down

The LCD controller is activated when the LCDC clocks are supplied from the CMU. Following initial reset, the LCD controller is set in power-save mode. Supplying the clocks does not immediately cause the LCD panel to initiate a power-up sequence and start displaying data. The LCD panel is placed in power-save mode, with all LCD signal output pins fixed low.
To change the LCD controller from power-save mode back into normal mode, set the PSAVE[1:0]/LCDC_PSAVE register to 0x3. The LCD controller starts a power-up sequence from that point, and outputs LCD signals. Conversely, to change from normal mode to power-save mode, set PSAVE[1:0] to 0x0. The LCD controller starts a power-down sequence from that point, and drives the LCD signals low.
The LCD control registers and look-up tables can be accessed even in power-save mode.
If the power to the LCD panel is turned on or off while LCD signals are not being correctly output, the panel may be damaged. Therefore, the power to the LCD panel must be turned on only after the LCD controller starts controlling LCD signals. Use an I/O port to control the power to the LCD panel for this purpose. When LCD signals have no effect, disable the LCD power supply by controlling the port output; when LCD signals become effective, enable the LCD power supply using the port.

The procedure for initializing the LCD at power-on is summarized below.
1. Configure the clocks, pins, and display memory area (refer to “28.4 System Settings”).
2. Set the LCD-panel parameters, display mode, and look-up tables (refer to “28.5 Setting the LCD Panel”).
3. Enable the LCDC interrupt if necessary.
4. Write display data to the display memory.
5. Set the display start address (refer to “28.6.2 Main Window Display Start Address and Virtual Screen Settings”).
6. Place the LCD controller in normal mode (PSAVE = 0x3).
7. The LCD controller starts outputting the LCD signals.
8. Wait time should be inserted depending on the LCD panel power source.
9. Control the port to turn the LCD panel power on.

The following is the power-down procedure.
1. Control the port to turn the LCD panel power off.
2. Wait time should be inserted depending on the LCD panel power source.
3. Place the LCD controller in power-save mode (PSAVE = 0x0).
4. The LCD controller pulls LCD signals down to low.

28.6.2 Main Window Display Start Address and Virtual Screen Settings

Main window display start address

The display memory address from which to start display for the main window can be changed as desired using the LCDC_MAINADR register. The start address set in this register corresponds to the upper left edge of the LCD panel. Note that a word boundary address (A[1:0] = 0b00) in IVRAM or the external VRAM must be specified to this register.

Main screen address offset for virtual screen

The S1C33L27 LCDC supports a virtual screen feature to load to the VRAM and display images with a different size from that of the LCD panel.

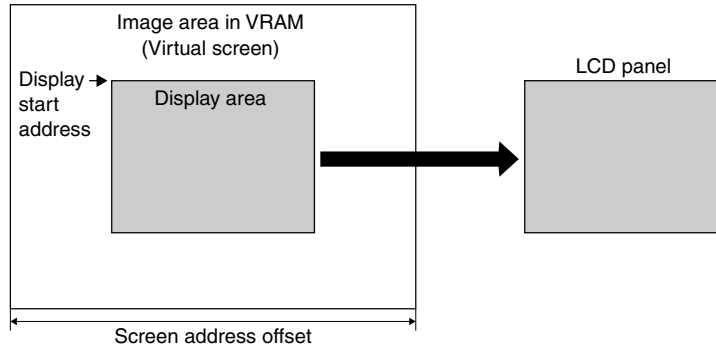


Figure 28.6.2.1 Virtual Screen Feature

The screen address offset is the number of words corresponding to the image width (virtual screen width). The LCDC manipulates display data in units of words. Therefore, the image width (number of pixels) must be a multiple of (32 bits ÷ bpp).
The screen address offset is calculated as follows:

Main screen address offset = Virtual screen width in pixels × bpp / 32 [words]

Set this value to MW_OFS[11:0]/LCDC_MAINOFS register.

Example: LCD panel size = 320 × 240 pixels, 8-bpp mode, image (virtual screen) size = 640 × 480 pixels
Display line width = 320 × 8 / 32 = 80 (=0x50) [words] (= 320 [bytes])
MW_OFS[11:0] = 640 × 8 / 32 = 160 (=0xa0) [words] (= 640 [bytes])

If the image data start address is 0x10000000 and the image area ([X1, Y1]–[X2, Y2] = [160, 120]–[479, 359]) is displayed in the main window, the main window start address is 0x10012ca0.
$(X1 \times 8 / 32)_{[word]} + MW\_OFS[11:0]_{[word]} \times Y1 = 0x4b28$ [words] = 0x12ca0 [bytes]
Main window start address = 0x10000000 + 0x12ca0 = 0x10012ca0

The LCDC determines the addresses of the first and end pixels in each line as follows:
First pixel address in *N*th line = Display start address + (*N* -1) × MW_OFS[11:0][word]
End pixel address in *N*th line = Display start address + (*N* -1) × MW_OFS[11:0][word]
+ Display line width[word] -1
1st line start address = 0x10012ca0
1st line end address = 0x10012ca0 + (0x50 -1) × 4 = 0x10012dc4
2nd line start address = 0x10012ca0 + 1 × 0xa0 × 4 = 0x10012f20
2nd line end address = 0x10012f20 + (0x50 -1) × 4 = 0x10013044
:
240th line start address = 0x10012ca0 + 239 × 0xa0 × 4 = 0x10038220
240th line end address = 0x10038220 + (0x50 -1) × 4 = 0x10038344

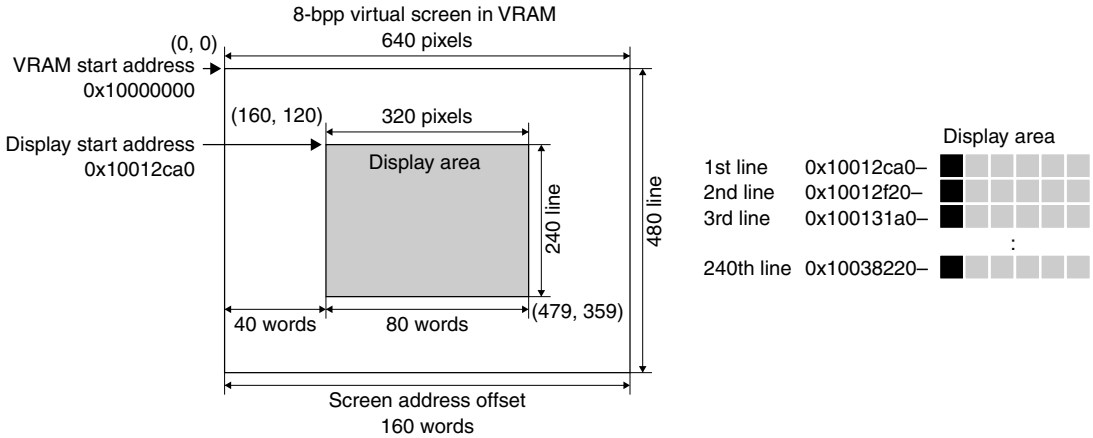


Figure 28.6.2.2 Example of Virtual Screen Configuration

28.6.3 Writing Display Data

The LCD controller may generate an interrupt at the beginning with the vertical non-display period after finishing each frame refresh sequence. Furthermore, VNDPF/LCDC_PSAVE register is provided and is set to 1 if the display is in a vertical non-display period.
To eliminate screen flicker, display data, LUT data and the display buffer should be changed in a vertical non-display period by using this interrupt or VNDPF.
For more information on the LCDC interrupt, see Section 28.7, "LCDC Interrupt."

28.6.4 Inverting and Blanking the Display

The display can be blanked (the entire screen turned black or white) without rewriting the contents of the VRAM. Setting BLANK/LCDC_DISPMOD register to 1 causes the FPDAT signal to go low or high, blanking the display. Setting it to 0 turns the display back on. Whether the screen turns black or white is determined by SWINV/LCDC_DISPMOD register described below.

The display can be inverted simply by manipulating a control bit. Setting SWINV to 1 inverts the display, and setting it to 0 returns the display to normal. This is accomplished by inverting the display data output from the LCDC. The screen can be made to blink using these operations. Make sure switching takes place within the vertical non-display period (VNDPF = 1).

28.6.5 Picture-in-Picture Plus and Sub-Window

Picture-in-Picture Plus enables a sub-window to be overlaid on the main window. The sub-window may be positioned anywhere within the main window and is controlled through the sub-window control registers. The sub-window retains the same color depth as the main window. The sub-window also supports the virtual screen feature the same as the main window.
The following diagram shows the sub-window configuration parameters.

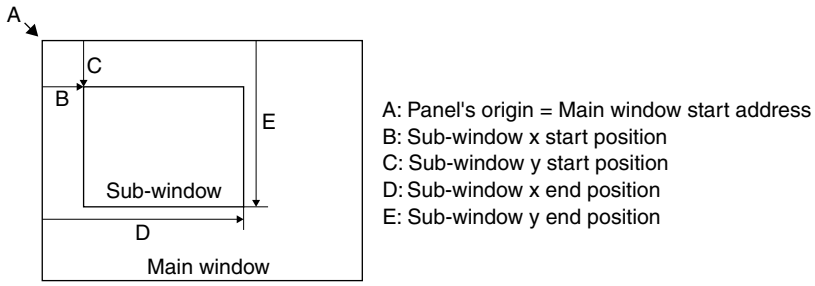


Figure 28.6.5.1 Sub-window Configuration Parameters

Display memory for the sub-window

The display data for the sub-window can be stored in IVRAM or the external VRAM. If the same memory as the main window is used, make sure that the display data areas for the main window and sub-window do not overlap.
The sub-window start address is specified by the LCDC_SUBADR register in the same manner as the main window. The start address set in the LCDC_SUBADR register corresponds to the upper left corner of the sub-window.
Note that a word boundary address (A[1:0] = 0b00) in IVRAM or the external VRAM must be specified to this register.
The sub-window width must be a multiple of (32 bits ÷ bpp).

Sub-window coordinates

The display position and size of the sub-window are configured with the X and Y coordinates of the start position (upper left corner) and end position (lower right corner).

Specify the sub-window start position using PIP_XSTART[9:0]/LCDC_SUBSP register for the X coordinate and PIP_YSTART[9:0]/LCDC_SUBSP register for the Y coordinate. Use PIP_XEND[9:0]/LCDC_SUBEP register for specifying the X coordinate of the end position and PIP_YEND[9:0]/LCDC_SUBEP register for the Y coordinate.
The X coordinate should be specified with the number of data words converted from the number of pixels from the LCD panel origin point according to the bpp mode. Therefore, it can be specified in (32 bits ÷ bpp) pixel increments.

1-bpp mode: 1-word = 32-pixel units
2-bpp mode: 1-word = 16-pixel units
4-bpp mode: 1-word = 8-pixel units
8-bpp mode: 1-word = 4-pixel units
12-bpp mode: 3-word = 8-pixel units (because the value must be an integer)
16-bpp mode: 1-word = 2-pixel units
24-bpp mode: 3-word = 4-pixel units (because the value must be an integer)

For example, to specify the sub-window horizontal start position as 80 pixels in 8-bpp mode, set PIP_XSTART[9:0] to 20.

The Y coordinate is specified with the number of lines from the LCD panel origin point in line units. For example, to specify the sub-window vertical start position as 60 lines, set PIP_YSTART[9:0] to 60.

Note: The sub-window must be located within the main window.

Sub-screen address offset for virtual screen

The virtual screen feature can also be used for the sub-window. Specify the screen address offset using SW_OFS[11:0]/LCDC_SUBOFS register.

Sub-screen address offset = Virtual screen width in pixels × bpp / 32 [words]

See "Main screen address offset for virtual screen" in Section 28.6.2 for more information on the virtual screen and the configurations.

Sub-window display control

The Picture-in-Picture Plus function is enabled and the sub-window is displayed by setting PIP_EN/LCDC_SUBSP register to 1. This bit must be set after the sub-window configuration parameters are set up.
At initial reset, PIP_EN is set to 0 and sub-window is disabled for display.

Sub-window configuration example

The following shows an example to configure main and sub-windows.
[Conditions]

- LCD panel resolution: 320 × 240 pixels
- bpp mode: 4 bpp (16 shades of gray/16 colors)
- Memory used: External SDRAM (Area 19, from 0x10000000)
- Virtual main screen size: 800 × 600 pixels, located at 0x10000000
- Virtual sub-screen size: 640 × 480 pixels, located at 0x10100000 (larger screen than the main screen, such as 1024 × 768 pixels can also be configured.)
- Sub-window size: 160 × 120 pixels
- Sub-window start position: X = 80 pixels, Y = 60 pixels

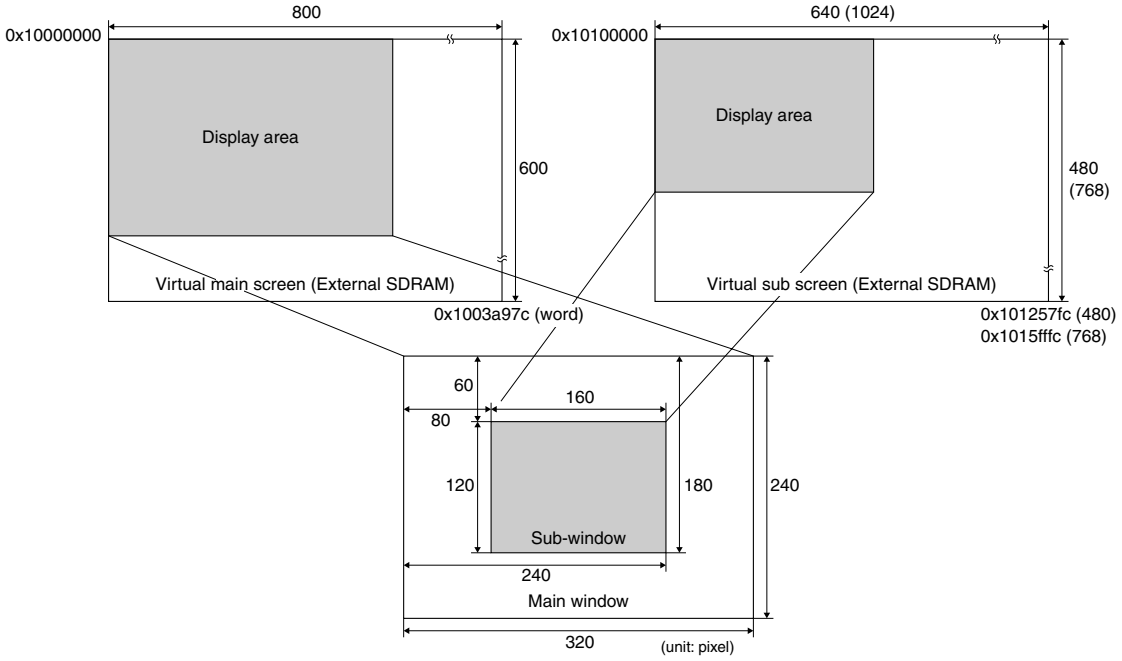


Figure 28.6.5.2 Sub-Window Configuration Example

1. Main window start address

   LCDC_MAINADR register = 0x10000000 to 0x1003a97c (word boundary address)

   This register is used to change the main window location in the virtual main screen VRAM.

2. Main screen address offset

   MW_OFS[11:0] = 800 pixels × 4 bpp ÷ 32 bits = 100 words
   LCDC_MAINOFS register = 100 = 0x64

3. Sub-window start address

   LCDC_SUBADR register = 0x10100000 to 0x101257fc (for 640 × 480 pixels, word boundary address)
   LCDC_SUBADR register = 0x10100000 to 0x1015fffc (for 1024 × 768 pixels, word boundary address)

   This register is used to change the sub-window location in the virtual sub-screen VRAM.

4. Sub-screen address offset

   SW_OFS[11:0] = 640 pixels × 4 bpp ÷ 32 bits = 80 words (for 640 × 480 pixels)
   LCDC_SUBOFS register = 80 = 0x50

   SW_OFS[11:0] = 1024 pixels × 4 bpp ÷ 32 bits = 128 words (for 1024 × 768 pixels)
   LCDC_SUBOFS register = 128 = 0x80

5. Sub-window end position

   PIP_XEND[9:0] = (80 + 160) pixels × 4 bpp ÷ 32 bits - 1 = 29 words = 0x1d
   PIP_YEND[9:0] = 60 + 120 lines - 1 = 179 lines = 0xb3
   LCDC_SUBEP register = 0x00b3001d

6. Sub-window start position

   PIP_XSTART[9:0] = 80 pixels × 4 bpp ÷ 32 bits = 10 words = 0x0a
   PIP_YSTART[9:0] = 60 lines = 0x3c
   LCDC_SUBSP register = 0x803c000a

   The MSB in this register is the sub-window enable bit. Setting the MSB enables the sub-window to be displayed.

7. Note on using 12 bpp or 24 bpp mode
   If the LCDC is used in 12 bpp or 24 bpp mode, the addresses must be aligned with a 3-word boundary.

   LCDC_MAINADR register = 0x10000000 + 3n (words)
   LCDC_SUBADR register = 0x10100000 + 3n (words)
   PIP_XSTART[9:0] = 3n (words)
   PIP_XEND[9:0] = 3n + 2 (words)

28.7 LCDC Interrupt

The LCDC module can generate frame interrupts.

Frame interrupt

To use this interrupt, set FRINTEN/LCDC_INT register to 1. If FRINTEN is set to 0 (default), interrupt requests for this cause will not be sent to the ITC.
When a vertical non-display period begins after a frame refresh cycle (vertical display period) has finished, VNDPF/LCDC_PSAVE register is set to 1. At the same time, FRINTF/LCDC_PSAVE register is set to 1 and the LCDC outputs an interrupt signal to the interrupt controller (ITC) if frame interrupts are enabled (FRINTEN = 1). An interrupt occurs if other interrupt conditions are met.

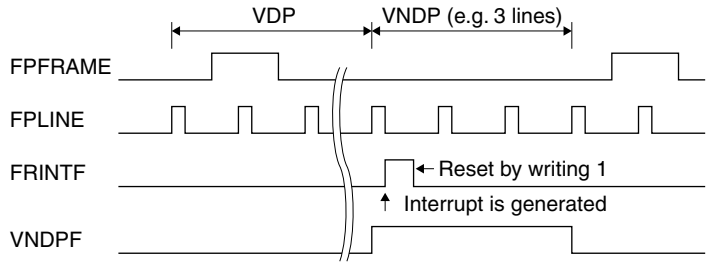


Figure 28.7.1 Frame Interrupt Timing

For more information on interrupt processing, see the “Interrupt Controller (ITC)” chapter.

28.8 Power Save

The LCD controller has a power-save mode. Use PSAVE[1:0]/LCDC_PSAVE register to set the power-save mode.

Table 28.8.1 Power-Save Mode Settings

PSAVE[1:0]	Mode
0x3	Normal operation
0x2	Reserved
0x1	Reserved
0x0	Power-save mode

(Default: 0x0)

Power-save mode

When the LCD controller enters this mode, all LCD signal output pins are dropped low, with the LCD panel placed in power-down mode. All operations of the LCD controller, other than accessing of its control registers and look-up tables are disabled.
The LCD controller is placed in power-save mode by setting PSAVE[1:0] to 0x0.
The LCD controller is taken out of power-save mode by setting PSAVE[1:0] to 0x3.

Comparison of power-save modes

The differences between power-save modes are summarized in Table 28.8.2.

Table 28.8.2 Differences between Power-Save Modes

Item	Power-save mode	Normal
Accessing I/O registers	Enabled	Enabled
Accessing look-up table	Enabled	Enabled
Accessing VRAM	Enabled	Enabled
Display (STN panels)	Inactive	Active
Display (TFT panels)	Inactive	Active
LCDC display-data-fetch operation	Inactive	Active
FPDAT[23:0] signals (STN, TFT panels)	Low	Active
FPSHIFT signal (STN panels)	Low	Active
FPLINE, FPFRAME, FPDRDY signals (STN panels)	Low	Active
FPSHIFT signal (TFT panels) when FPSHIFT_POL/LCDC_TFTSO register = 0	High	Active
FPSHIFT signal (TFT panels) when FPSHIFT_POL/LCDC_TFTSO register = 1	Low	Active
FPFRAME signal (TFT panels)	High/Low	Active
FPLINE signal (TFT panels)	High/Low	Active
TFT_CTL0 signal (TFT panels)	High/Low	Active
TFT_CTL1 signal (TFT panels)	High/Low	Active
TFT_CTL2 signal (TFT panels)	High/Low	Active
TFT_CTL3 signal (TFT panels)	Low	Active

28.9 Reload Functions

The LCDC supports two reload functions, Control table reload function and LUT reload function, that reset the LCDC control registers and look-up tables using the reload data prepared in the memory.

Control table reload function

The control table reload function is used to back up and restore LCDC control register settings.
The table below shows the contents of the reload table used in the control table reload function.

Table 28.9.1 Reload Table Contents (LCDC Registers)

Address	Control register
Base + 0x00	LCDC Display Mode Register (LCDC_DISPMOD), 0x302060
Base + 0x04	Main Window Display Start Address Register (LCDC_MAINADR), 0x302070
Base + 0x08	Main Screen Address Offset Register (LCDC_MAINOFS), 0x302074
Base + 0x0c	Sub-window Display Start Address Register (LCDC_SUBADR), 0x302080
Base + 0x10	Sub-screen Address Offset Register (LCDC_SUBOFS), 0x302084
Base + 0x14	Sub-window Start Position Register (LCDC_SUBSP), 0x302088
Base + 0x18	Sub-window End Position Register (LCDC_SUBEP), 0x30208c

Base: Reload table start address

The reload table can be located in IVRAM, and the start address (Base) can be specified using RTBL_BADR[31:10]/LCDC_RLDADR register. The low-order 10 bits of the LCDC_RLDADR register is fixed at 0x0, so the reload table always begins from a 1K-byte boundary address. Two or more reload tables can be prepared and switched by changing RTBL_BADR[31:10].

Note: Be sure not to locate the reload table in an external memory.

When a memory space is allocated to a reload table, the contents shown above must be programmed in the application program.

The reload table bit configuration is the same as that of the LCDC control registers.

Writing 1 to CTABRLD/LCDC_RLDCTL register resets the LCDC control registers with the reload table data. This reload operation should be performed during a vertical non-display period. CTABRLD retains 1 during reloading and it reverts to 0 when the reloading is completed.

LUT reload function

The LUT reload function is used to replace the look-up table settings. This function is effective when the look-up table function is enabled (LUTPASS/LCDC_DISPMOD register = 0).
The table below shows the contents of the reload table used in the LUT reload function.

Table 28.9.2 Reload Table Contents (LUT data)

Address	Monochrome LUT data (when COLOR = 0)
Base + 0x100	MLUT0[3:0]–MLUT7[3:0]
Base + 0x102	(same as the LCDC_MLUT0 register)
Base + 0x104	MLUT8[3:0]–MLUT15[3:0]
Base + 0x106	(same as the LCDC_MLUT1 register)

Base: Reload table start address

"Base" in the table refers the reload table start address specified by RTBL_BADR[31:10] (same as the control table reload function).
The monochrome LUT data configuration is the same as the LCDC_MLUT0 and LCDC_MLUT1 registers.
Writing 1 to LUTRLD/LCDC_RLDCTL register resets the look-up tables with the reload table data. This reload operation should be performed during a vertical non-display period. LUTRLD retains 1 during reloading and it reverts to 0 when the reloading is completed.
If LUTRLD and CTABRLD are both set to 1 at the same time, the LCDC replace the control register data first, then LUT data.

28.10 Control Register Details

Table 28.10.1 List of LCDC Registers

Address	Register name		Function
0x302000	LCDC_INT	LCDC Interrupt Enable Register	Enable/disable LCDC interrupts
0x302004	LCDC_PSAVE	Status and Power Save Configuration Register	Indicate LCDC status and set power save mode
0x302010	LCDC_HDISP	Horizontal Display Register	Set horizontal display period
0x302014	LCDC_VDISP	Vertical Display Register	Set vertical display period
0x302018	LCDC_MODR	MOD Rate Register	Set MOD rate
0x302020	LCDC_HDPS	Horizontal Display Start Position Register	Set horizontal display start position for TFT
0x302024	LCDC_VDPS	Vertical Display Start Position Register	Set vertical display start position for TFT
0x302028	LCDC_FPLINE	FPLINE Pulse Setting Register	Configure FPLINE pulse for TFT
0x30202c	LCDC_FPFR	FPFRAME Pulse Setting Register	Configure FPFRAME pulse for TFT
0x302030	LCDC_FPFROFS	FPFRAME Pulse Offset Register	Adjust FPLINE pulse timings for TFT
0x302040	LCDC_TFTSO	TFT Special Output Register	Set TFT control signals
0x302044	LCDC_TFT_CTL1	TFT_CTL1 Pulse Register	Set TFT_CTL1 pulse timings
0x302048	LCDC_TFT_CTL0	TFT_CTL0 Pulse Register	Set TFT_CTL0 pulse timings
0x30204c	LCDC_TFT_CTL2	TFT_CTL2 Register	Set TFT_CTL2 signal timing
0x302050	LCDC_RLDCTL	LCDC Reload Control Register	Control reloading
0x302054	LCDC_RLDADR	LCDC Reload Table Base Address Register	Set reload table base address
0x302060	LCDC_DISPMOD	LCDC Display Mode Register	Set display conditions
0x302070	LCDC_MAINADR	Main Window Display Start Address Register	Set main window display start address
0x302074	LCDC_MAINOFS	Main Screen Address Offset Register	Set virtual main screen width
0x302080	LCDC_SUBADR	Sub-window Display Start Address Register	Set sub-window display start address
0x302084	LCDC_SUBOFS	Sub-screen Address Offset Register	Set virtual sub-screen width
0x302088	LCDC_SUBSP	Sub-window Start Position Register	Set sub-window start position
0x30208c	LCDC_SUBEP	Sub-window End Position Register	Set sub-window end position
0x302090	LCDC_MLUT0	Monochrome Look-up Table Register 0	Monochrome look-up table data entries 0–7
0x302094	LCDC_MLUT1	Monochrome Look-up Table Register 1	Monochrome look-up table data entries 8–15

The LCDC registers are described in detail below.

Note: When data is written to the registers, the "Reserved" bits must always be written as 0 and not 1.

LCDC Interrupt Enable Register (LCDC_INT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
LCDC Interrupt Enable Register (LCDC_INT)	**0x302000** (32 bits)	D31–1	–	reserved	–	–	–	0 when being read.
		D0	**FRINTEN**	Frame interrupt enable	1 Enable 0 Disable	0	R/W	

D[31:1] Reserved

D0 FRINTEN: Frame Interrupt Enable Bit

Enables or disables LCDC frame interrupts.

1 (R/W): Enabled

0 (R/W): Disabled (default)

When using the frame interrupt, set FRINTEN to 1. The frame interrupt requests to the ITC is enabled. When this bit is set to 0, the frame interrupt will not be generated.

Status and Power Save Configuration Register (LCDC_PSAVE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Status and Power Save Configuration Register (LCDC_PSAVE)	0x302004 (32 bits)	D31	FRINTF	Frame interrupt flag	1 Occurred / 0 Not occurred	0	R/W	Reset by writing 1.
		D30–8	–	reserved	–	–	–	0 when being read.
		D7	VNDPF	Vertical display status flag	1 VNDP / 0 VDP	1	R	
		D6–2	–	reserved	–	–	–	0 when being read.
		D1–0	PSAVE[1:0]	Power save mode select	PSAVE[1:0] / Mode: 0x3 Normal; 0x2 reserved; 0x1 reserved; 0x0 Power save	0x0	R/W	

D31 FRINTF: Frame Interrupt Flag Bit

Indicates the frame 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): Ignored

FRINTF is set to 1 when a vertical non-display period begins. If FRINTEN/LCDC_INT register has been set to 1, a frame interrupt request is sent to the ITC at the same time.

D[30:8] Reserved

D7 VNDPF: Vertical Display Status Flag Bit

Indicates whether the LCD panel is in a vertical non-display period or not.

1 (R): Vertical non-display period (default)

0 (R): Vertical display period

VNDPF is set to 1 during a vertical non-display period, and set to 0 during a vertical display period. When images must be switched without causing the screen to flicker, it is possible to switch within a vertical non-display period by reading this bit.

D[6:2] Reserved

D[1:0] PSAVE[1:0]: Power Save Mode Select Bits

Selects the power-save mode.

Table 28.10.2 Power-Save Mode Settings

PSAVE[1:0]	Mode
0x3	Normal operation
0x2	Reserved
0x1	Reserved
0x0	Power-save mode

(Default: 0x0)

The LCD controller is placed in power-save mode by setting PSAVE[1:0] to 0x0. In this mode, all LCD signal output pins are dropped low and all operations of the LCD controller, other than accessing of its control registers and look-up tables are disabled. The LCD controller is taken out of power-save mode by setting PSAVE[1:0] to 0x3.

Horizontal Display Register (LCDC_HDISP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Horizontal Display Register (LCDC_HDISP)	0x302010 (32 bits)	D31–23	–	reserved	–	–	–	0 when being read.
		D22–16	**HTCNT[6:0]**	Horizontal total period (HT) setup HT = HDP + HNDP HT > HDPS + HDP (for TFT)	HT = (HTCNT + 1) × 8 [Ts] HNDP = (HTCNT - HDPCNT) × 8 [Ts]	0x0	R/W	
		D15–7	–	reserved	–	–	–	0 when being read.
		D6–0	**HDPCNT [6:0]**	Horizontal display period (HDP) setup	HDP = (HDPCNT + 1) × 8 [Ts]	0x0	R/W	

D[31:23] Reserved

D[22:16] HTCNT[6:0]: Horizontal Total Period (HT) Setup Bits

Sets the horizontal total period (HT) in 8-pixel increments. (Default: 0x0)

HT = (HTCNT[6:0] + 1) × 8 [Ts] (Ts: pixel clock period)

The horizontal total period contains horizontal display period and horizontal non-display period and the maximum value that can be set is 1,024-pixel period.

The following conditions must be satisfied when setting HTCNT[6:0]:

HTCNT[6:0] ≥ HDPCNT[6:0] + 3

HT > HDP + HDPS

Note: HT should be determined so that the horizontal non-display period (HNDP = HT - HDP) will be longer than the time required when the LCDC accesses eight words in the VRAM.

D[15:7] Reserved

D[6:0] HDPCNT[6:0]: Horizontal Display Period (HDP) Setup Bits

Sets the horizontal display period (HDP, panel horizontal resolution) in 8-pixel increments. (Default: 0x0)

HDP = (HDPCNT[6:0] + 1) × 8 [Ts]

The following condition must be satisfied when setting HDPCNT[6:0]:

HDP ≥ 16 (HDPCNT[6:0] ≥ 1)

∗ HDP means the panel horizontal resolution that is not related to the LCD panel type, panel data width, and bpp display mode.

Example: when 320 × 240 LCD (STN/TFT) panel is used

HDP = 320

HDPCNT[6:0] = 320/8 - 1 = 39 (= 0x27)

Vertical Display Register (LCDC_VDISP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Vertical Display Register (LCDC_VDISP)	0x302014 (32 bits)	D31–26	–	reserved	–	–	–	0 when being read.
		D25–16	**VTCNT[9:0]**	Vertical total period (VT) setup VT = VDP + VNDP VT > VDPS + VDP (for TFT)	VT = VTCNT + 1 [lines] VNDP = VTCNT - VDPCNT [lines]	0x0	R/W	
		D15–10	–	reserved	–	–	–	0 when being read.
		D9–0	**VDPCNT [9:0]**	Vertical display period (VDP) setup	VDP = VDPCNT + 1 [lines]	0x0	R/W	

D[31:26] Reserved

D[25:16] VTCNT[9:0]: Vertical Total Period (VT) Setup Bits

Sets the vertical total period (VT) in line units. (Default: 0x0)

VT = VTCNT[9:0] + 1 [lines]

The vertical total period contains vertical display period and vertical non-display period and the maximum value that can be set is 1,024 lines.

The following condition must be satisfied when setting VTCNT[9:0]:

VT > VDP + VDPS

D[15:10] Reserved

D[9:0] VDPCNT[9:0]: Vertical Display Period (VDP) Setup Bits

Sets the vertical display period (VDP, panel vertical resolution) in line units. (Default: 0x0)

VDP = VDPCNT[9:0] + 1 [lines]

The following condition must be satisfied when setting VDPCNT[9:0]:
VT ≥ VDP + 1

* VDP means the panel vertical resolution that is not related to the LCD panel type, panel data width, and bpp display mode.
Example: when 320 × 240 LCD (STN/TFT) panel is used
VDP = 240
VDPCNT[9:0] = 240 - 1 = 239 (= 0xef)

MOD Rate Register (LCDC_MODR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
MOD Rate Register (LCDC_MODR)	**0x302018** (32 bits)	D31–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**MOD[5:0]**	LCD MOD rate setup	0x0 to 0x3f	0x0	R/W	

D[31:6] Reserved

D[5:0] MOD[5:0]: LCD MOD Rate Setup Bits

Sets the cycle time at which to switch the MOD signal. (Default: 0x0)
When this register is 0x0, the MOD signal switches at the cycle time of the FPFRAME signal. If another period is desired, set the FPLINE pulse-count value.

Horizontal Display Start Position Register (LCDC_HDPS)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Horizontal Display Start Position Register (LCDC_HDPS)	**0x302020** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**HDPSCNT [9:0]**	Horizontal display period start position for TFT HT > HDP + HDPS	HDPS = HDPSCNT [Ts]	0x0	R/W	0x0 must be set for STN panels.

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:10] Reserved

D[9:0] HDPSCNT[9:0]: Horizontal Display Period Start Position Bits

Sets the horizontal display period start position (HDPS) for TFT panels in pixel clock units. (Default: 0x0)

HDPS = HDPSCNT[9:0] + 1 [Ts] (Ts: pixel clock period)

The following condition must be satisfied when setting HDPSCNT[9:0]:
HT > HDP + HDPS

Vertical Display Start Position Register (LCDC_VDPS)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Vertical Display Start Position Register (LCDC_VDPS)	**0x302024** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**VDPSCNT [9:0]**	Vertical display period start position for TFT VT > VDP + VDPS	VDPS = VDPSCNT [lines]	0x0	R/W	0x0 must be set for STN panels.

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:10] Reserved

D[9:0] VDPSCNT[9:0]: Vertical Display Period Start Position Bits

Sets the vertical display period start position (VDPS) for TFT panels in line units. (Default: 0x0)

VDPS = VDPSCNT[9:0] [lines]

The following condition must be satisfied when setting VDPSCNT[9:0]:
VT > VDP + VDPS

FPLINE Pulse Setting Register (LCDC_FPLINE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPLINE Pulse Setting Register (LCDC_ FPLINE)	**0x302028** (32 bits)	D31–26	–	reserved	–	–	–	0 when being read.
		D25–16	**FPLINE_ ST[9:0]**	FPLINE pulse start position setup	Start position = FPLINE_ST + 1 [Ts]	0x0	R/W	∗1: For TFT 0x0 must be set for STN panels.
		D15–8	–	reserved	–	–	–	0 when being read.
		D7	**FPLINE_ POL**	FPLINE pulse polarity setup	1 Active high 0 Active low	0	R/W	(∗1)
		D6–0	**FPLINE_ WD[6:0]**	FPLINE pulse width setup	Pulse width = FPLINE_WD + 1 [Ts]	0x0	R/W	

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:26] Reserved

D[25:16] FPLINE_ST[9:0]: FPLINE Pulse Start Position Setup Bits

Sets the horizontal sync pulse (FPLINE or LP) start position (HPS) for TFT panels in pixel clock units. (Default: 0x0)

HPS = FPLINE_ST[9:0] + 1 [Ts] (Ts: pixel clock period)

D[15:8] Reserved

D7 FPLINE_POL: FPLINE Pulse Polarity Setup Bit

Sets the horizontal sync pulse polarity for TFT panels.
1 (R/W): Active high
0 (R/W): Active low (default)

D[6:0] FPLINE_WD[6:0]: FPLINE Pulse Width Setup Bits

Sets the horizontal sync pulse width (HPW) for TFT panels in pixel clock units. (Default: 0x0)

HPW = FPLINE_WD[6:0] + 1 [Ts] (Ts: pixel clock period)

FPFRAME Pulse Setting Register (LCDC_FPFR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPFRAME Pulse Setting Register (LCDC_FPFR)	**0x30202c** (32 bits)	D31–26	–	reserved	–	–	–	0 when being read.
		D25–16	**FPFRAME_ ST[9:0]**	FPFRAME pulse start position setup	Start position = FPFRAME_ST × HT [Ts]	0x0	R/W	∗1: For TFT 0x0 must be set for STN panels.
		D15–8	–	reserved	–	–	–	0 when being read.
		D7	**FPFRAME_ POL**	FPFRAME pulse polarity setup	1 Active high 0 Active low	0	R/W	(∗1)
		D6–0	**FPFRAME_ WD[6:0]**	FPFRAME pulse width setup	Pulse width = (FPFRAME_WD+1) × HT [Ts]	0x0	R/W	(∗1)

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:26] Reserved

D[25:16] FPFRAME_ST[9:0]: FPFRAME Pulse Start Position Setup Bits

Sets the vertical sync pulse (FPFRAME or SPS) start position (VPS) for TFT panels. (Default: 0x0)

VPS = FPFRAME_ST[9:0] [lines] = FPFRAME_ST[9:0] × HT [Ts] (Ts: pixel clock period)

D[15:8] Reserved

D7 FPFRAME_POL: FPFRAME Pulse Polarity Setup Bit

Sets the vertical sync pulse polarity for TFT panels.
1 (R/W): Active high
0 (R/W): Active low (default)

D[6:0] FPFRAME_WD[6:0]: FPFRAME Pulse Width Setup Bits

Sets the vertical sync pulse width (VPW) for TFT panels. (Default: 0x0)

VPW = FPFRAME_WD[6:0] + 1 [lines] = (FPFRAME_WD[6:0] + 1) × HT [Ts]
(Ts: pixel clock period)

FPFRAME Pulse Offset Register (LCDC_FPFROFS)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPFRAME Pulse Offset Register (LCDC_FPFROFS)	0x302030 (32 bits)	D31–26	–	reserved	–	–	–	0 when being read.
		D25–16	FPFRAME_STPOFS[9:0]	FPFRAME pulse stop offset	Stop offset = FPFRAME_STPOFS [Ts]	0x0	R/W	∗1: For TFT 0x0 must be set for STN panels.
		D15–10	–	reserved	–	–	–	0 when being read.
		D9–0	FPFRAME_STOFS[9:0]	FPFRAME pulse start offset	Start offset = FPFRAME_STOFS [Ts]	0x0	R/W	(∗1)

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:26] Reserved

D[25:16] FPFRAME_STPOFS[9:0]: FPFRAME Pulse Stop Offset Bits

Adjusts the vertical sync pulse end position (pulse width), which has been set in line units, in pixel clock units. (Default: 0x0)

VPW' = (FPFRAME_WD[6:0] + 1) × HT - FPFRAME_STOFS[9:0]
+ FPFRAME_STPOFS[9:0] [Ts] (Ts: pixel clock period)

D[15:10] Reserved

D[9:0] FPFRAME_STOFS[9:0]: FPFRAME Pulse Start Offset Bits

Adjusts the vertical sync pulse start position, which has been set in line units, in pixel clock units. (Default: 0x0)

VPS' = FPFRAME_ST[9:0] × HT + FPFRAME_STOFS[9:0] [Ts] (Ts: pixel clock period)

TFT Special Output Register (LCDC_TFTSO)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
TFT Special Output Register (LCDC_TFTSO)	0x302040 (32 bits)	D31–4	–	reserved	–				–	–	0 when being read.
		D3	CTL1CTL	TFT_CTL1 control	1	Program	0	Toggle/line	0	R/W	For TFT 0x0 must be set for STN panels.
		D2	CTLCNT_RUN	TFT_CTL0–2 control counter run/stop	1	Run	0	Stop	0	R/W	
		D1	FPSHIFT_POL	FPSHIFT polarity	1	Falling	0	Rising	0	R/W	
		D0	CTL01SWAP	TFT_CTL0/TFT_CTL1 swap	1	Swap	0	Not swap	0	R/W	

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:4] Reserved

D3 CTL1CTL: TFT_CTL1 Control Bit

Selects the behavior of the TFT_CTL1 (CLS) signal.
1 (R/W): Toggle at the programmed timing
0 (R/W): Toggle every line (default)

Set CTL1CTL to 1 when using the TFT_CTL1 (CLS) signal that has been programmed using the LCDC_TFT_CTL1 register or preset with standard conditions. CTL1CTL is set to 0 by default, in this case the TFT_CTL1 (CLS) signal toggles between high and low every time the FPLINE (LP) pulse is output.

D2 CTLCNT_RUN: TFT_CTL0–2 Control Counter Run/Stop Bit

Starts and stops the TFT_CTL0–2 control counters.
1 (R/W): Run
0 (R/W): Stop (default)

The LCDC has a built-in counters to control the TFT_CTL0–2 signal output timings. Setting CTLCNT_RUN to 1 starts the counters enabling the application to program the TFT_CTL0 (PS), TFT_CTL1 (CLS), and TFT_CTL2 (REV) signals. Be sure to set this bit to 1 when using the TFT_CTL0–2 signals.
When CTLCNT_RUN = 0, the counters stop counting. When the TFT_CTL0–2 signals are not used, please set this bit to 0 to reduce power consumption.

D1 FPSHIFT_POL: FPSHIFT Polarity Bit

Selects the polarity of the FPSHIFT (DCLK) signal for TFT panels.
1 (R/W): Falling edge
0 (R/W): Rising edge (default)

When FPSHIFT_POL is set to 1, the FPDAT[23:0] output signal toggles at the rising edge (sampled at the falling edge) of the FPSHIFT (DCLK) signal. When FPSHIFT_POL is set to 0, the FPDAT[23:0] output signal toggles at the falling edge (sampled at the rising edge) of the FPSHIFT (DCLK) signal.

D0 CTL01SWAP: TFT_CTL0/TFT_CTL1 Swap Bit

Swaps the signal between TFT_CTL1 and TFT_CTL0.
1 (R/W): Swapped (TFT_CTL0 = CLS, TFT_CTL1 = PS)
0 (R/W): Not swapped (TFT_CTL0 = PS, TFT_CTL1 = CLS) (default)

TFT_CTL1 Pulse Register (LCDC_TFT_CTL1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
TFT_CTL1 Pulse Register (LCDC_TFT_CTL1)	**0x302044** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**CTL1STP [9:0]**	TFT_CTL1 pulse stop offset TFT_CTL1 pulse width = (CTL1STP - CTL1ST +1) Ts	Stop offset = CTL1STP + 1 [Ts]	0x0	R/W	∗2: For TFT This register is enabled when CTLCNT_RUN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**CTL1ST [9:0]**	TFT_CTL1 pulse start offset	Start offset = CTL1ST [Ts]	0x0	R/W	(∗2)

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:26] Reserved

D[25:16] CTL1STP[9:0]: TFT_CTL1 Pulse Stop Offset Bits

Specifies the TFT_CTL1 (CLS) pulse end position with an offset value (in pixel clock units) from the FPLINE pulse start position. (Default: 0x0)

D[15:10] Reserved

D[9:0] CTL1ST[9:0]: TFT_CTL1 Pulse Start Offset Bits

Specifies the TFT_CTL1 (CLS) pulse start position with an offset value (in pixel clock units) from the FPLINE pulse start position. (Default: 0x0)

Setting this register configures the TFT_CTL1 pulse width to "CTL1STP[9:0] - CTL1ST[9:0] + 1 [Ts]." To enable this register, set CTL1CTL/LCDC_TFTSO register and CTLCNT_RUN/LCDC_TFTSO register to 1.

TFT_CTL0 Pulse Register (LCDC_TFT_CTL0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
TFT_CTL0 Pulse Register (LCDC_TFT_CTL0)	0x302048 (32 bits)	D31–26	–	reserved	–	–	–	0 when being read.
		D25–16	CTL0STP [9:0]	TFT_CTL0 pulse stop offset TFT_CTL0 pulse width = (CTL0STP - CTL0ST +1) Ts	Stop offset = CTL0STP + 1 [Ts]	0x0	R/W	∗2: For TFT This register is enabled when CTLCNT_RUN = 1.
		D15–10	–	reserved	–	–	–	0 when being read.
		D9–0	CTL0ST [9:0]	TFT_CTL0 pulse start offset	Start offset = CTL0ST [Ts]	0x0	R/W	(∗2)

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:26] Reserved

D[25:16] CTL0STP[9:0]: TFT_CTL0 Pulse Stop Offset Bits

Specifies the TFT_CTL0 (PS) pulse end position with an offset value (in pixel clock units) from the FPLINE pulse start position. (Default: 0x0)

D[15:10] Reserved

D[9:0] CTL0ST[9:0]: TFT_CTL0 Pulse Start Offset Bits

Specifies the TFT_CTL0 (PS) pulse start position with an offset value (in pixel clock units) from the FPLINE pulse start position. (Default: 0x0)

Setting this register configures the TFT_CTL0 pulse width to "CTL0STP[9:0] - CTL0ST[9:0] + 1 [Ts]." To enable this register, set CTLCNT_RUN/LCDC_TFTSO register to 1.

TFT_CTL2 Register (LCDC_TFT_CTL2)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
TFT_CTL2 Register (LCDC_TFT_CTL2)	0x30204c (32 bits)	D31–10	–	reserved	–	–	–	0 when being read.
		D9–0	CTL2DLY [9:0]	TFT_CTL2 delay setup	Delay = CTL2DLY [Ts]	0x0	R/W	∗2: For TFT This register is enabled when CTLCNT_RUN = 1.

Note: This register is used only for setting TFT panel parameters. When using an STN panel, leave this register unaltered as 0x0.

D[31:10] Reserved

D[9:0] CTL2DLY[9:0]: TFT_CTL2 Delay Setup Bits

Sets the delay time (in pixel clock units) from the FPLINE pulse start position until the TFT_CTL2 signal toggles. (Default: 0x0)

To enable this register, set CTLCNT_RUN/LCDC_TFTSO register to 1.

LCDC Reload Control Register (LCDC_RLDCTL)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
LCDC Reload Control Register (LCDC_RLDCTL)	0x302050 (32 bits)	D31–2	–	reserved	–	–	–	0 when being read.
		D1	LUTRLD	LUT reload trigger	1 Trigger / 0 Ignored	0	W	
					1 Reloading / 0 Finished		R	
		D0	CTABRLD	Control table reload trigger	1 Trigger / 0 Ignored	0	W	
					1 Reloading / 0 Finished		R	

D[31:2] Reserved

D1 LUTRLD: LUT Reload Trigger Bit

Replaces the look-up table values by the reload table data.

1 (W): Trigger to reload
0 (W): Ignored
1 (R): Reloading
0 (R): Reloading has finished (default)

Writing 1 to LUTRLD resets the look-up table entries with the reload table data. This reload operation should be performed during a vertical non-display period. LUTRLD retains 1 during reloading and it reverts to 0 when the reloading is completed.
The LUT reload function is effective when the look-up table function is enabled (LUTPASS/LCDC_DISPMOD = 0).
If LUTRLD and CTABRLD are both set to 1 at the same time, the LCDC replace the control register data first, then LUT data.
See Section 28.9 for the reload table contents.

D0 CTABRLD: Control Table Reload Trigger Bit

Replaces the LCDC control registers by the reload table data.

1 (W): Trigger to reload
0 (W): Ignored
1 (R): Reloading
0 (R): Reloading has finished (default)

Writing 1 to CTABRLD resets the control registers with the reload table data. This reload operation should be performed during a vertical non-display period. CTABRLD retains 1 during reloading and it reverts to 0 when the reloading is completed.
See Section 28.9 for the reload table contents.

LCDC Reload Table Base Address Register (LCDC_RLDADR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
LCDC Reload Table Base Address Register (LCDC_RLDADR)	**0x302054** (32 bits)	D31–10	**RTBL_BADR[31:10]**	Reload table base address (1KB boundary address, A[9:0] = 0x0)	Area 3*	0x0	R/W	* DSTRAM cannot be used.
		D9–0	–	reserved	–	–	–	0 when being read.

D[31:10] RTBL_BADR[31:10]: Reload Table Base Address Bits

Specifies the reload table base address. (Default: 0x0)
A 1K-byte boundary address in IVRAM must be specified.

Notes: • Be sure not to locate the reload table in an external memory.

• DSTRAM in Area 3 cannot be used to locate the reload table. For more information, see “Bus Masters and Accessible Memories” in the “Memory Map” chapter.

D[9:0] Reserved

LCDC Display Mode Register (LCDC_DISPMOD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
LCDC Display Mode Register (LCDC_ DISPMOD)	0x302060 (32 bits)	D31	PANELSEL	Panel type select	1 TFT / 0 STN	0	R/W	
		D30	COLOR	Color/mono select	1 Color / 0 Mono	0	R/W	
		D29	FPSHIFT_ MSK	FPSHIFT mask enable	1 Enable / 0 Disable	0	R/W	
		D28	–	reserved	–	–	–	0 when being read.
		D27–26	DWD[1:0]	LCD panel data width select	DWD[1:0] / Data width: 0x3 8 bits (fmt2); 0x2 reserved; 0x1 8 bits (fmt1); 0x0 4 bits	0x0	R/W	
		D25	SWINV	Software video invert	1 Invert / 0 Normal	0	R/W	
		D24	BLANK	Display blank enable	1 Blank / 0 Normal	0	R/W	
		D23–8	–	reserved	–	–	–	0 when being read.
		D7	FRMRPT	Frame repeat for EL panel	1 Repeat / 0 Not repeat	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	LUTPASS	LUT bypass mode select	1 Bypass / 0 Use	1	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	BPP[2:0]	Bit-per-pixel select	BPP[2:0] / bpp: 0x7 reserved; 0x6 24 bpp; 0x5 16 bpp; 0x4 12 bpp; 0x3 8 bpp; 0x2 4 bpp; 0x1 2 bpp; 0x0 1 bpp	0x0	R/W	

D31 PANELSEL: Panel Type Select Bit

Selects the type of connected LCD panel (STN or TFT).

1 (R/W): TFT panel

0 (R/W): STN panel (default)

When TFT panel is selected, COLOR and DWD[1:0] settings are disabled.

D30 COLOR: Color/Mono Select Bit

Selects the type of connected LCD panel (color or monochrome).

1 (R/W): Color panel

0 (R/W): Monochrome panel (default)

D29 FPSHIFT_MSK: FPSHIFT Mask Enable Bit

Enables the FPSHIFT mask (effective only for STN monochrome LCD panels and TFT panels).

1 (R/W): Enabled

0 (R/W): Disabled (default)

When FPSHIFT_MSK is set to 1, the FPSHIFT signal is masked and is not output during the non-display period. When FPSHIFT_MSK is set to 0, the FPSHIFT signal is output even during the non-display period. This setting is effective only for STN monochrome LCD panels (COLOR = 0) and TFT panels. When an STN color LCD panel is used, the FPSHIFT signal is always masked regardless of the setting of this bit.

D28 Reserved

D[27:26] DWD[1:0]: LCD Panel Data Width Select Bits

Selects the STN LCD panel's data width and format.

Table 28.10.3 Data Width Selection of STN Panels

COLOR	DWD[1:0]	Data width
1	0x3	Color single 8-bit passive LCD panel (format 2)
	0x2	Reserved
	0x1	Color single 8-bit passive LCD panel (format 1)
	0x0	Color single 4-bit passive LCD panel
0	0x3	Monochrome single 8-bit passive LCD panel
	0x2	Reserved
	0x1	Monochrome single 8-bit passive LCD panel
	0x0	Monochrome single 4-bit passive LCD panel

D25 SWINV: Software Video Invert Bit

Inverts the display.
1 (R/W): Invert
0 (R/W): Normal display (default)

When SWINV is set to 1, the display on the LCD panel is inverted (displayed in inverse video). When SWINV is set to 0, normal display is maintained. Inverse operation is applied to the display data output from the LCDC, and does not affect the display memory.

D24 BLANK: Display Blank Enable Bit

Clears the display (entire screen turned blank).
1 (R/W): Blank
0 (R/W): Normal display (default)

When BLANK is set to 0, data in the display memory is displayed on the LCD panel. When BLANK is set to 1, all FPDAT signals are dropped low (when SWINV = 0) or high (when SWINV = 1) to clear the display. This setting does not affect the display memory.
This function is effective for both STN and TFT panels.

D[23:8] Reserved

D7 FRMRPT: Frame Repeat for EL Panel Bit

Selects whether to repeat the frame-rate modulation pattern (effective only for EL panels).
1 (R/W): Repeated
0 (R/W): Not repeated (default)

When FRMRPT is set to 1, the internal 19-bit frame counter is enabled and starts counting the number of frames. Each time this counter overflows (0x40000 → 0), the frame-rate modulation pattern is repeated. When FRMRPT is set to 0, the counter is disabled and the frame-rate modulation pattern is not repeated.

D[6:5] Reserved

D4 LUTPASS: LUT Bypass Mode Select Bit

Selects whether the look-up table is bypassed.
1 (R/W): Bypassed (default)
0 (R/W): Used

When LUTPASS is set to 1, the look-up table is bypassed and the pixel data in the display memory represents the display data to be sent to the LCD panel.
When LUTPASS is set to 0, the look-up table is used to convert pixel data in the display memory into LCD interface data.
In color mode, this bit must be set to 1 as the look-up table cannot be used.

D3 Reserved

D[2:0] BPP[2:0]: Bit-Per-Pixel Select Bits

Selects the bpp mode.

Table 28.10.4 Bpp Mode Settings

BPP[2:0]	bpp mode
0x7	Reserved
0x6	24 bpp
0x5	16 bpp
0x4	12 bpp
0x3	8 bpp
0x2	4 bpp
0x1	2 bpp
0x0	1 bpp

(Default: 0x0)

Main Window Display Start Address Register (LCDC_MAINADR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Main Window Display Start Address Register (LCDC_ MAINADR)	**0x302070** (32 bits)	D31–0	**MW_START [31:0]**	Main window start address MW_START31 = MSB MW_START0 = LSB	0x0 to 0xffffffff (Areas 3–5, 7–10, 13–16, and 19–22)	0x0	R/W	

D[31:0] MW_START[31:0]: Main Window Start Address Bits

Sets the main window display start address. (Default: 0x0)

Note that a word boundary address (A[1:0] = 0b00) in the IVRAM or external VRAM must be specified to this register.

Main Screen Address Offset Register (LCDC_MAINOFS)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Main Screen Address Offset Register (LCDC_ MAINOFS)	**0x302074** (32 bits)	D31–12	**–**	reserved	–	–	–	0 when being read.
		D11–0	**MW_OFS [11:0]**	Main screen address offset	Main screen width (pixels) × bpp/32	0x0	R/W	

D[31:12] Reserved

D[11:0] MW_OFS[11:0]: Main Screen Address Offset Bits

Sets the main virtual screen width in words. (Default: 0x0)

The set value is calculated as follows:

MW_OFS[11:0] = virtual screen width in pixels × bpp / 32

See "Main screen address offset for virtual screen" in Section 28.6.2 for more information on the virtual screen and the configurations.

Sub-window Display Start Address Register (LCDC_SUBADR)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Sub-window Display Start Address Register (LCDC_ SUBADR)	**0x302080** (32 bits)	D31–0	**SW_START [31:0]**	Sub-window start address SW_START31 = MSB SW_START0 = LSB	0x0 to 0xffffffff (Areas 3–5, 7–10, 13–16, and 19–22)	0x0	R/W	

D[31:0] SW_START[31:0]: Sub-Window Start Address Bits

Sets the sub-window display start address. (Default: 0x0)

Note that a word boundary address (A[1:0] = 0b00) in the IVRAM or external VRAM must be specified to this register.

Sub-Screen Address Offset Register (LCDC_SUBOFS)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Sub-screen Address Offset Register (LCDC_ SUBOFS)	**0x302084** (32 bits)	D31–12	**–**	reserved	–	–	–	0 when being read.
		D11–0	**SW_OFS [11:0]**	Sub-screen address offset	Sub-screen width (pixels) × bpp/32	0x0	R/W	

D[31:12] Reserved

D[11:0] SW_OFS[11:0]: Sub-Screen Address Offset Bits

Sets the sub-virtual screen width in words. (Default: 0x0)

The set value is calculated as follows:

SW_OFS[11:0] = virtual screen width in pixels × bpp / 32

See "Main screen address offset for virtual screen" in Section 28.6.2 for more information on the virtual screen and the configurations.

Sub-Window Start Position Register (LCDC_SUBSP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Sub-window Start Position Register (LCDC_SUBSP)	**0x302088** (32 bits)	D31	**PIP_EN**	PIP enable	1 Enable 0 Disable	0	R/W	
		D30–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**PIP_ YSTART [9:0]**	Sub-window vertical (Y) start position	Y start position = PIP_YSTART (lines) from the origin	0x0	R/W	∗3: This register is enabled when PIP_EN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**PIP_ XSTART [9:0]**	Sub-window horizontal (X) start position	X start position = PIP_XSTART (pixels) from the origin (word units)	0x0	R/W	(∗3)

D31 PIP_EN: PIP Enable Bit

Enables the Picture-in-Picture Plus function to display the sub-window in the main window.
1 (R/W): Enabled
0 (R/W): Disabled (default)

Configure the sub-window using the registers at 0x302080 to 0x30208c before setting PIP_EN to 1.

D[30:26] Reserved

D[25:16] PIP_YSTART[9:0]: Sub-Window Vertical (Y) Start Position Bits

Sets the sub-window vertical display start position. (Default: 0x0)
Specify the number of lines from the LCD panel origin point to the upper left corner of the sub-window in 1-line increments.

PIP_YSTART[9:0] = YSTART [lines]

For example, to specify the sub-window vertical start position as 60 lines, set PIP_YSTART[9:0] to 60.

Note: The sub-window cannot locate at the first line of main window.
The sub-window vertical (Y) start position (lines) cannot be set to the first line of main window. Do not set 0x0 to PIP_YSTART[9:0].

D[15:10] Reserved

D[9:0] PIP_XSTART[9:0]: Sub-Window Horizontal (X) Start Position Bits

Sets the sub-window horizontal display start position. (Default: 0x0)
Convert the number of pixels from the LCD panel origin point to the upper left corner of the sub-window into the number of data words according to the bpp mode and set it to these bits.

PIP_XSTART[9:0] = XSTART pixels × bpp ÷ 32 [words]

It can be specified in (32 bits ÷ bpp) pixel increments.
1-bpp mode: 1-word = 32-pixel units
2-bpp mode: 1-word = 16-pixel units
4-bpp mode: 1-word = 8-pixel units
8-bpp mode: 1-word = 4-pixel units
12-bpp mode: 3-word = 8-pixel units (because the value must be an integer)
16-bpp mode: 1-word = 2-pixel units
24-bpp mode: 3-word = 4-pixel units (because the value must be an integer)

For example, to specify the sub-window horizontal start position as 80 pixels in 8-bpp mode, set PIP_ XSTART[9:0] to 20.

Sub-Window End Position Register (LCDC_SUBEP)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Sub-window End Position Register (LCDC_SUBEP)	**0x30208c** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**PIP_YEND [9:0]**	Sub-window vertical (Y) end position	Y end position = PIP_YEND (lines) from the origin	0x0	R/W	∗3: This register is enabled when PIP_EN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**PIP_XEND [9:0]**	Sub-window horizontal (X) end position	X end position = PIP_XEND (pixels) from the origin (word units)	0x0	R/W	(∗3)

D[31:26] Reserved

D[25:16] PIP_YEND[9:0]: Sub-Window Vertical (Y) End Position Bits

Sets the sub-window vertical display end position. (Default: 0x0)
Specify the number of lines from the LCD panel origin point to the lower right corner of the sub-window in 1-line increments.

PIP_YEND[9:0] = YEND - 1 [lines]

Note: The sub-window cannot locate at the first line of main window.
The sub-window vertical (Y) end position (lines) cannot be set to the first line of main window. Do not set 0x0 to PIP_YEND[9:0].

D[15:10] Reserved

D[9:0] PIP_XEND[9:0]: Sub-Window Horizontal (X) End Position Bits

Sets the sub-window horizontal display end position. (Default: 0x0)
Convert the number of pixels from the LCD panel origin point to the lower right corner of the sub-window into the number of data words according to the bpp mode and set it to these bits.

PIP_XEND[9:0] = XEND pixels × bpp ÷ 32 - 1 [words]

Monochrome Look-up Table Registers 0 and 1 (LCDC_MLUT0/1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Monochrome Look-up Table Register 0 (LCDC_MLUT0)	**0x302090** (32 bits)	D31–28	**MLUT7[3:0]**	Monochrome LUT entry 7 data	0x0 to 0xf	0x0	R/W	
		D27–24	**MLUT6[3:0]**	Monochrome LUT entry 6 data	0x0 to 0xf	0x0	R/W	
		D23–20	**MLUT5[3:0]**	Monochrome LUT entry 5 data	0x0 to 0xf	0x0	R/W	
		D19–16	**MLUT4[3:0]**	Monochrome LUT entry 4 data	0x0 to 0xf	0x0	R/W	
		D15–12	**MLUT3[3:0]**	Monochrome LUT entry 3 data	0x0 to 0xf	0x0	R/W	
		D11–8	**MLUT2[3:0]**	Monochrome LUT entry 2 data	0x0 to 0xf	0x0	R/W	
		D7–4	**MLUT1[3:0]**	Monochrome LUT entry 1 data	0x0 to 0xf	0x0	R/W	
		D3–0	**MLUT0[3:0]**	Monochrome LUT entry 0 data	0x0 to 0xf	0x0	R/W	
Monochrome Look-up Table Register 1 (LCDC_MLUT1)	**0x302094** (32 bits)	D31–28	**MLUT15[3:0]**	Monochrome LUT entry 15 data	0x0 to 0xf	0x0	R/W	
		D27–24	**MLUT14[3:0]**	Monochrome LUT entry 14 data	0x0 to 0xf	0x0	R/W	
		D23–20	**MLUT13[3:0]**	Monochrome LUT entry 13 data	0x0 to 0xf	0x0	R/W	
		D19–16	**MLUT12[3:0]**	Monochrome LUT entry 12 data	0x0 to 0xf	0x0	R/W	
		D15–12	**MLUT11[3:0]**	Monochrome LUT entry 11 data	0x0 to 0xf	0x0	R/W	
		D11–8	**MLUT10[3:0]**	Monochrome LUT entry 10 data	0x0 to 0xf	0x0	R/W	
		D7–4	**MLUT9[3:0]**	Monochrome LUT entry 9 data	0x0 to 0xf	0x0	R/W	
		D3–0	**MLUT8[3:0]**	Monochrome LUT entry 8 data	0x0 to 0xf	0x0	R/W	

These registers are used to set data to the monochrome look-up table entries. The entries used depend on the bpp mode.
1-bpp mode: Entries 0 and 1 are used.
2-bpp mode: Entries 0 to 3 are used.
4-bpp mode: All entries are used.

D[31:28], D[27:24], D[23:20], D[19:16], D[15:12], D[11:8], D[7:4], D[3:0]
MLUT*n*[3:0]: Monochrome LUT Entry *n* Data Bits

Sets the 4-bit data for the monochrome look-up table entry *n*. (Default: 0x0)

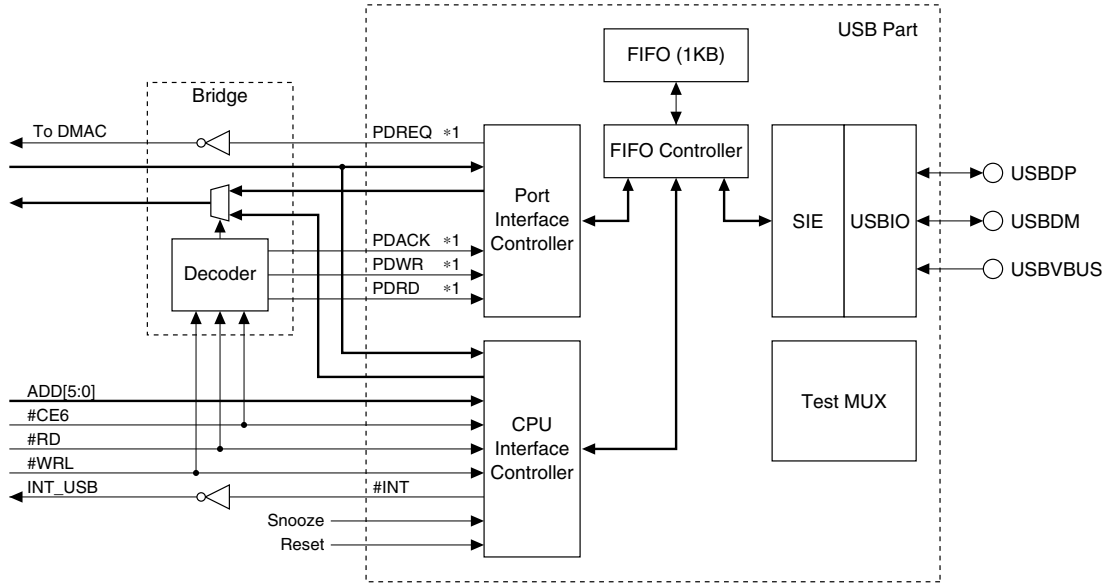
29 USB Function Controller (USB)

29.1 USB Function Controller Overview

The S1C33L27 includes a USB function controller that supports 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.
- Equipped with a general-purpose DMA port.
- 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 29.1.1 shows the block diagram of the USB function controller.



*1 The PDREQ, PDACK, PDWR and PDRD signals level must be configured as "Active High".

Figure 29.1.1 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.

29.2 Pins for the USB Interface

Table 29.2.1 list the pins used for the USB interface.

Table 29.2.1 USB Interface Pins

Pin name	I/O	Qty	Function
USBDP	I/O	1	USB D+ pin
USBDM	I/O	1	USB D- pin
USBVBUS	I	1	USB VBUS pin (Allows input of 5 V.)

29.3 USB Operating Clocks

The USB function controller operates with USB_CLK supplied from the CMU.
USBREG_CLK is also used for accessing the USB control registers.
By default, these two clocks are not supplied to the USB function controller. Therefore the clock supply must be enabled before running the USB function controller. For more information on clock control, see the "Clock Management Unit (CMU)" chapter.

29.4 Settings in Other Modules

Before using the USB function controller, the following modules/register must be programmed.

Clock management unit (CMU):
- Control the clock supply to the USB function controller.

Misc register (MISC):
- Set the number of wait cycles to be inserted when the USB control register is accessed.
- Enable or disable the snooze control by the USB function controller.
- Enable or disable the USB interrupts.

Interrupt controller (ITC):
- Set the USB interrupt level.

DMA controller (DMAC):
- Program the control table for the DMAC channel to be triggered from the USB function controller.

For more information, see the respective chapters.

Note: The DMA transfer address for the USB function controller must be located in Area 6 (0x300d00 to 0x300dff, 256 bytes). Make sure that the transfer address does not exceed the address range from 0x300d00 to 0x300dff by the address increment operation during DMA transfer.

29.5 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"

29.5.1 USB Control

Endpoints

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 29.5.1.1 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 29.5.1.1 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 29.5.1.2 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 29.5.1.2 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 29.5.1.3 lists control items and statuses related to transaction control on the EP0 endpoint.

Table 29.5.1.3 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 29.5.1.4 lists control items and statuses related to transaction processing on the EPa, EPb, EPc, and EPd endpoints.

Table 29.5.1.4 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 29.5.1.1 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.

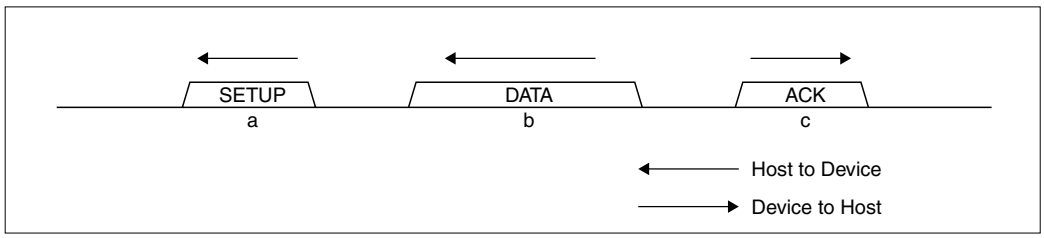


Figure 29.5.1.1 SETUP Transaction

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 29.5.1.2 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.

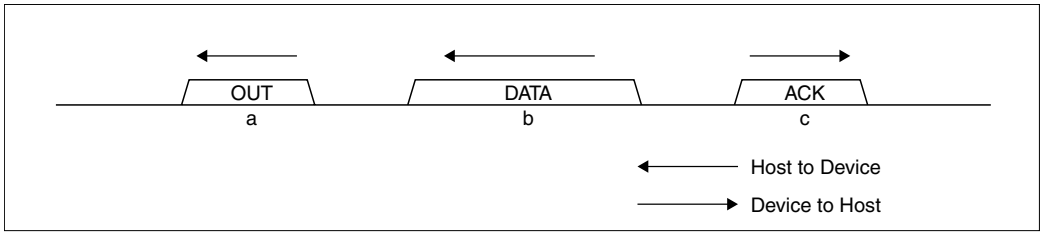


Figure 29.5.1.2 OUT Transaction

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 29.5.1.3 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.

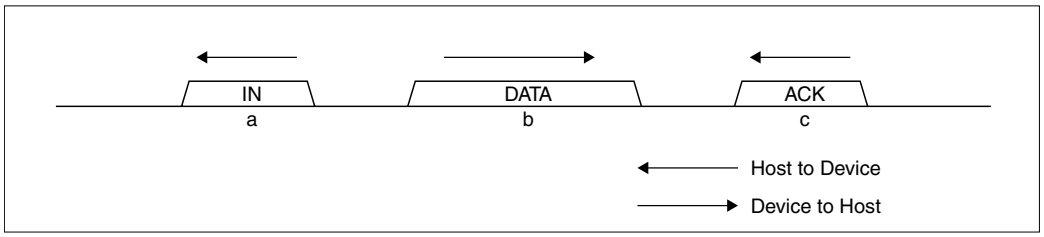


Figure 29.5.1.3 IN Transaction

Control transfer

Control transfer on the EP0 endpoint is controlled as a combination of a number of discrete transactions. Figure 29.5.1.4 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.

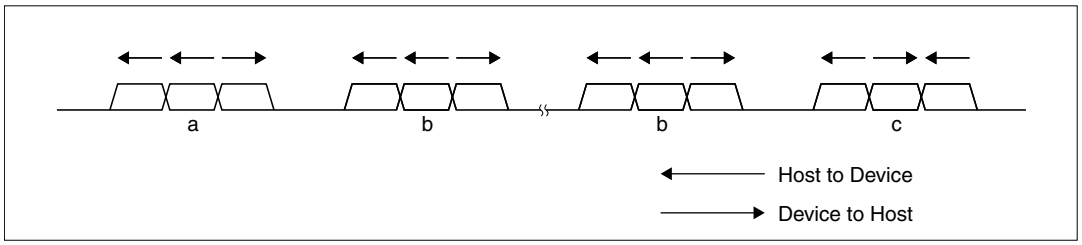


Figure 29.5.1.4 Control Transfer Having an OUT Data Stage

Figure 29.5.1.5 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.

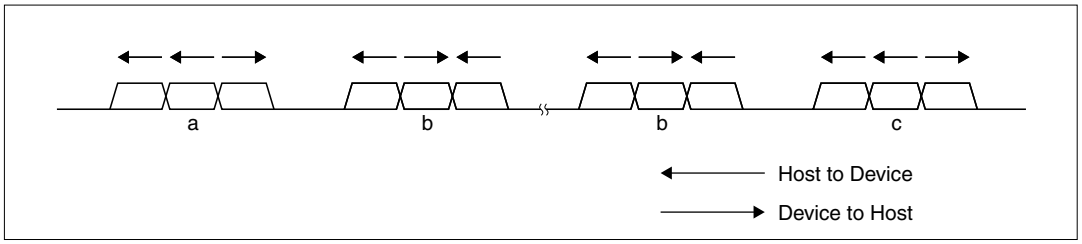


Figure 29.5.1.5 Control Transfer Having an IN Data Stage

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 canceled 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 a 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 (see the "Data flow control" section) 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. The FIFO data can be read via either the CPU interface (EP0, EPa, EPb, EPc, EPd) or the Port interface (EPa, EPb, EPc, EPd).

To read the FIFO data via the CPU interface, 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 a blank FIFO causes dummy reading to be performed.

To read the FIFO data via the Port interface, select one and only one OUT endpoint using the DMA_Join register. Perform the Port interface procedure to read the FIFO data of the selected endpoint; they are read sequentially in the order of reception. Also, you can refer to the DMA_Remain_H and DMA_Remain_L registers to check the number of remaining data. After the FIFO is emptied, the Port interface automatically pauses to perform flow control.

Do not set the CPU and Port interfaces with the CPU_JoinRd and DMA_Join registers for reading from the same endpoint. Additionally, 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.
Data cannot be read from the IN endpoint via the Port interface.
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 29.5.1.6 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.

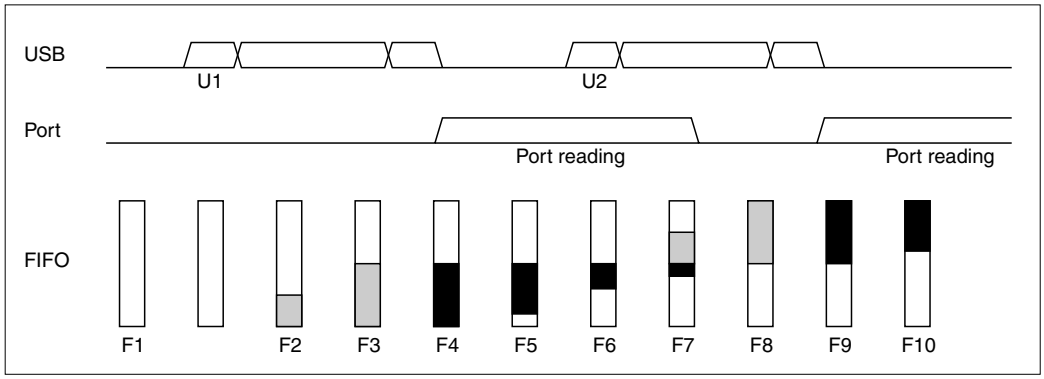


Figure 29.5.1.6 Example of Data Flow in OUT Transfer

(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. Although the Port interface is invoked, no transfer is performed since the FIFO is blank. (The PDREQ signal is negated.)
(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 presence of valid data in the FIFO triggers data transfer via the Port interface. (The PDREQ signal is asserted.)
(F6) As Port transfer develops, the amount of the remaining valid data in the FIFO is reduced.
(F7) Starting the next transaction starts writing data. Port transfer continues as long as any valid data remains.
(F8) Port transfer has 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 presence of valid data in the FIFO restarts Port transfer.

IN transfer

Place data transmitted thorough IN transfer on each endpoint's FIFO. The FIFO data can be written via either the CPU interface (EP0, EPa, EPb, EPc, EPd) or the Port interface (EPa, EPb, EPc, EPd).
To write data into the FIFO via the CPU interface, 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.

To write data into the FIFO via the Port interface, select one and only one IN endpoint using the DMA_Join register. Perform the Port interface procedure to write data into the selected endpoint's FIFO. These data are transmitted in data packets in the order of writing. After the FIFO becomes full, the Port interface automatically pauses to perform flow control.
Do not set the CPU and Port interfaces with the CPU_JoinWr and DMA_Join registers for writing data into the same endpoint. Additionally, 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.
Data cannot be written into an OUT endpoint via the Port interface.
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.
When the DMA_FIFO_Control.AutoEnShort bit is set, the EP*x*{*x*=a,b,c,d}Control.EnShortPkt bit of the relevant endpoint is automatically set if the FIFO still contains any fractional amount of data under the maximum packet size after writing via the Port interface is completed. Using this function provides automatic control to the end that only a non-zero-length short packet is returned, eliminating return of a zero-length data packet.
Figure 29.5.1.7 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.

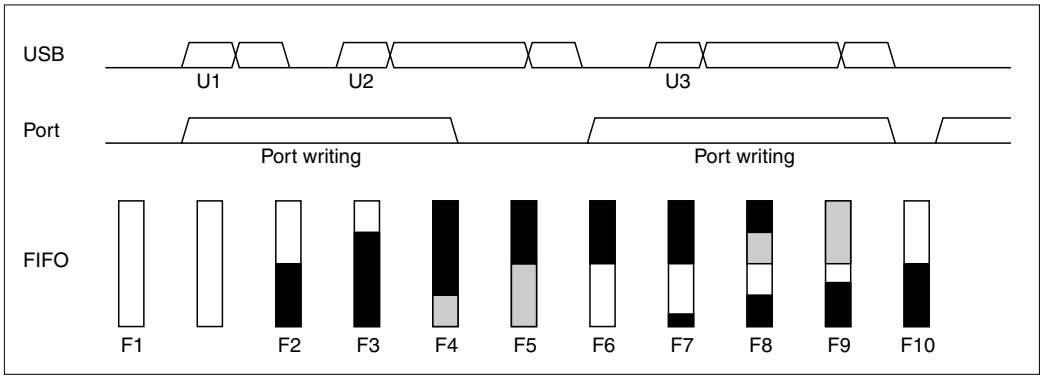


Figure 29.5.1.7 Example of Data Flow in IN Transfer

(U1) In the first IN transaction, a 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) Port transfer is started and valid data is written into the FIFO. (The PDREQ signal is asserted.)
(F3) As the FIFO still has an available space, Port transfer is continuing.
(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 Port transfer to stop. (The PDREQ signal is negated.)
(F5) Although data packet transmission in the IN transaction has been completed, the FIFO region is not freed since the transaction is not closed. Port transfer remains discontinued.
(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, Port transfer is resumed. (The PDREQ signal is asserted.)
(F8) The macro responds to an IN transaction and transmits a data packet. Since the FIFO has some available space, Port transfer continues.

(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, Port transfer continues.

(F10) The FIFO region is freed when the transaction is closed upon reception of an ACK handshake packet. Although Port transfer pauses 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.

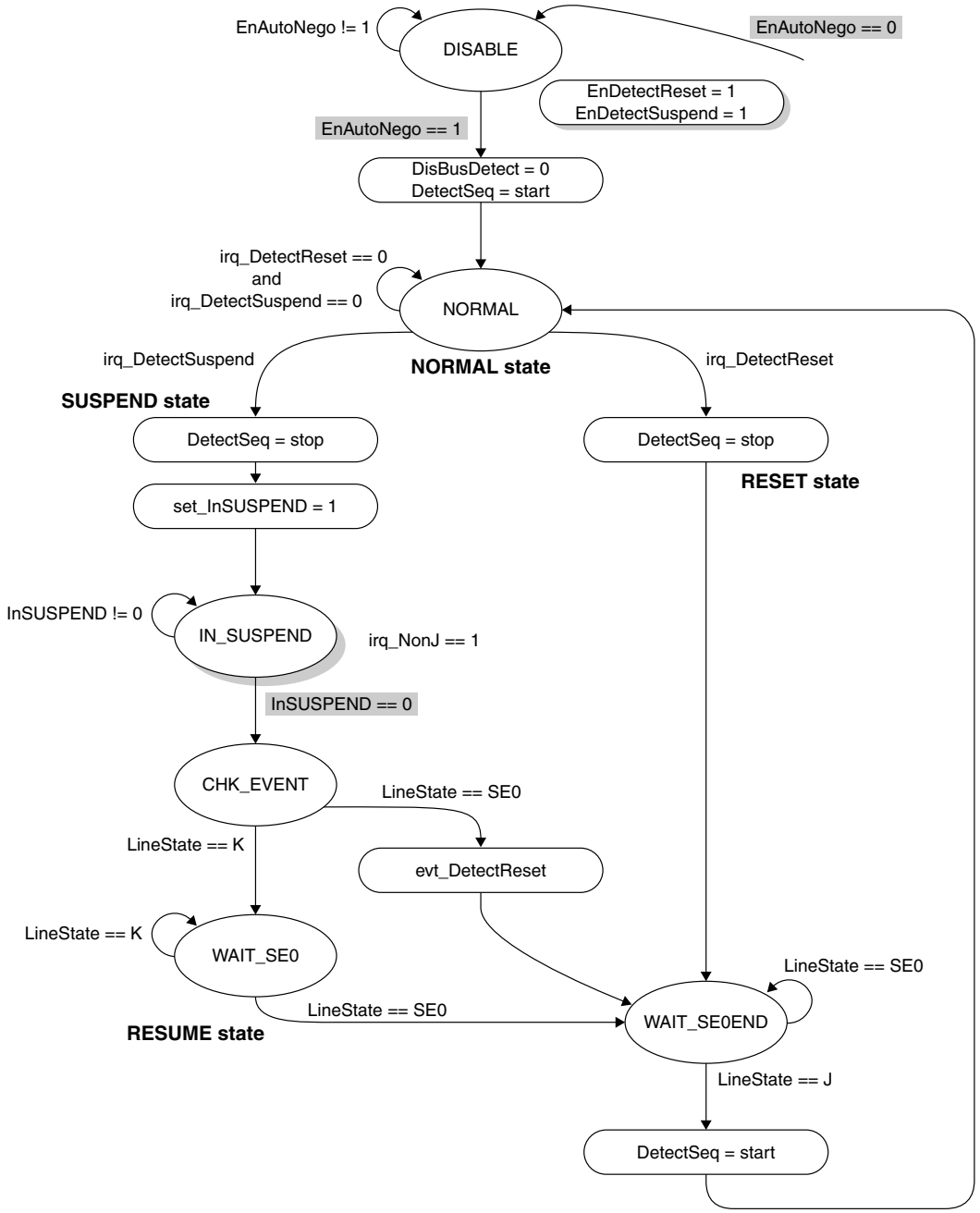


Figure 29.5.1.8 Auto-negotiator

(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/MISC_USB register 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 29.5.1.9 shows the operation when Snooze is performed.)

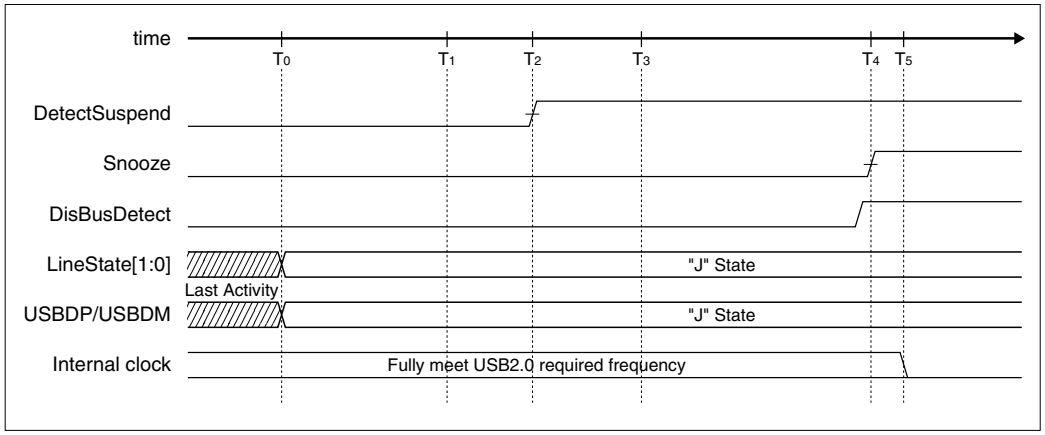


Figure 29.5.1.9 Suspend Timing (FS mode)

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.

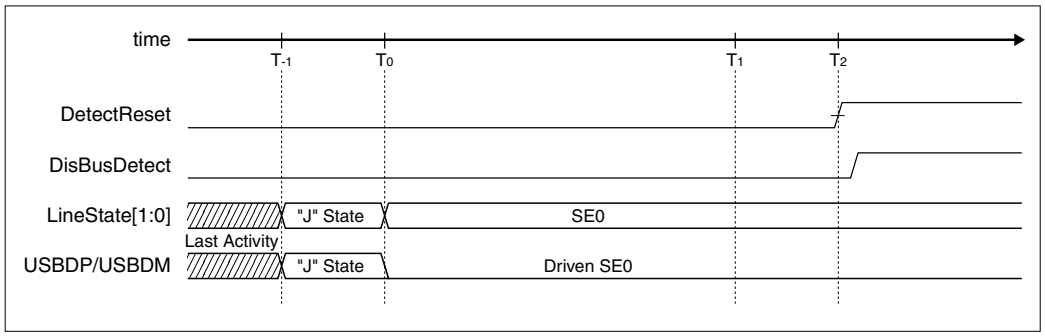


Figure 29.5.1.10 Reset Timing (FS mode)

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 S1C33L27 supports Snooze state. This section describes the operation for issuing Resume when the oscillation circuit is in operation (USB_CKE/CMU_CLKCTL2 register = 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 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_CKE = 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 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.
(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_CKE = 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 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_CKE to start supplying the USB clock (T_1).
(4) Clear USBSNZ (T_2).
(5) The downstream port sends out Reset (T_4).

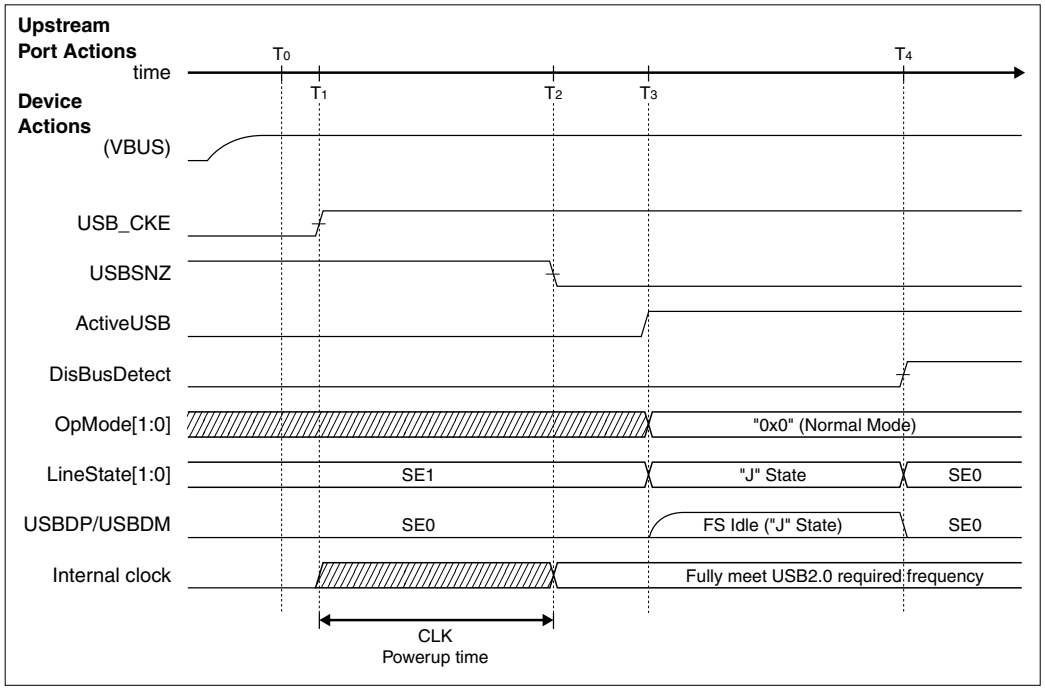


Figure 29.5.1.11 Device Attach Timing

Table 29.5.1.5 Device Attach Timing Values

Timing parameter	Description	Value
T0	VBUS is enabled.	0 (Reference)
T1	Set USB_CKE to 1 (on the firmware). The clock input starts.	T_1
T2	Clear USBSNZ to 0 (on the firmware).	T_1 + 250 µs < T_2
T3	Set ActiveUSB to 1. Set OpMode[1:0] to 0x0 (on the firmware).	T_0 + 100 ms < T_3
T4	The downstream port sends out Reset. Set DisBusDetect to 1 (on the firmware).	T_3 + 100 ms < T_4

29.5.2 FIFO Management

FIFO memory map

This section describes the memory map for the FIFO region.

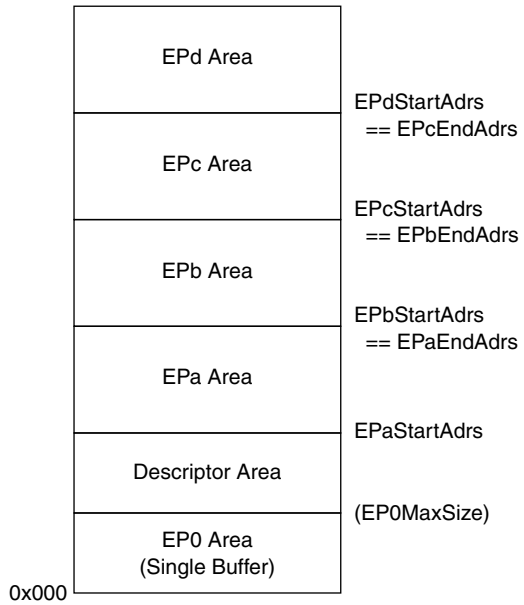


Figure 29.5.2.1 FIFO Memory Map

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.

29.5.3 Port Interface

Functional description

The Port interface is a DMA interface designed for fast data transfer between this macro and the FIFO for its built-in endpoints. It provides Asynchronous DMA Transfer mode for transfer triggered by the Read/Write-strobe signal.

Basic operations

This section describes the basic operations of the Port interface. Note that "DMA" in the descriptions refers to the DMA circuit in the USB macro and "DMAC" refers to the DMA controller module in the S1C33L27.

Register setting

Table 29.5.3.1 lists the registers used for setting basic items of the Port interface. Set desired values for the respective registers. To enable the DMA to write, set the DMA_Join register to connect the Port interface to the endpoint set to the IN direction of the USB. To enable the DMA to read, connect to the endpoint set to the OUT direction.

Do not modify the basic setting registers while the DMA is transferring data (when DMA_Control.DMA_Running is set to 1). We do not guarantee normal operations if the basic setting registers are modified while the DMA in transferring data.

Table 29.5.3.1 Port Interface's Registers for Basic Setting Items

Item	Register/bit	Description
Endpoint connection	DMA_Join.JoinEPr{r=a,b,c,d}DMA	Connects the Port interface to the endpoint of the bit set to 1. Writing/reading is enabled to/from the connected endpoint.
Counter setting	DMA_Count_r{r=HH,HL,LH,LL}	Sets the number of bytes to be down-counted in Countdown mode.
Active port	DMA_Config_0.ActivePort	Enables the port for the Port interface.
Active level	DMA_Config_0.PDREQ_Level DMA_Config_0.PDACK_Level DMA_Config_0.PDRDWR_Level	Sets the active level of the Port interface signal. 0: High-active. 1: Low-active.
RcvLimit mode	DMA_Config_1.RcvLimitMode	Only enabled while writing in Asynchronous transfer mode. If this bit is set to 1, up to 16 bytes of data can be received even after negating PDREQ.
Single-/multi-word	DMA_Config_1.SingleWord	Sets the transfer mode for operation in Asynchronous transfer mode. 0: Multi-word transfer. 1: Single-word transfer.
Count mode	DMA_Config_1.CountMode	Sets Countdown/Free-run mode. 0: Free-run mode. 1: Countdown mode.

DMA transfer

After setting the basic setting registers, write 1 to the DMA_Control.DMA_Go bit to cause the Port interface to start running the DMA. After the DMA starts running, the DMA_Control.DMA_Running bit is set to 1, indicating that the DMA is running.

If the DMA is set to the Countdown mode with DMA_Config_1.CountMode = 1, the DMA completes data transfer when the DMA_Count_HH, HL, LH and LL registers reach 0x00000000. To cancel (negate) the DMA request (PDREQ), provide 1 to the DMA_Control.DMA_Stop bit. After the DMA completes data transfer, the DMA_Control.DMA_Running bit attains 0 and the DMA_IntStat.DMA_Cmp bit 1. At this time, if the DMA_IntEnb.EnDMA_Cmp bit is set, the #INT signal is asserted to the CPU.

Asynchronous DMA transfer

This macro provides an 8-bit asynchronous DMA transfer function that outputs/inputs data, triggered by the Data Transfer Request signal PDREQ, Data Transfer Permit signal PDACK and Read-strobe PDRD/Write-strobe PDWR. This mode only supports the slave functionality, and enables data transfer either in Multi-word or Single-word mode.

Asynchronous multi-word DMA transfer mode - slave

1) Writing operation

The Port interface starts writing operation in Asynchronous multi-word DMA transfer mode when the following register settings are established:

- DMA_Config_1.SingleWord bit = 0
- Direction of the target endpoint = IN

The Port interface starts data transfer on the DMA when 1 is written on the DMA_Control.DMA_Go bit. After data transfer starts on the DMA, the USB macro requests data transfer by asserting PDREQ if any available space is found at the connected endpoint. The DMA loads the data and writes them to the endpoint when PDWR is rising (when the DMA_Config_0.PDRDWR_Level bit is set to 1). When available space is entirely consumed at the endpoint, the interface negates PDREQ to reject data transfer.

If any data is set to the DMA_Latency.DMA_Latency[3:0] bit other than 0h, this mode negates PDREQ once after completing transfer of 4-byte data, and does not assert PDREQ as long as 130 ns × N (N = DMA_ Latency.DMA_Latency[3:0]).

If the DMA is set to the Countdown mode with DMA_Config_1.CountMode = 1, the DMA completes data transfer when the DMA_Count_HH, HL, LH and LL registers reach 0x00000000. To cancel (negate) the DMA request (PDREQ), provide 1 to the DMA_Control.DMA_Stop bit. Note that writing 1 to the DMA_Control.DMA_Stop bit does not stop the DMAC. So to terminate data transfer, first terminate the DMAC (master) and then terminate the macro's DMA transfer.

Note: The S1C33L27 DMAC can only be triggered to start data transfer by the Rising Edge of PDREQ. After that no DMAC trigger will be issued while PDREQ stays active (high level) in multi-word DMA transfer mode. The subsequent DMAC trigger will be issued at the next PDREQ Rising Edge. Therefore, when using the USB macro in multi-word DMA transfer mode, configure the DMAC in successive transfer mode and set the DMAC transfer counter to the same value set in the DMA_Remain_H and DMA_Remain_L registers.

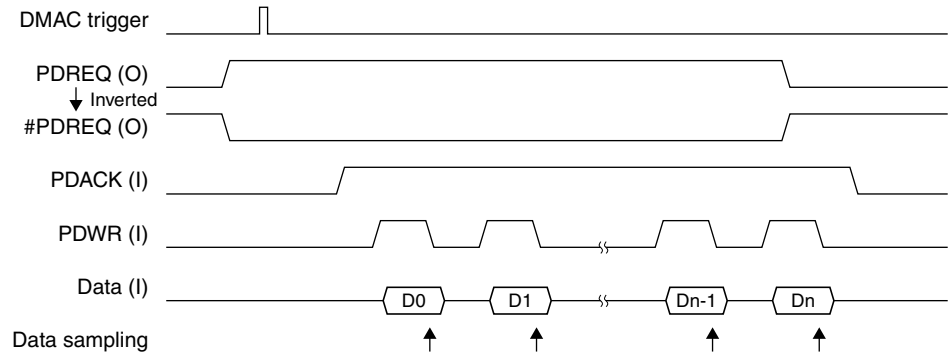


Figure 29.5.3.1 Transfer Waveforms in Asynchronous Multi-Word DMA Transfer Mode - Writing

Providing 1 to the DMA_Config_1.DMA_RcvLimitMode bit enables the RcvLimit mode. The RcvLimit mode is not available in Countdown mode.

When the DMA is writing asynchronously in RcvLimit mode, up to 16 bytes of data can be received even after this macro negates PDREQ.

In this mode, PDREQ is negated when the available space is less than 32 bytes at the relevant endpoint as a result of the DMA's writing operation. However, when PDREQ is negated, 16 bytes of data that have not been written to the endpoint may exist within the internal circuit. Therefore, up to 16 bytes of data can be received after PDREQ is negated.

In this mode, PDREQ is negated before the endpoint becomes completely full. If the region set with the EP{a,b,c,d}StartAdrs register is the same as that set with the EP{a,b,c,d}MaxSize register (Single Buffer), the endpoint never becomes full, and data cannot be transmitted through USB IN transfer.
Therefore, you should set up an area exceeding the EP{a,b,c,d}MaxSize value + 32 bytes to use the Rcv-Limit mode, using the EP{a,b,c,d}StartAdrs register.

Note: In the S1C33L27, the USB DMA data transfer count is determined according to the DMAC transfer counter setting. Negating PDREQ by the USB macro does not affect the transfer count. So in RcvLimit mode, the DMAC continues data transfer until the DMAC transfer counter reaches 0 even after the macro negates PDREQ. Therefore, make sure that the DMAC transfer counter is set properly.

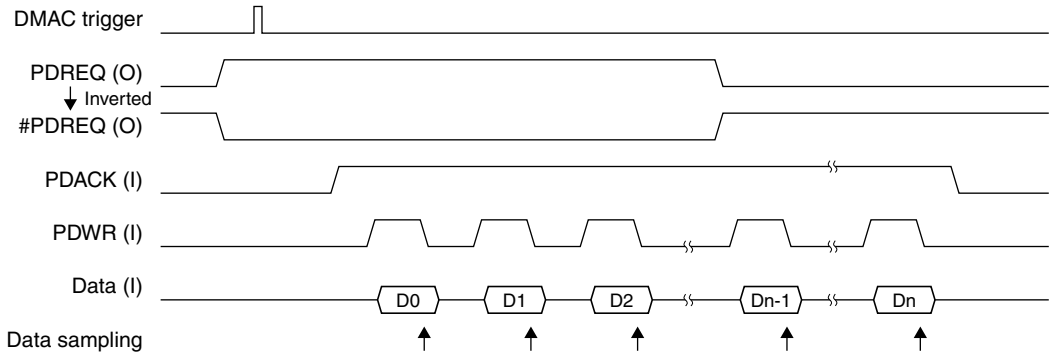


Figure 29.5.3.2 Waveforms in Asynchronous Multi-Word DMA Transfer Mode - Writing (RcvLimit mode)

2) Reading operation

The Port interface starts reading operation in the Asynchronous Multi-word DMA transfer mode when the following register settings are established:

- DMA_Config_1.SingleWord bit = 0
- Direction of the target endpoint = OUT

The Port interface starts data transfer on the DMA when 1 is written on the DMA_Control.DMA_Go bit. After data transfer starts on the DMA, the USB macro requests data transfer by asserting PDREQ if any data exist at the connected endpoint. Turning PDACK to active starts outputting transferred data to the data bus. Have the DMAC (master) load the data while PDRD is rising (when the DMA_Config_0.PDRDWR_Level bit is set to 1). When no data remains at the endpoint, the interface negates PDREQ to reject data transfer.
If any data is set to the DMA_Latency.DMA_Latency[3:0] bit other than 0h, this mode negates PDREQ once after completing transfer of 4-byte data, and does not assert PDREQ as long as 130 ns × N (N = DMA_ Latency.DMA_Latency[3:0]).
If the DMA is set to the Countdown mode with DMA_Config_1.CountMode = 1, the DMA completes data transfer when the DMA_Count_HH, HL, LH and LL registers reach 0x00000000. To cancel (negate) the DMA request (PDREQ), provide 1 to the DMA_Control.DMA_Stop bit. Note that writing 1 to the DMA_Control.DMA_Stop bit does not stop the DMAC. So to terminate data transfer, first terminate the DMAC (master) and then terminate the macro's DMA transfer.

Note: The S1C33L27 DMAC can only be triggered to start data transfer by the Rising Edge of PDREQ. After that no DMAC trigger will be issued while PDREQ stays active (high level) in multi-word DMA transfer mode. The subsequent DMAC trigger will be issued at the next PDREQ Rising Edge. Therefore, when using the USB macro in multi-word DMA transfer mode, configure the DMAC in successive transfer mode and set the DMAC transfer counter to the same value set in the DMA_Remain_H and DMA_Remain_L registers.

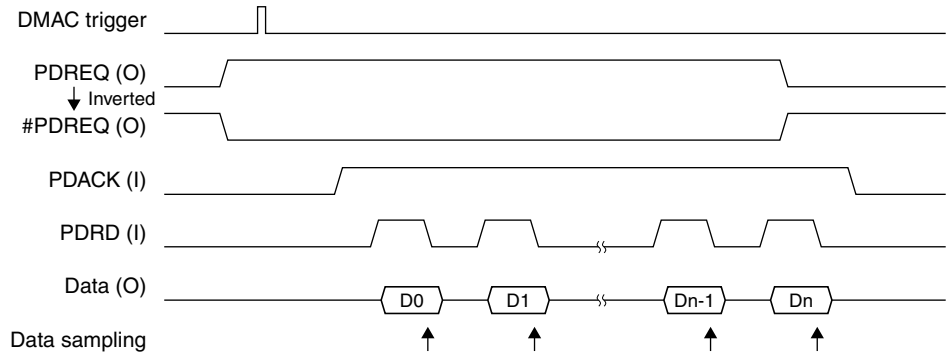


Figure 29.5.3.3 Transfer Waveforms in Asynchronous Multi-Word DMA Transfer Mode - Reading

Asynchronous single-word DMA transfer mode - slave

1) Writing operation

The Port interface starts writing operation in Asynchronous single-word DMA transfer mode when the following register settings are established:

- DMA_Config_1.SingleWord bit = 1
- Direction of the target endpoint = IN

The Port interface starts data transfer on the DMA when 1 is written on the DMA_Control.DMA_Go bit. After data transfer starts on the DMA, the USB macro requests data transfer by asserting PDREQ if any available space is found at the connected endpoint. The DMA loads the data and writes them to the endpoint when PDWR is rising (when the DMA_Config_0.PDRDWR_Level bit is set to 1). This mode negates PDREQ after transferring 1-byte data (PDWR becomes active).

At this point, if any space is still available at the endpoint, it requests data transfer by asserting PDREQ. If there is no available space left at the endpoint, PDREQ is not asserted and data transfer is rejected.

If any data is set to the DMA_Latency.DMA_Latency[3:0] bit other than 0h, this mode negates PDREQ once after completing transfer of 4-byte data, and does not assert PDREQ as long as 130 ns × N (N = DMA_Latency.DMA_Latency[3:0]).

If the DMA is set to the Countdown mode with DMA_Config_1.CountMode = 1, the DMA completes data transfer when the DMA_Count_HH, HL, LH and LL registers reach 0x00000000. To cancel (negate) the DMA request (PDREQ), provide 1 to the DMA_Control.DMA_Stop bit. Note that writing 1 to the DMA_Control.DMA_Stop bit does not stop the DMAC. So to terminate data transfer, first terminate the DMAC (master) and then terminate the macro's DMA transfer.

Note: The S1C33L27 DMAC can only be triggered to start data transfer by the Rising Edge of PDREQ. The subsequent DMAC trigger will be issued at the next PDREQ Rising Edge. When the DMAC transfer counter reaches 0, DMA transfer will not be started even if a DMAC trigger is issued. Therefore, when using the USB macro in single-word DMA transfer mode, configure the DMAC in single transfer mode and set the DMAC transfer counter to a value equal to or less than that set in the DMA_Remain_H and DMA_Remain_L registers.

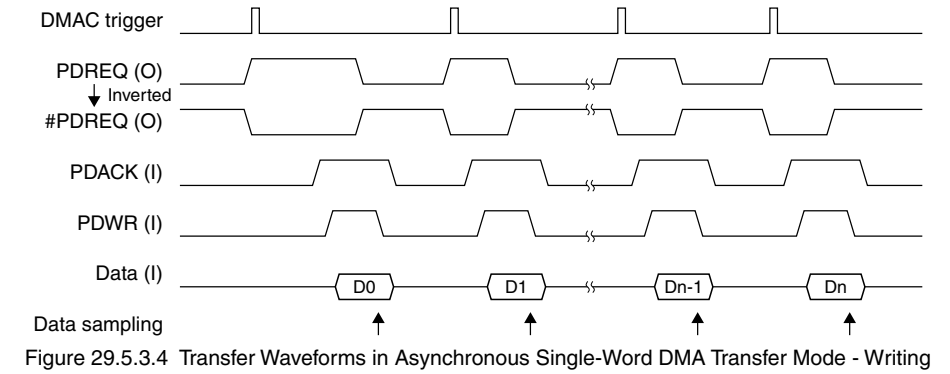


Figure 29.5.3.4 Transfer Waveforms in Asynchronous Single-Word DMA Transfer Mode - Writing

2) Reading operation

The Port interface starts reading operation in the Asynchronous single-word DMA transfer mode when the following register settings are established:

- DMA_Config_1.SingleWord bit = 1
- Direction of the target endpoint = OUT

The Port interface starts data transfer on the DMA when 1 is written on the DMA_Control.DMA_Go bit. After data transfer starts on the DMA, the USB macro requests data transfer by asserting PDREQ if any data exist at the connected endpoint. Turning PDACK to active starts outputting transferred data to the data bus. Have the DMAC (master) load the data while PDRD is rising (when the DMA_Config_0.PDRDWR_Level bit is set to 1). This mode negates PDREQ after transferring 1-byte data (PDRD becomes active). At this point, if any data still remain at the endpoint, it requests data transfer by asserting PDREQ. If there are no data left at the endpoint, PDREQ is not asserted and data transfer is rejected.

If any data is set to the DMA_Latency.DMA_Latency[3:0] bit other than 0h, this mode negates PDREQ once after completing transfer of 4-byte data, and does not assert PDREQ as long as 130 ns × N (N = DMA_Latency.DMA_Latency[3:0]).

If the DMA is set to the Countdown mode with DMA_Config_1.CountMode = 1, the DMA completes data transfer when the DMA_Count_HH, HL, LH and LL registers reach 0x00000000. To cancel (negate) the DMA request (PDREQ), provide 1 to the DMA_Control.DMA_Stop bit. Note that writing 1 to the DMA_Control.DMA_Stop bit does not stop the DMAC. So to terminate data transfer, first terminate the DMAC (master) and then terminate the macro's DMA transfer.

Note: The S1C33L27 DMAC can only be triggered to start data transfer by the Rising Edge of PDREQ. The subsequent DMAC trigger will be issued at the next PDREQ Rising Edge. When the DMAC transfer counter reaches 0, DMA transfer will not be started even if a DMAC trigger is issued. Therefore, when using the USB macro in single-word DMA transfer mode, configure the DMAC in single transfer mode and set the DMAC transfer counter to a value equal to or less than that set in the DMA_Remain_H and DMA_Remain_L registers.

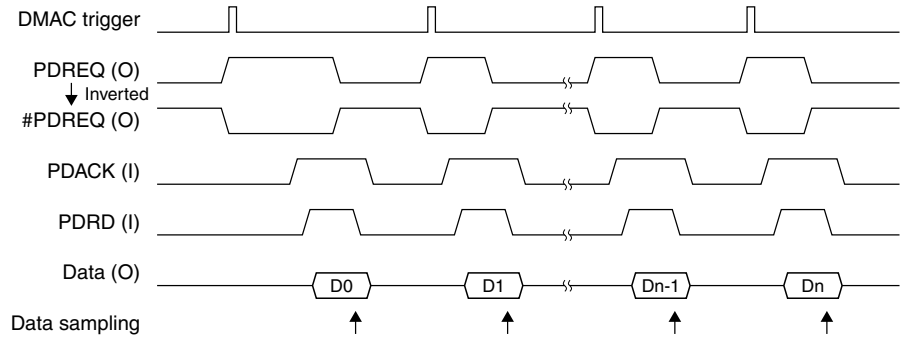


Figure 29.5.3.5 Transfer Waveforms in Asynchronous Single-Word DMA Transfer Mode - Reading

29.5.4 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/MISC_USB register, 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.

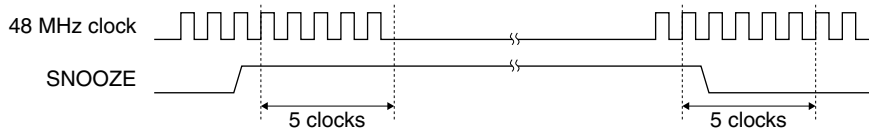


Figure 29.5.4.1 Snooze Sequence

Snooze mode should be set or canceled by the following procedure:

Setting snooze mode

(1) Write 0x96 to the MISC_PROT0 register to disable write protection for the Misc registers.
(2) Set USBSNZ in the MISC_USB register to 1 to enable the snooze control.
(3) Write a value other than 0x96 to the MISC_PROT0 register to enable write protection for the Misc registers.
(4) Write 0x96 to the CMU_PROTECT register to disable write protection for the CMU registers.
(5) Set USB_CKE in the CMU_CLKCTL2 register to 0 to stop supplying the USB clock.
(6) Write a value other than 0x96 to the CMU_PROTECT register to enable write protection for the CMU registers.

Canceling snooze mode

(1) Write 0x96 to the CMU_PROTECT register to disable write protection for the CMU registers.
(2) Set USB_CKE in the CMU_CLKCTL2 register to 1 to start supplying the USB clock.
(3) Write a value other than 0x96 to the CMU_PROTECT register to enable write protection for the CMU registers.
(4) Write 0x96 to the MISC_PROT0 register to disable write protection for the Misc registers.
(5) Set USBSNZ in the MISC_USB register to disable the snooze control.
(6) Write a value other than 0x96 to the MISC_PROT0 register to enable write protection for the Misc registers.

29.6 Registers

29.6.1 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
0x300c00	MainIntStat	R/(W)	0x00	***SIE_IntStat***	EPrIntStat	DMA_IntStat	FIFO_IntStat	–	–	EP0IntStat	RcvEP0SETUP
0x300c01	SIE_IntStat	R/(W)	0x00	***VBUS_Changed***	***NonJ***	DetectReset	DetectSuspend	RcvSOF	DetectJ	–	SetAddressCmp
0x300c02	EPrIntStat	R	0x00	–	–	–	–	EPdIntStat	EPcIntStat	EPbIntStat	EPaIntStat
0x300c03	DMA_IntStat	R/(W)	0x00	–	–	–	–	–	–	DMA_CountUp	DMA_Cmp
0x300c04	FIFO_IntStat	R/(W)	0x00	DescriptorCmp	–	–	–	–	–	FIFO_IN_Cmp	FIFO_OUT_Cmp
0x300c05											
0x300c06											
0x300c07	EP0IntStat	R/(W)	0x00	–	–	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x300c08	EPaIntStat	R/(W)	0x00	–	OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x300c09	EPbIntStat	R/(W)	0x00	–	OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x300c0a	EPcIntStat	R/(W)	0x00	–	OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x300c0b	EPdIntStat	R/(W)	0x00	–	OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x300c0c											
0x300c0d											
0x300c0e											
0x300c0f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c10	MainIntEnb	R/W	0x00	***EnSIE_IntStat***	EnEPrIntStat	EnDMA_IntStat	EnFIFO_IntStat	–	–	EnEP0IntStat	EnRcvEP0SETUP
0x300c11	SIE_IntEnb	R/W	0x00	***EnVBUS_Changed***	***EnNonJ***	EnDetectReset	EnDetectSuspend	EnRcvSOF	EnDetectJ	–	EnSetAddressCmp
0x300c12	EPrIntEnb	R/W	0x00	–	–	–	–	EnEPdIntStat	EnEPcIntStat	EnEPbIntStat	EnEPaIntStat
0x300c13	DMA_IntEnb	R/W	0x00	–	–	–	–	–	–	EnDMA_CountUp	EnDMA_Cmp
0x300c14	FIFO_IntEnb	R/W	0x00	EnDescriptorCmp	–	–	–	–	–	EnFIFO_IN_Cmp	EnFIFO_OUT_Cmp
0x300c15											
0x300c16											
0x300c17	EP0IntEnb	R/W	0x00	–	–	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x300c18	EPaIntEnb	R/W	0x00	–	EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x300c19	EPbIntEnb	R/W	0x00	–	EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x300c1a	EPcIntEnb	R/W	0x00	–	EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x300c1b	EPdIntEnb	R/W	0x00	–	EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x300c1c											
0x300c1d											
0x300c1e											
0x300c1f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c20	RevisionNum	R	0x12	***Revision Num[7:0]***							
0x300c21	USB_Control	R/W	0x00	DisBusDetect	EnAutoNego	InSUSPEND	StartDetectJ	SendWakeup	–	–	ActiveUSB
0x300c22	USB_Status	R	0xXX	***VBUS***	***1(FS)***	–	–	–	–	***LineState[1:0]***	
0x300c23	XcvrControl	R/W	0x01	RpuEnb	–	–	–	–	–	OpMode[1:0]	
0x300c24	USB_Test	R/W	0x00	EnUSB_Test	–	–	–	Test_SE0_NAK	Test_J	Test_K	Test_Packet
0x300c25	EPnControl	W	0x00	AllForceNAK	EPrForceSTALL	AllFIFO_Clr	–	–	–	–	EP0FIFO_Clr
0x300c26	EPrFIFO_Clr	W	0x00	–	–	–	–	EPdFIFO_Clr	EPcFIFO_Clr	EPbFIFO_Clr	EPaFIFO_Clr
0x300c27											
0x300c28											
0x300c29											
0x300c2a											
0x300c2b											
0x300c2c											
0x300c2d											
0x300c2e	FrameNumber_H	R	0x80	FnInvalid	–	–	–	–	FrameNumber[10:8]		
0x300c2f	FrameNumber_L	R	0x00	FrameNumber[7:0]							

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c30	EP0Setup_0	R	0x00	EP0Setup_0[7:0]							
0x300c31	EP0Setup_1	R	0x00	EP0Setup_1[7:0]							
0x300c32	EP0Setup_2	R	0x00	EP0Setup_2[7:0]							
0x300c33	EP0Setup_3	R	0x00	EP0Setup_3[7:0]							
0x300c34	EP0Setup_4	R	0x00	EP0Setup_4[7:0]							
0x300c35	EP0Setup_5	R	0x00	EP0Setup_5[7:0]							
0x300c36	EP0Setup_6	R	0x00	EP0Setup_6[7:0]							
0x300c37	EP0Setup_7	R	0x00	EP0Setup_7[7:0]							
0x300c38	USB_Address	R/W	0x00	AutoSetAddress	USB_Address[6:0]						
0x300c39	EP0Control	R/W	0x00	INxOUT	–	–	–	–	–	–	ReplyDescriptor
0x300c3a	EP0ControlIN	R/W	0x00	–	EnShortPkt	–	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x300c3b	EP0ControlOUT	R/W	0x00	AutoForceNAK	–	–	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x300c3c											
0x300c3d											
0x300c3e											
0x300c3f	EP0MaxSize	R/W	0x08	–	EP0MaxSize[6:3]				–	–	–

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c40	EPaControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x300c41	EPbControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x300c42	EPcControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x300c43	EPdControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x300c44											
0x300c45											
0x300c46											
0x300c47											
0x300c48											
0x300c49											
0x300c4a											
0x300c4b											
0x300c4c											
0x300c4d											
0x300c4e											
0x300c4f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c50	EPaMaxSize_H	R/W	0x00	–	–	–	–	–	–	EPaMaxSize[9:8]	
0x300c51	EPaMaxSize_L	R/W	0x00	EPaMaxSize[7:0]							
0x300c52	EPaConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint	–	EndPointNumber[3:0]			
0x300c53	EPaConfig_1	R/W	0x00	ISO	ISO_CRCmode	–	–	–	–	–	–
0x300c54	EPbMaxSize_H	R/W	0x00	–	–	–	–	–	–	EPbMaxSize[9:8]	
0x300c55	EPbMaxSize_L	R/W	0x00	EPbMaxSize[7:0]							
0x300c56	EPbConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint	–	EndPointNumber[3:0]			
0x300c57	EPbConfig_1	R/W	0x00	ISO	ISO_CRCmode	–	–	–	–	–	–
0x300c58	EPcMaxSize_H	R/W	0x00	–	–	–	–	–	–	EPcMaxSize[9:8]	
0x300c59	EPcMaxSize_L	R/W	0x00	EPcMaxSize[7:0]							
0x300c5a	EPcConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint	–	EndPointNumber[3:0]			
0x300c5b	EPcConfig_1	R/W	0x00	ISO	ISO_CRCmode	–	–	–	–	–	–
0x300c5c	EPdMaxSize_H	R/W	0x00	–	–	–	–	–	–	EPdMaxSize[9:8]	
0x300c5d	EPdMaxSize_L	R/W	0x00	EPdMaxSize[7:0]							
0x300c5e	EPdConfig_0	R/W	0x00	INxOUT	ToggleMode	EnEndPoint	–	EndPointNumber[3:0]			
0x300c5f	EPdConfig_1	R/W	0x00	ISO	ISO_CRCmode	–	–	–	–	–	–

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c70	EPaStartAdrs_H	R/W	0x00	–	–	–	–	EPaStartAdrs[11:8]			
0x300c71	EPaStartAdrs_L	R/W	0x00	EPaStartAdrs[7:2]						–	–
0x300c72	EPbStartAdrs_H	R/W	0x00	–	–	–	–	EPbStartAdrs[11:8]			
0x300c73	EPbStartAdrs_L	R/W	0x00	EPbStartAdrs[7:2]						–	–
0x300c74	EPcStartAdrs_H	R/W	0x00	–	–	–	–	EPcStartAdrs[11:8]			
0x300c75	EPcStartAdrs_L	R/W	0x00	EPcStartAdrs[7:2]						–	–
0x300c76	EPdStartAdrs_H	R/W	0x00	–	–	–	–	EPdStartAdrs[11:8]			
0x300c77	EPdStartAdrs_L	R/W	0x00	EPdStartAdrs[7:2]						–	–
0x300c78											
0x300c79											
0x300c7a											
0x300c7b											
0x300c7c											
0x300c7d											
0x300c7e											
0x300c7f											

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c80	CPU_JoinRd	R/W	0x00	–	–	–	–	JoinEPdRd	JoinEPcRd	JoinEPbRd	JoinEPaRd
0x300c81	CPU_JoinWr	R/W	0x00	–	–	–	–	JoinEPdWr	JoinEPcWr	JoinEPbWr	JoinEPaWr
0x300c82	EnEPnFIFO_Access	R/W	0x00	–	–	–	–	–	–	EnEPnFIFO_Wr	EnEPnFIFO_Rd
0x300c83	EPnFIFOforCPU	R/W	0xXX	EPnFIFOData[7:0]							
0x300c84	EPnRdRemain_H	R	0x00	–	–	–	–	EPnRdRemain[11:8]			
0x300c85	EPnRdRemain_L	R	0x00	EPnRdRemain[7:0]							
0x300c86	EPnWrRemain_H	R	0x00	–	–	–	–	EPnWrRemain[11:8]			
0x300c87	EPnWrRemain_L	R	0x00	EPnWrRemain[7:0]							
0x300c88	DescAdrs_H	R/W	0x00	–	–	–	–	DescAdrs[11:8]			
0x300c89	DescAdrs_L	R/W	0x00	DescAdrs[7:0]							
0x300c8a	DescSize_H	R/W	0x00	–	–	–	–	–	–	DescSize[9:8]	
0x300c8b	DescSize_L	R/W	0x00	DescSize[7:0]							
0x300c8c											
0x300c8d											
0x300c8e											
0x300c8f	DescDoor	R/W	0x00	DescMode[7:0]							

Address	Register name	R/W	Init	D7	D6	D5	D4	D3	D2	D1	D0
0x300c90	DMA_FIFO_Control	R/W	0x00	FIFO_Running	AutoEnShort	–	–	–	–	–	–
0x300c91	DMA_Join	R/W	0x00	–	–	–	–	JoinEPdDMA	JoinEPcDMA	JoinEPbDMA	JoinEPaDMA
0x300c92	DMA_Control	R/W	0x00	DMA_Running	PDREQ	PDACK	–	CounterClr	–	DMA_Stop	DMA_Go
0x300c93											
0x300c94	DMA_Config_0	R/W	0x00	ActivePort	–	–	–	PDREQ_Level	PDACK_Level	PDRDWR_Level	–
0x300c95	DMA_Config_1	R/W	0x00	RcvLimitMode	–	–	–	SingleWord	–	–	CountMode
0x300c96											
0x300c97	DMA_Latency	R/W	0x00	–	–	–	–	DMA_Latency[3:0]			
0x300c98	DMA_Remain_H	R	0x00	–	–	–	–	DMA_Remain[11:8]			
0x300c99	DMA_Remain_L	R	0x00	DMA_Remain[7:0]							
0x300c9a											
0x300c9b											
0x300c9c	DMA_Count_HH	R/W	0x00	DMA_Count[31:24]							
0x300c9d	DMA_Count_HL	R/W	0x00	DMA_Count[23:16]							
0x300c9e	DMA_Count_LH	R/W	0x00	DMA_Count[15:8]							
0x300c9f	DMA_Count_LL	R/W	0x00	DMA_Count[7:0]							

29.6.2 Detailed Description of Registers

MainIntStat (Main Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
MainIntStat (Main interrupt status)	**0x300c00** (8 bits)	D7	**SIE_IntStat**	1	SIE interrupts	0	None	0	R	
		D6	**EPrIntStat**	1	EPr interrupts	0	None	0	R	
		D5	**DMA_IntStat**	1	DMA interrupts	0	None	0	R	
		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 DMA_IntStat

Shows a cause of interrupt indirectly.
When the DMA_IntStat register has a cause of interrupt and the DMA_IntEnb register bit corresponding to the cause of interrupt is enabled, this bit is set to 1.

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, 1 and 1, respectively.

SIE_IntStat (SIE Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
SIE_IntStat (SIE interrupt status)	0x300c01 (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 In-SUSPEND bit of the USB_Control register is set to 1. This bit is valid during snooze as well.

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 MISC_USB register 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.

EPrIntStat (EPr Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPrIntStat (EPr interrupt status)	**0x300c02** (8 bits)	D7–4	**–**	–				–	–	0 when being read.
		D3	**EPdIntStat**	1	EPd interrupt	0	None	0	R	
		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.

DMA_IntStat (DMA Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_IntStat (DMA interrupt status)	**0x300c03** (8 bits)	D7–2	**–**	–				–	–	0 when being read.
		D1	**DMA_CountUp**	1	DMA counter overflow	0	None	0	R(W)	
		D0	**DMA_Cmp**	1	DMA complete	0	None	0	R(W)	

This register displays the interrupt status of the DMA.

D[7:2] Reserved

D1 DMA_CountUp

Shows a cause of interrupt directly.
Set to 1 when values of DMA_Count_HH, HL, LH and LL overflow while the DMA operates in the free run mode. Then values of DMA_Count_HH, HL, LH and LL return to 0, and the DMA operation continues.

D0 DMA_Cmp

Shows a cause of interrupt directly.
Set to 1 when the DMA is stopped or completes the specified number of transfer operations and the end processing.

FIFO_IntStat (FIFO Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
FIFO_IntStat (FIFO interrupt status)	**0x300c04** (8 bits)	D7	**DescriptorCmp**	1	Descriptor complete	0	None	0	R(W)	
		D6–2	**–**	–				–	–	0 when being read.
		D1	**FIFO_IN_Cmp**	1	IN FIFO Complete	0	None	0	R(W)	
		D0	**FIFO_OUT_Cmp**	1	OUT FIFO complete	0	None	0	R(W)	

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[6:2] Reserved

D1 FIFO_IN_Cmp

Shows a cause of interrupt directly.
If the transfer direction of the endpoint bound to DMA (refer to the DMA_Join register) is the IN direction, this bit is set to 1 when the FIFO becomes empty after completion of the DMA transfer.

D0 FIFO_OUT_Cmp

Shows a cause of interrupt directly.
If the transfer direction of the endpoint bound to DMA (refer to the DMA_Join register) is the OUT direction, this bit is set to 1 when the DMA transfer is completed.

EP0IntStat (EP0 Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0IntStat (EP0 interrupt status)	0x300c07 (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.

EPaIntStat (EPa Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaIntStat (EPa interrupt status)	0x300c08 (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 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.

EPbIntStat (EPb Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbIntStat (EPb interrupt status)	0x300c09 (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 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.

EPcIntStat (EPc Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcIntStat (EPc interrupt status)	0x300c0a (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 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.

EPdIntStat (EPd Interrupt Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdIntStat (EPd interrupt status)	0x300c0b (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.

MainIntEnb (Main Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
MainIntEnb (Main interrupt enable)	0x300c10 (8 bits)	D7	EnSIE_IntStat	1	Enable	0	Disable	0	R/W	
		D6	EnEPrIntStat					0	R/W	
		D5	EnDMA_IntStat					0	R/W	
		D4	EnFIFO_IntStat					0	R/W	
		D3–2	–	–				–	–	0 when being read.
		D1	EnEP0IntStat	1	Enable	0	Disable	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.

SIE_IntEnb (SIE Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
SIE_IntEnb (SIE interrupt enable)	0x300c11 (8 bits)	D7	EnVBUS_Changed	1	Enable	0	Disable	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	Enable	0	Disable	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.

EPrIntEnb (EPr Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPrIntEnb (EPr interrupt enable)	0x300c12 (8 bits)	D7–4	–	–				–	–	0 when being read.
		D3	EnEPdIntStat	1	Enable	0	Disable	0	R/W	
		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.

DMA_IntEnb (DMA Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_IntEnb (DMA interrupt enable)	0x300c13 (8 bits)	D7–2	–	–				–	–	0 when being read.
		D1	EnDMA_CountUp	1	Enable	0	Disable	0	R/W	
		D0	EnDMA_Cmp					0	R/W	

This register enables/disables assertion of the DMA_IntStat bit of the MainIntStat register with the cause of interrupt of the DMA_IntStat register.

FIFO_IntEnb (FIFO Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
FIFO_IntEnb (FIFO interrupt enable)	0x300c14 (8 bits)	D7	EnDescriptorCmp	1	Enable	0	Disable	0	R/W	
		D6–2	–	–				–	–	0 when being read.
		D1	EnFIFO_IN_Cmp	1	Enable	0	Disable	0	R/W	
		D0	EnFIFO_OUT_Cmp					0	R/W	

This register enables/disables assertion of the FIFO_IntStat bit of the MainIntStat register with the cause of interrupt of the FIFO_IntStat register.

EP0IntEnb (EP0 Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0IntEnb (EP0 interrupt enable)	0x300c17 (8 bits)	D7–6	–	–				–	–	0 when being read.
		D5	EnIN_TranACK	1	Enable	0	Disable	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 EP0IntStat bit of the MainIntStat register with the cause of interrupt of the EP0IntStat register.

EPaIntEnb (EPa Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaIntEnb (EPa interrupt enable)	0x300c18 (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enable	0	Disable	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	

This register enables/disables assertion of the EPaIntStat bit of the EPrIntStat register with the cause of interrupt of the EPaIntStat register.

EPbIntEnb (EPb Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbIntEnb (EPb interrupt enable)	0x300c19 (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enable	0	Disable	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	

This register enables/disables assertion of the EPbIntStat bit of the EPrIntStat register with the cause of interrupt of the EPbIntStat register.

EPcIntEnb (EPc Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcIntEnb (EPc interrupt enable)	0x300c1a (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enable	0	Disable	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	

This register enables/disables assertion of the EPcIntStat bit of the EPrIntStat register with the cause of interrupt of the EPcIntStat register.

EPdIntEnb (EPd Interrupt Enable)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdIntEnb (EPd interrupt enable)	0x300c1b (8 bits)	D7	–	–				–	–	0 when being read.
		D6	EnOUT_ShortACK	1	Enable	0	Disable	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	

This register enables/disables assertion of the EPdIntStat bit of the EPrIntStat register with the cause of interrupt of the EPdIntStat register.

RevisionNum (Revision Number)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
RevisionNum (Revision number)	0x300c20 (8 bits)	D7–0	RevisionNum[7:0]	Revision number (0x12)	0x12	R	

This register shows the revision number of the USB function controller. This register is valid during snooze as well.

USB_Control (USB Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
USB_Control (USB control)	0x300c21 (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	Deactivate 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.

USB_Status (USB Status)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
USB_Status (USB status)	0x300c22 (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.
		D1–0	LineState[1:0]	LineState[1:0]	DP/DM			X	R	
				0x3	SE1					
				0x2	K					
				0x1	J					
				0x0	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[1:0]	DP/DM
0x3	SE1
0x2	K
0x1	J
0x0	SE0

XcvrControl (Xcvr Control)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
XcvrControl (Xcvr control)	0x300c23 (8 bits)	D7	RpuEnb	1 Enable pull-up / 0 Disable pull-up	0	R/W	
		D6–2	–	–	–	–	0 when being read.
		D1–0	OpMode[1:0]	0x3 reserved 0x2 Disable bitstuffing and NRZI encoding 0x1 Non-driving 0x0 Normal operation	0x1	R/W	

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[1:0]	Operation mode
0x3	Reserved
0x2	Disable bitstuffing and NRZI encoding
0x1	Non-driving
0x0	Normal operation

* When the USB cable is pulled off, it is recommended to set this register to 0x1.

USB_Test (USB Test)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
USB_Test (USB test)	0x300c24 (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 Bitstuffing 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 Bitstuffing 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.)

EPnControl (Endpoint Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPnControl (Endpoint control)	0x300c25 (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

Sets the ForceNAK bit of all endpoints to 1.

D6 EPrForceSTALL

Sets the ForceSTALL bit of EPa, EPb, EPc and EPd endpoints to 1.

D5 AllFIFO_Clr

Clears 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.

Do not set this bit to 1 during start operation of the general port (when the DMA_Running bit of the DMA_Control register is 1). Otherwise, a malfunction may occur.

D[4:1] Reserved

D0 EP0FIFO_Clr

Clears the FIFO of the endpoint EP0. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

EPrFIFO_Clr (EPr FIFO Clear)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPrFIFO_Clr (EPr FIFO clear)	0x300c26 (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

Clears the FIFO of the endpoint EPd. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

Do not set this bit to 1 when the endpoint EPd is connected to the general port (the JoinEPdDMA bit of the DMA_Join register is set to 1) and the start operation of the general port is being done (when the DMA_Running bit of the DMA_Control register is 1). Otherwise, a malfunction may occur.

D2 EPcFIFO_Clr

Clears the FIFO of the endpoint EPc. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

Do not set this bit to 1 when the endpoint EPc is connected to the general port (the JoinEPcDMA bit of the DMA_Join register is set to 1) and the start operation of the general port is being done (when the DMA_Running bit of the DMA_Control register is 1). Otherwise, a malfunction may occur.

D1 EPbFIFO_Clr

Clears the FIFO of the endpoint EPb. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

Do not set this bit to 1 when the endpoint EPb is connected to the general port (the JoinEPbDMA bit of the DMA_Join register is set to 1) and the start operation of the general port is being done (when the DMA_Running bit of the DMA_Control register is 1). Otherwise, a malfunction may occur.

D0 EPaFIFO_Clr

Clears the FIFO of the endpoint EPa. This bit is automatically set 0 (to be cleared) after completing the FIFO clear operation.

Do not set this bit to 1 when the endpoint EPa is connected to the general port (the JoinEPaDMA bit of the DMA_Join register is set to 1) and the start operation of the general port is being done (when the DMA_Running bit of the DMA_Control register is 1). Otherwise, a malfunction may occur.

FrameNumber_H (Frame Number HIGH)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
FrameNumber_H (Frame number high)	**0x300c2e** (8 bits)	D7	**FnInvalid**	1 Invalid frame number / 0 Valid frame number	1	R	
		D6–3	**–**	–	–	–	0 when being read.
		D2–0	**FrameNumber[10:8]**	Frame number high	0x0	R	

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.

FrameNumber_L (Frame Number LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
FrameNumber_L (Frame number low)	**0x300c2f** (8 bits)	D7–0	**FrameNumber[7:0]**	Frame number low	0x0	R	

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.

EP0Setup_0 (EP0 Setup 0)–EP0Setup_7 (EP0 Setup 7)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EP0Setup_0 (EP0 setup 0) | EP0Setup_7 (EP0 setup 7)	**0x300c30 | 0x300c37** (8 bits)	D7–0	**EP0Setup_*n*[7:0]**	Endpoint 0 setup data 0 | Endpoint 0 setup data 7	0x0	R	

Eight-byte data received at the endpoint EP0 setup stage are stored from the EP0Setup_0 sequentially.

EP0Setup_0 register

BmRequestType is set.

EP0Setup_1 register

BRequest is set.

EP0Setup_2 register

The lower order 8 bits in Wvalue are set.

EP0Setup_3 register

The upper order 8 bits in Wvalue are set.

EP0Setup_4 register

The lower order 8 bits in WIndex are set.

EP0Setup_5 register

The upper order 8 bits in WIndex are set.

EP0Setup_6 register

The lower order 8 bits in WLength are set.

EP0Setup_7 register

The upper order 8 bits in WLength are set.

USB_Address (USB Address)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
USB_Address (USB address)	0x300c38 (8 bits)	D7	AutoSetAddress	1 Auto set address 0 Do nothing	0	R/W	
		D6–0	USB_Address[6:0]	USB address	0x0	R/W	

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[6:0]

These bits set up the USB address.

The USB address is written automatically by the AutoSetAddress function. Or it can be written.

EP0Control (EP0 Control)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EP0Control (EP0 control)	0x300c39 (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

Sets 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 EP0IntStat register is set to 1.
Refer to the section describing operations, for details.

EP0ControlIN (EP0 Control IN)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EP0ControlIN (EP0 control IN)	0x300c3a (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.

EP0ControlOUT (EP0 Control OUT)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EP0ControlOUT (EP0 control OUT)	0x300c3b (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.

EP0MaxSize (EP0 Max Packet Size)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EP0MaxSize (EP0 max packet size)	0x300c3f (8 bits)	D7	–	–	–	–	0 when being read.
		D6–3	EP0MaxSize[6:3]	Endpoint EP0 max packet size	0x1	R/W	
		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

EPaControl (EPa Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaControl (EPa control)	0x300c40 (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

Sets the toggle sequence bit of the endpoint EPa to 1.

D2 ToggleClr

Sets 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.

EPbControl (EPb Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbControl (EPb control)	0x300c41 (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

Sets the toggle sequence bit of the endpoint EPb to 1.

D2 ToggleClr

Sets 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.

EPcControl (EPc Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPcControl (EPc control)	0x300c42 (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

Sets the toggle sequence bit of the endpoint EPc to 1.

D2 ToggleClr

Sets 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.

EPdControl (EPd Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdControl (EPd control)	**0x300c43** (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

Sets the toggle sequence bit of the endpoint EPd to 1.

D2 ToggleClr

Sets 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.

EPaMaxSize_H (EPa Max Packet Size HIGH)
EPaMaxSize_L (EPa Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPaMaxSize_H (EPa max packet size high)	**0x300c50** (8 bits)	D7–2	–	–	–	–	0 when being read.
		D1–0	**EPaMaxSize[9:8]**	Endpoint EPa max packet size	0x0	R/W	
EPaMaxSize_L (EPa max packet size low)	**0x300c51** (8 bits)	D7–0	**EPaMaxSize[7:0]**	Endpoint EPa max packet size	0x0	R/W	

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.

EPaConfig_0 (EPa Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaConfig_0 (EPa configuration 0)	0x300c52 (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.
		D3–0	EndPointNumber [3:0]	Endpoint number (0x1 to 0xf)				0x0	R/W	

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

Sets the transfer direction of the endpoint.

D6 ToggleMode

Sets 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[3:0]

Sets an endpoint number between 0x1 and 0xf.

EPaConfig_1 (EPa Configuration 1)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaConfig_1 (EPa configuration 1)	0x300c53 (8 bits)	D7	ISO	1	ISO	0	Non-ISO	0	R/W	
		D6	ISO_CRCmode	1	CRC mode	0	Normal ISO	0	R/W	
		D5–0	–	–				–	–	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

Sets the isochronous mode.

D6 ISO_CRCmode

According to USB spec, a packet must be discarded when CRC error occurs in isochronous transaction. When this bit is set, a packet with CRC error is not discarded. This bit is valid when ISO bit (D7) is set.

D[5:0] Reserved

EPbMaxSize_H (EPb Max Packet Size HIGH)
EPbMaxSize_L (EPb Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPbMaxSize_H (EPb max packet size high)	0x300c54 (8 bits)	D7–2	–	–	–	–	0 when being read.
		D1–0	EPbMaxSize[9:8]	Endpoint EPb max packet size	0x0	R/W	
EPbMaxSize_L (EPb max packet size low)	0x300c55 (8 bits)	D7–0	EPbMaxSize[7:0]	Endpoint EPb max packet size	0x0	R/W	

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.

EPbConfig_0 (EPb Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbConfig_0 (EPb configuration 0)	**0x300c56** (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.
		D3–0	**EndPointNumber [3:0]**	Endpoint number (0x1 to 0xf)				0x0	R/W	

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

Sets the transfer direction of the endpoint.

D6 ToggleMode

Sets 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[3:0]

Sets an endpoint number between 0x1 and 0xf.

EPbConfig_1 (EPb Configuration 1)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPbConfig_1 (EPb configuration 1)	**0x300c57** (8 bits)	D7	**ISO**	1	ISO	0	Non-ISO	0	R/W	
		D6	**ISO_CRCmode**	1	CRC mode	0	Normal ISO	0	R/W	
		D5–0	**–**	–				–	–	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

Sets the isochronous mode.

D6 ISO_CRCmode

According to USB spec, a packet must be discarded when CRC error occurs in isochronous transaction. When this bit is set, a packet with CRC error is not discarded. This bit is valid when ISO bit (D7) is set.

D[5:0] Reserved

EPcMaxSize_H (EPc Max Packet Size HIGH)
EPcMaxSize_L (EPc Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcMaxSize_H (EPc max packet size high)	0x300c58 (8 bits)	D7–2	–	–	–	–	0 when being read.
		D1–0	EPcMaxSize[9:8]	Endpoint EPc max packet size	0x0	R/W	
EPcMaxSize_L (EPc max packet size low)	0x300c59 (8 bits)	D7–0	EPcMaxSize[7:0]	Endpoint EPc max packet size	0x0	R/W	

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.

EPcConfig_0 (EPc Configuration 0)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcConfig_0 (EPc configuration 0)	0x300c5a (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.
		D3–0	EndPointNumber [3:0]	Endpoint number (0x1 to 0xf)	0x0	R/W	

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

Sets the transfer direction of the endpoint.

D6 ToggleMode

Sets 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[3:0]

Sets an endpoint number between 0x1 and 0xf.

EPcConfig_1 (EPc Configuration 1)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcConfig_1 (EPc configuration 1)	0x300c5b (8 bits)	D7	ISO	1 ISO / 0 Non-ISO	0	R/W	
		D6	ISO_CRCmode	1 CRC mode / 0 Normal ISO	0	R/W	
		D5–0	–	–	–	–	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

Sets the isochronous mode.

D6 ISO_CRCmode

According to USB spec, a packet must be discarded when CRC error occurs in isochronous transaction. When this bit is set, a packet with CRC error is not discarded. This bit is valid when ISO bit (D7) is set.

D[5:0] Reserved

EPdMaxSize_H (EPd Max Packet Size HIGH)
EPdMaxSize_L (EPd Max Packet Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdMaxSize_H (EPd max packet size high)	0x300c5c (8 bits)	D7–2	–	–	–	–	0 when being read.
		D1–0	EPdMaxSize[9:8]	Endpoint EPd max packet size	0x0	R/W	
EPdMaxSize_L (EPd max packet size low)	0x300c5d (8 bits)	D7–0	EPdMaxSize[7:0]	Endpoint EPd max packet size	0x0	R/W	

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.

EPdConfig_0 (EPd Configuration 0)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdConfig_0 (EPd configuration 0)	0x300c5e (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.
		D3–0	EndPointNumber [3:0]	Endpoint number (0x1 to 0xf)	0x0	R/W	

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

Sets the transfer direction of the endpoint.

D6 ToggleMode

Sets 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[3:0]

Sets an endpoint number between 0x1 and 0xf.

EPdConfig_1 (EPd Configuration 1)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdConfig_1 (EPd configuration 1)	0x300c5f (8 bits)	D7	ISO	1 ISO / 0 Non-ISO	0	R/W	
		D6	ISO_CRCmode	1 CRC mode / 0 Normal ISO	0	R/W	
		D5–0	–	–	–	–	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

Sets the isochronous mode.

D6 ISO_CRCmode

According to USB spec, a packet must be discarded when CRC error occurs in isochronous transaction. When this bit is set, a packet with CRC error is not discarded. This bit is valid when ISO bit (D7) is set.

D[5:0] Reserved

EPaStartAdrs_H (EPa FIFO Start Address HIGH)
EPaStartAdrs_L (EPa FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPaStartAdrs_H (EPa FIFO start address high)	0x300c70 (8 bits)	D7–4	–	–	–	–	0 when being read.
		D3–0	EPaStartAdrs[11:8]	Endpoint EPa start address	0x0	R/W	
EPaStartAdrs_L (EPa FIFO start address low)	0x300c71 (8 bits)	D7–2	EPaStartAdrs[7:2]	Endpoint EPa start address	0x0	R/W	
		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.

EPbStartAdrs_H (EPb FIFO Start Address HIGH)
EPbStartAdrs_L (EPb FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPbStartAdrs_H (EPb FIFO start address high)	0x300c72 (8 bits)	D7–4	–	–	–	–	0 when being read.
		D3–0	EPbStartAdrs[11:8]	Endpoint EPb start address	0x0	R/W	
EPbStartAdrs_L (EPb FIFO start address low)	0x300c73 (8 bits)	D7–2	EPbStartAdrs[7:2]	Endpoint EPb start address	0x0	R/W	
		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.

EPcStartAdrs_H (EPc FIFO Start Address HIGH)
EPcStartAdrs_L (EPc FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPcStartAdrs_H (EPc FIFO start address high)	**0x300c74** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPcStartAdrs[11:8]**	Endpoint EPc start address	0x0	R/W	
EPcStartAdrs_L (EPc FIFO start address low)	**0x300c75** (8 bits)	D7–2	**EPcStartAdrs[7:2]**	Endpoint EPc start address	0x0	R/W	
		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.

EPdStartAdrs_H (EPd FIFO Start Address HIGH)
EPdStartAdrs_L (EPd FIFO Start Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdStartAdrs_H (EPd FIFO start address high)	**0x300c76** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPdStartAdrs[11:8]**	Endpoint EPd start address	0x0	R/W	
EPdStartAdrs_L (EPd FIFO start address low)	**0x300c77** (8 bits)	D7–2	**EPdStartAdrs[7:2]**	Endpoint EPd start address	0x0	R/W	
		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.

CPU_JoinRd (CPU Join FIFO Read)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
CPU_JoinRd (CPU join FIFO read)	**0x300c80** (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 or DMA 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.
If CPU I/F needs to read from the OUT direction endpoint, check the DMA_Running bit of the DMA_Control register to avoid reading data from DMA I/F at the same time.
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.

CPU_JoinWr (CPU Join FIFO Write)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
CPU_JoinWr (CPU join FIFO write)	**0x300c81** (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 or DMA 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.
If CPU I/F needs to write to the IN direction endpoint, check the DMA_Running bit of the DMA_Control register to avoid writing data from DMA I/F at the same time.
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.

EnEPnFIFO_Access (EPn FIFO Access Enable)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EnEPnFIFO_Access (Enable EPn FIFO access)	0x300c82 (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	

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.

EPnFIFOforCPU (EPn FIFO for CPU)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPnFIFOforCPU (EPn FIFO for CPU)	0x300c83 (8 bits)	D7–0	EPnFIFOData[7:0]	Endpoint n FIFO access from CPU	X	R/W	

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.

EPnRdRemain_H (EPn FIFO Read Remain HIGH) EPnRdRemain_L (EPn FIFO Read Remain LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPnRdRemain_H (EPn FIFO read remain high)	**0x300c84** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPnRdRemain[11:8]**	Endpoint n FIFO read remain	0x0	R	
EPnRdRemain_L (EPn FIFO read remain low)	**0x300c85** (8 bits)	D7–0	**EPnRdRemain[7:0]**	Endpoint n FIFO read remain	0x0	R	

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.

EPnWrRemain_H (EPn FIFO Write Remain HIGH) EPnWrRemain_L (EPn FIFO Write Remain LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPnWrRemain_H (EPn FIFO write remain high)	**0x300c86** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPnWrRemain[11:8]**	Endpoint n FIFO write remain	0x0	R	
EPnWrRemain_L (EPn FIFO write remain low)	**0x300c87** (8 bits)	D7–0	**EPnWrRemain[7:0]**	Endpoint n FIFO write remain	0x0	R	

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.

DescAdrs_H (Descriptor Address HIGH) DescAdrs_L (Descriptor Address LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescAdrs_H (Descriptor address high)	**0x300c88** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**DescAdrs[11:8]**	Descriptor address	0x0	R/W	
DescAdrs_L (Descriptor address low)	**0x300c89** (8 bits)	D7–0	**DescAdrs[7:0]**	Descriptor address	0x0	R/W	

DescAdrs[11:0]

Specifies 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.

DescSize_H (Descriptor Size HIGH)
DescSize_L (Descriptor Size LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescSize_H (Descriptor size high)	**0x300c8a** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1–0	**DescSize[9:8]**	Descriptor size	0x0	R/W	
DescSize_L (Descriptor size low)	**0x300c8b** (8 bits)	D7–0	**DescSize[7:0]**	Descriptor size	0x0	R/W	

DescSize[9:0]

Specifies 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.

DescDoor (Descriptor Door)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescDoor (Descriptor door)	**0x300c8f** (8 bits)	D7–0	**DescMode[7:0]**	Descriptor door	0x0	R/W	

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.

DMA_FIFO_Control (DMA FIFO Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_FIFO_ Control (DMA FIFO control)	0x300c90 (8 bits)	D7	FIFO_Running	1	FIFO is running	0	FIFO is not running	0	R	
		D6	AutoEnShort	1	Auto enable short packet	0	Do nothing	0	R/W	
		D5–0	–	–				–	–	0 when being read.

D7 FIFO_Running

Shows that the FIFO of the endpoint connected to the DMA is operating. If the DMA is started, this bit is set to 1. After completing the DMA operation, this bit is set to 0 (to be cleared) when the FIFO becomes empty.

D6 AutoEnShort

When the DMA operation ends and the data smaller than the MaxPacketSize remains in the FIFO, the EnShortPkt bit of that endpoint is set to 1.
This function is valid when the direction of the endpoint connected to the DMA is the IN direction.

D[5:0] Reserved

DMA_Join (DMA Join FIFO)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_Join (DMA join FIFO)	0x300c91 (8 bits)	D7–4	–	–				–	–	0 when being read.
		D3	JoinEPdDMA	1	Join EPd to DMA	0	Do nothing	0	R/W	
		D2	JoinEPcDMA	1	Join EPc to DMA	0	Do nothing	0	R/W	
		D1	JoinEPbDMA	1	Join EPb to DMA	0	Do nothing	0	R/W	
		D0	JoinEPaDMA	1	Join EPa to DMA	0	Do nothing	0	R/W	

The endpoint to perform the DMA transfer can be specified by setting the JoinEPd–aDMA bits. After setting these bits, the remained data quantity for the endpoint of the OUT direction or the space capacity for endpoint of the IN direction can be referred by the DMA_Remain_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.

D[7:4] Reserved

D[3:0] JoinEPdDMA, JoinEPcDMA, JoinEPbDMA, JoinEPaDMA

When this bit is set to 1, the DMA transfer is enabled through the endpoint EP*x* (*x*=a,b,c,d). In addition, reference to the space capacity (for the IN direction) or the data quantity (for the OUT direction) in the FIFO of the endpoint EP*x* (*x*=a,b,c,d) by the DMA_Remain H, L register, is enabled.

DMA_Control (DMA Control)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_Control (DMA control)	0x300c92 (8 bits)	D7	DMA_Running	1	DMA is running	0	DMA is not running	0	R	
		D6	PDREQ	PDREQ signal logic				0	R	
		D5	PDACK	PDACK signal logic				0	R	
		D4	–	–				–	–	0 when being read.
		D3	CounterClr	1	Clear DMA counter	0	Do nothing	0	W	
		D2	–	–				–	–	
		D1	DMA_Stop	1	Finish DMA	0	Do nothing	0	W	
		D0	DMA_Go	1	Start DMA	0	Do nothing	0	W	

This register controls the DMA transfer and shows the status of the interface.

D7 DMA_Running

This bit is automatically set 1 during the DMA transfer. The DMA_Join register cannot be written when this bit is 1.

D6 PDREQ

Shows the logic level of the PDREQ signal for monitoring.

D5 PDACK

Shows the logic level of the PDACK signal for monitoring.

D4 Reserved

D3 CounterClr

When this bit is set to 1, the DMA_Count_HH, HL, LH and LL registers are set to 0x00 (to be cleared). When the DMA_Running bit is 1, writing into this bit is neglected.

D2 Reserved

D1 DMA_Stop

Setting this bit to 1 negates the DMA request (PDREQ) signal. After this bit is set to 1, the DMA_Running bit is set to 0 (to be cleared) and the DMA_Cmp bit of the DMA_IntStat register is set to 1. When restarting the DMA transfer, check the DMA_Running bit or the DMA_Cmp bit, and wait until the DMA operation ends.

Note: Setting this bit to 1 does not stop the DMAC. So to terminate data transfer, first terminate the DMAC (master) and then set this bit to 1.

D0 DMA_Go

Setting this bit to 1 starts the DMA transfer.

DMA_Config_0 (DMA Configuration 0)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_Config_0 (DMA configuration 0)	0x300c94 (8 bits)	D7	ActivePort	1	Activate DMA port	0	Deactivate DMA port	0	R/W	
		D6–4	–	–				–	–	0 when being read.
		D3	PDREQ_Level	1	Active-low	0	Active-high	0	R/W	
		D2	PDACK_Level	1	Active-low	0	Active-high	0	R/W	
		D1	PDRDWR_Level	1	Active-low	0	Active-high	0	R/W	
		D0	–	–				–	–	0 when being read.

This register sets fields on the bus of the DMA interface.

D7 ActivePort

Sets the DMA interface to "active".
When this bit is set to 0, the DMA interface signals become "Hi-Z/Don't care" state.

D[6:4] Reserved

D3 PDREQ_Level

Sets the PDREQ logic level. Set to 0 (active-high).

D2 PDACK_Level

Sets the PDACK logic level. Set to 0 (active-high).

D1 PDRDWR_Level

Sets the logic levels of the PDRD and PDWR signals. Set to 0 (active-high).

D0 Reserved

DMA_Config_1 (DMA Configuration 1)

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
DMA_Config_1 (DMA configuration 1)	0x300c95 (8 bits)	D7	RcvLimitMode	1	Receive limit mode	0	Normal	0	R/W	
		D6–4	–	–				–	–	0 when being read.
		D3	SingleWord	1	Single word	0	Multi word	0	R/W	
		D2–1	–	–				–	–	0 when being read.
		D0	CountMode	1	Count-down mode	0	Free-run mode	0	R/W	

This register sets fields on the operation mode of the DMA interface.

D7 RcvLimitMode

Setting this bit to 1 realizes the RcvLimit mode. This function is available only during write operation for the asynchronous multi-word DMA transfer, and not available in the count down mode.
During the asynchronous DMA write operation in the RcvLimit mode, data up to 16 bytes can be received even after this macro negates the PDREQ signal.
In this mode, the PDREQ signal is negated when the space of the endpoint becomes less than 32 bytes by the DMA write operation. However, when the PDREQ signal is negated, 16-byte data that are not written into the endpoint may exist in the internal circuit. Therefore, the data that can be received after the PDREQ signal is negated, is 16 bytes or less.
In this mode, the PDREQ signal is negated before the endpoint becomes completely full.
When the area of the endpoint set by the EP{a,b,c,d}StartAdrs registers is the same as the value set by the EP{a,b,c,d}MaxSize register (Single Buffer), the endpoint never becomes full. Therefore, the data cannot be transmitted by the IN transfer of the USB.
To avoid this limitation, when using the RcvLimit mode, be sure to enter the value of the EP{a,b,c,d} MaxSize register + 32-byte or larger area, into the EP{a,b,c,d}StartAdrs register.

Note: In the S1C33L27, the USB DMA data transfer count is determined according to the DMAC transfer counter setting. Negating PDREQ by the USB macro does not affect the transfer count. So in RcvLimit mode, the DMAC continues data transfer until the DMAC transfer counter reaches 0 even after the macro negates PDREQ. Therefore, make sure that the DMAC transfer counter is set properly.

D[6:4] Reserved

D3 SingleWord

Sets the handshake mode in the Asynchronous (handshake) mode.
In the Single Word mode, the PDREQ signal is negated every time when one word is transferred.
In the Multi-Word mode, the PDREQ signal is not negated if the next data communication is possible when one word is transferred.

Notes:
- In multi-word DMA transfer mode, the DMAC can only be triggered to start data transfer by the Rising Edge of PDREQ. After that no DMAC trigger will be issued while PDREQ stays active (high level). The subsequent DMAC trigger will be issued at the next PDREQ Rising Edge. Therefore, when using the USB macro in multi-word DMA transfer mode, configure the DMAC in successive transfer mode and set the DMAC transfer counter to the same value set in the DMA_Remain_H and DMA_Remain_L registers.
- In single-word DMA transfer mode, the DMAC can only be triggered to start data transfer by the Rising Edge of PDREQ. The subsequent DMAC trigger will be issued at the next PDREQ Rising Edge. When the DMAC transfer counter reaches 0, DMA transfer will not be started even if a DMAC trigger is issued. Therefore, when using the USB macro in single-word DMA transfer mode, configure the DMAC in single transfer mode and set the DMAC transfer counter to a value equal to or less than that set in the DMA_Remain_H and DMA_ Remain_L registers.

D[2:1] Reserved

D0 CountMode

Sets the mode to control the number of the DMA transmissions.
In the free-run mode, the DMA transfer operation is continued until the DMAC is stopped. The Transfer Byte Counter (DMA_Count_HH, HL, LH, LL) shows the number of transmissions for reference.
In the Count-down mode, the DMA request (PDREQ) signal is asserted up to the number of bytes set in the Transfer Byte Counter (DMA_Count_HH, HL, LH, LL) or until the DMA_Stop is enabled to stop it. The Transfer Byte Counter shows the remained transmission quantity, for reference.

DMA_Latency (DMA Latency)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DMA_Latency (DMA latency)	0x300c97 (8 bits)	D7–4	–	–	–	–	0 when being read.
		D3–0	DMA_Latency[3:0]	Latency	0x0	R/W	

This register sets the Data transfer latency for the transfer in the Asynchronous (handshake) mode.
The unit time of the latency is approximately 130 ns.

D[7:4] Reserved

D[3:0] DMA_Latency[3:0]

If a value between 0x1 and 0xf is written, the PDREQ signal is negated every time when the 4-word is transmitted either in the Single Word mode or in the Multi-Word mode, and the PDREQ signal is not be asserted for (130 × N) ns period.

DMA_Remain_H (DMA FIFO Remain HIGH)
DMA_Remain_L (DMA FIFO Remain LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DMA_Remain_H (DMA FIFO remain high)	0x300c98 (8 bits)	D7–4	–	–	–	–	0 when being read.
		D3–0	DMA_Remain[11:8]	DMA FIFO remain	0x0	R	
DMA_Remain_L (DMA FIFO remain low)	0x300c99 (8 bits)	D7–0	DMA_Remain[7:0]	DMA FIFO remain	0x0	R	

DMA_Remain[11:0]

When the direction of the endpoint connected to the DMA by the DMA_Join register is the OUT direction, this register shows the remained data quantity in the FIFO of the endpoint.
When the direction of the endpoint connected to the DMA by the DMA_Join register is the IN direction, this register shows the space capacity in the FIFO of the endpoint.
The DMA_Remain_H register and the DMA_Remain_L register must be accessed as a pair. When accessing them, access the DMA_Remain_H register first.

DMA_Count_HH (DMA Transfer Byte Counter HIGH/HIGH)
DMA_Count_HL (DMA Transfer Byte Counter HIGH/LOW)
DMA_Count_LH (DMA Transfer Byte Counter LOW/HIGH)
DMA_Count_LL (DMA Transfer Byte Counter LOW/LOW)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DMA_Count_HH (DMA transfer byte counter high/high)	0x300c9c (8 bits)	D7–0	DMA_Count[31:24]	DMA transfer byte counter	0x0	R/W	
DMA_Count_HL (DMA transfer byte counter high/low)	0x300c9d (8 bits)	D7–0	DMA_Count[23:16]	DMA transfer byte counter	0x0	R/W	
DMA_Count_LH (DMA transfer byte counter low/high)	0x300c9e (8 bits)	D7–0	DMA_Count[15:8]	DMA transfer byte counter	0x0	R/W	
DMA_Count_LL (DMA transfer byte counter low/low)	0x300c9f (8 bits)	D7–0	DMA_Count[7:0]	DMA transfer byte counter	0x0	R/W	

DMA_Count[31:0]

These registers specify the data length in the DMA transfer in units of byte, and displays it. Its setting can be done as large as up to 0xffffffff bytes.

When the DMA is set to be in the free run mode by the setting of the CountMode bit of the DMA_Config_1 register (CountMode = 0), values transmitted by the DMA can be referred at any time. In this mode, when the DMA Transfer Byte Counter exceeds 0xffffffff, it returns to 0x00000000 and the DMA_CountUp bit of the DMA_IntStat register is set to 1.
When the DMA is set to be in the countdown mode by the setting of the CountMode bit of the DMA_Config_1 register (CountMode = 1), specify the total number of transmissions in the DMA Transfer Byte Counter, set the DMA_Go bit of the DMA_Control register to 1, and then start the DMA transfer.
In this mode, the DMA Transfer Byte Counter is decreased as much as the data quantity transferred by the DMA. When it reaches 0x00000000, the DMA ends. In this mode, the remained quantity of the data to transfer can be referred. Writing into these registers during the DMA transfer is neglected.
For reading these registers, access the DMA_Count_HH, HL, LH and LL registers in this order.

30 Misc Registers (MISC)

Notes: • The Misc registers at addresses 0x300010–0x300018 and 0x300024 are write-protected. Before the Misc registers can be rewritten, write protection of these registers must be removed by writing data 0x96 to PROT0[7:0]/MISC_PROT0 register (for 0x300010–0x300016) or PROT1[7:0]/MISC_PROT1 register (for 0x300024). Note that since unnecessary rewrites to the Misc registers could lead to erratic system operation, PROT*x*[7:0] should be set to other than 0x96 unless the Misc registers must be rewritten.

• PCLK must be supplied from the CMU to access the Misc registers.
For details of the clock control bit, see the "Clock Management Unit (CMU)" chapter.

30.1 RTC Wait Control

The MISC_RTCWT register contains RTCWT[2:0] to set the number of wait cycles to be inserted when accessing the RTC registers. The number of wait cycles should be set according to the SYSCLK clock frequency.

Table 30.1.1 RTCWT[2:0] (RTC Wait Cycle) Settings

RTCWT[2:0]	Number of wait cycles	SYSCLK frequency
0x7	7 wait cycles	$f_{SYSCLK} \leq 60$ MHz
0x6	6 wait cycles	
0x5	5 wait cycles	
0x4	4 wait cycles	
0x3	3 wait cycles	
0x2	2 wait cycles	
0x1	1 wait cycle	
0x0	0 wait cycles	Cannot be set (Note)

(Default: 0x7)

Note: The S1C33L27 RTC cannot operate if RTCWT[2:0] is set to 0x0 (0 wait cycles).

30.2 Internal RAM Wait Control

The MISC_RAMWT register contains A0WT, A3WT, and CALRWT to set the number of wait cycles (in SYSCLK) to be inserted when accessing Area 0 RAM (32KB IRAM), Area 3 RAM (32KB IVRAM), and calculator RAM (32KB CALCRAM), respectively.
When the control bit is set to 1 (default), one wait cycle will be inserted when the RAM is accessed. When the control bit is set to 0, no wait cycle will be inserted.

Table 30.2.1 A0WT/A3WT/CALRWT (RAM Wait Cycle) Settings

A0WT/A3WT/CALRWT	Number of wait cycles	SYSCLK frequency
1	1 wait cycle	$f_{SYSCLK} \leq 60$ MHz
0	0 wait cycles	

(Default: 1)

In the S1C33L27, the MISC_RAMWT register should be set to 0x0.

Note: The wait cycle is effective only for 32KB IVRAM. Depending on the IVRAM location (IRAM, VRAM, or CALCRAM), the corresponding wait cycle setting will be used. These wait cycle settings do not affect other RAMs.

30.3 USB Settings

30.3.1 USB Wait Control

The MISC_USB register contains USBWT[2:0] to set the number of wait cycles to be inserted when accessing the USB registers. The number of wait cycles should be set according to the SYSCLK clock frequency.

Table 30.3.1.1 USBWT[2:0] (USB Wait Cycle) Settings

USBWT[2:0]	Number of wait cycles	SYSCLK frequency
0x7	7 wait cycles	$f_{SYSCLK} \leq 60$ MHz
0x6	6 wait cycles	$f_{SYSCLK} \leq 56$ MHz
0x5	5 wait cycles	$f_{SYSCLK} \leq 45$ MHz
0x4	4 wait cycles	$f_{SYSCLK} \leq 36$ MHz
0x3	3 wait cycles	$f_{SYSCLK} \leq 24$ MHz
0x2	2 wait cycles	$f_{SYSCLK} \leq 16$ MHz
0x1	1 wait cycle	$f_{SYSCLK} < 8$ MHz
0x0	0 wait cycles	

(Default: 0x7)

30.3.2 Snooze Control

The MISC_USB register contains USBSNZ that controls Snooze mode for the USB function controller. Setting USBSNZ to 1 enables Snooze mode.
Refer to the "USB Function Controller (USB)" chapter for details on control of the USB function controller.

30.3.3 USB Interrupt Enable

The MISC_USB register contains USBINTEN that enables or disables the USB function controller to generate interrupts. Setting USBINTEN to 1 enables USB interrupts; setting to 0 disables USB interrupts.

30.4 Boot Register

The MISC_BOOT register is used to confirm the boot device and configure #CE10 boot conditions.
BOOT[1:0] indicates the boot device that has been specified by the BOOT[1:0] pins.

Table 30.4.1 BOOT[3:0] Bits

BOOT[1:0]	Boot mode
0x3	SPI EEPROM/RS232C boot
0x2	NOR Flash/external ROM boot
0x1	Host Interface boot
0x0	NAND Flash boot

The MISC_BOOT register contains another control bit BOOTEN that is used for the booting process by the internal boot sequencer.

Note: When programming a Flash memory using ICD33, BOOTEN must be set to 0. Be sure to avoid changing the boot mode when writing data to the MISC_BOOT register.

30.5 RAM Configuration

IVRAM and DSTRAM can be configured to a general-purpose RAM or a hardware-specific memory.
RAMCFG[2:0]/MISC_RAMCFG register is used for IVRAM/DSTRAM configuration.

Table 30.5.1 IVRAM/DSTRAM Configuration

RAMCFG[2:0]	IVRAM	DSTRAM
0x7–0x4	CALCRAM (CALC)	DSTRAM (Area 3)
0x3	IRAM (Area 0)	CALCRAM (CALC)
0x2	IRAM (Area 0)	DSTRAM (Area 3)
0x1	VRAM (Area 3)	CALCRAM (CALC)
0x0	VRAM (Area 3)	DSTRAM (Area 3)

(Default: 0x0)

Notes: • When IVRAM is configured as CALCRAM, the first 8KB can only be used for the CALC module.

• A read buffer is provided for IRAM (Area 0) to increase the read speed. After IRAM is relocated to Area 0, a write operation to IRAM is required to flush the read buffer.

• After a RAM is switched to CALCRAM, it can only be accessed by the CALC module. Other bus master (CPU, DMAC, and LCDC) cannot access the CALCRAM.

30.6 Control Register Details

Table 30.6.1 List of Misc Registers

Address	Register name		Function
0x300010	MISC_RTCWT	RTC Wait Control Register	Configure RTC access cycles
0x300012	MISC_USB	USB Configuration Register	Enable USB interrupt, snooze, and configure access cycle
0x300014	MISC_RAMWT	Internal RAM Wait Control Register	Configure internal RAM access cycles
0x300016	MISC_BOOT	Boot Register	Indicate/set boot conditions
0x300018	MISC_TRACE	Debug Trace Enable Register	Enable PC trace debugging
0x300020	MISC_PROT0	Misc Protect Register 0	Enable/disable Misc register write protection
0x300022	MISC_PROT1	Misc Protect Register 1	
0x300024	MISC_RAMCFG	RAM Configuration Register	Configure internal RAM

The Misc registers are described in detail below.

Note: The Misc registers at addresses 0x300010–0x300018 and 0x300024 are write-protected. Before the Misc registers can be rewritten, write protection of these registers must be removed by writing data 0x96 to PROT0[7:0]/MISC_PROT0 register (for 0x300010–0x300018) or PROT1[7:0]/MISC_PROT1 register (for 0x300024). Note that since unnecessary rewrites to the Misc registers could lead to erratic system operation, PROT*x*[7:0] should be set to other than 0x96 unless the Misc registers must be rewritten.

RTC Wait Control Register (MISC_RTCWT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Wait Control Register (MISC_RTCWT)	**0x300010** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**RTCWT[2:0]**	RTC register access wait control	0 to 7 cycles	0x7	R/W	Write-protected

D[7:3] Reserved

D[2:0] RTCWT[2:0]: RTC Register Access Wait Control Bits

Sets the number of wait cycles to be inserted when accessing the RTC control register.

Table 30.6.2 RTCWT[2:0] (RTC Wait Cycle) Settings

RTCWT[2:0]	Number of wait cycles	SYSCLK frequency
0x7	7 wait cycles	fSYSCLK ≤ 60 MHz
0x6	6 wait cycles	
0x5	5 wait cycles	
0x4	4 wait cycles	
0x3	3 wait cycles	
0x2	2 wait cycles	
0x1	1 wait cycle	
0x0	0 wait cycles	Cannot be set (Note)

(Default: 0x7)

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

Note: The S1C33L27 RTC cannot operate if RTCWT[2:0] is set to 0x0 (0 wait cycles).

USB Configuration Register (MISC_USB)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USB Configuration Register (MISC_USB)	0x300012 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	USBINTEN	USB interrupt enable	1 Enable 0 Disable	0	R/W	Write-protected
		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	0 to 7 cycles	0x7	R/W	Write-protected

D7 Reserved

D6 USBINTEN: USB Interrupt Enable Bit

Enables or disables USB interrupts.
1 (R/W): Interrupt enabled
0 (R/W): Interrupt disabled (default)

For more information on the causes of USB interrupts, see the "USB Function Controller (USB)" chapter.

D5 USBSNZ: USB Snooze Control Bit

Enables or disables the USB to enter snooze mode.
1 (R/W): Enabled
0 (R/W): Disabled (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 the "Snooze" section in the "USB Function Controller (USB)" chapter.

D[4:3] Reserved

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

Sets the number of wait cycles to be inserted when accessing the USB control register.

Table 30.6.3 USBWT[2:0] (USB Wait Cycle) Settings

USBWT[2:0]	Number of wait cycles	SYSCLK frequency
0x7	7 wait cycles	$f_{SYSCLK} \leq 60$ MHz
0x6	6 wait cycles	$f_{SYSCLK} \leq 56$ MHz
0x5	5 wait cycles	$f_{SYSCLK} \leq 45$ MHz
0x4	4 wait cycles	$f_{SYSCLK} \leq 36$ MHz
0x3	3 wait cycles	$f_{SYSCLK} \leq 24$ MHz
0x2	2 wait cycles	$f_{SYSCLK} \leq 16$ MHz
0x1	1 wait cycle	$f_{SYSCLK} < 8$ MHz
0x0	0 wait cycles	

(Default: 0x7)

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

Internal RAM Wait Control Register (MISC_RAMWT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Internal RAM Wait Control Register (MISC_RAMWT)	0x300014 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	CALRWT	CALCRAM access wait control	1 1 wait cycle 0 0 wait cycles	1	R/W	Write-protected Should be set to 0.
		D2	–	reserved	–	–	–	Should be fixed at 0.
		D1	A3WT	Area 3 RAM access wait control	1 1 wait cycle 0 0 wait cycles	1	R/W	Write-protected
		D0	A0WT	Area 0 RAM access wait control	1 1 wait cycle 0 0 wait cycles	1	R/W	Should be set to 0.

D[7:4] Reserved

D3 CALRWT: CALCRAM Access Wait Control Bit

Sets the number of wait cycles to be inserted when accessing CALCRAM (only when IVRAM is switched to CALCRAM).
1 (R/W): 1 cycle (default)
0 (R/W): 0 cycles (no wait inserted)

D2 Reserved

D1 A3WT: Area 3 RAM Access Wait Control Bit

Sets the number of wait cycles to be inserted when accessing Area 3 IVRAM (only when IVRAM is located in Area 3).

1 (R/W): 1 cycle (default)

0 (R/W): 0 cycles (no wait inserted)

D0 A0WT: Area 0 RAM Access Wait Control Bit

Sets the number of wait cycles to be inserted when accessing Area 0 IRAM (only when IVRAM is switched to IRAM in Area 0).

1 (R/W): 1 cycle (default)

0 (R/W): 0 cycles (no wait inserted)

Table 30.6.4 A0WT/A3WT/CALRWT (RAM Wait Cycle) Settings

A0WT/A3WT/CALRWT	Number of wait cycles	SYSCLK frequency
1	1 wait cycle	$f_{SYSCLK} \leq 60$ MHz
0	0 wait cycles	

(Default: 1)

In the S1C33L27, the MISC_RAMWT register should be set to 0x0.

Boot Register (MISC_BOOT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Boot Register (MISC_BOOT)	0x300016 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5–4	BOOT[1:0]	Boot mode indicator	BOOT[1:0] / Boot mode: 0x3 SPI/RS232C; 0x2 NOR/ROM; 0x1 HIF; 0x0 NAND	*	R	* Depends on the BOOT pin status at initial reset
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	BOOTEN	#CE10 area boot enable	1 Internal / 0 External	1	R/W	Write-protected
		D0	–	reserved	–	–	–	0 when being read.

D[7:6] Reserved

D[5:4] BOOT[1:0]: Boot Mode Indicator Bits

Indicates the boot device that has been specified by the BOOT[1:0] pins.

Table 30.6.5 BOOT[1:0] Bits

BOOT[1:0]	Boot mode
0x3	SPI EEPROM/RS232C boot
0x2	NOR Flash/external ROM boot
0x1	Host Interface boot
0x0	NAND Flash boot

D[3:2] Reserved

D1 BOOTEN: #CE10 Area Boot Enable Bit

Enables fetching the RESET vector from the #CE10 external area (0xc00000).

1 (R/W): Internal boot (default)

0 (R/W): External boot

BOOTEN is set in the system boot sequencer. When programming a Flash memory using ICD33, BOOTEN must be set to 0.

D0 Reserved

Debug Trace Enable Register (MISC_TRACE)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Debug Trace Enable Register (MISC_TRACE)	0x300018 (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	TRACE	PC trace debugging enable	1 Disable / 0 Enable	0	R/W	Write-protected

D[7:1] Reserved

D0 TRACE: PC Trace Debugging Enable Bit

Enables or disables the PC trace function in debug mode.
1 (R/W): Disabled
0 (R/W): Enabled (default)

When TRACE is set to 0, the PC trace function and the debug pins (DST0, DST1, DPCO) are enabled. When TRACE is set to 1, these pins can be configured as GPIO or other peripheral pins using the port function select bits.

Misc Protect Register 0 (MISC_PROT0)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Misc Protect Register 0 (MISC_ PROT0)	**0x300020** (8 bits)	D7–0	**PROT0[7:0]**	Misc register write-protect flag	Writing 10010110 (0x96) removes the write protection of the Misc registers (0x300010–0x300018). Writing another value set the write protection.	0x0	R/W	

D[7:0] PROT0[7:0]: Misc Register Write-Protect Flag Bits

Enables or disables write protection of the Misc registers (0x300010–0x300018).
0x96 (R/W): Disable write protection
Other than 0x96 (R/W): Write-protect the register (default: 0x0)

Before altering any Misc register from 0x300010 to 0x300018, write data 0x96 to PROT0[7:0] to disable write protection. If PROT0[7:0] is set to other than 0x96, even if an attempt is made to alter any Misc register by executing a write instruction, the content of the register will not be altered even though the instruction may have been executed without an error. Once PROT0[7:0] is set to 0x96, the Misc registers can be rewritten any number of times until being reset to other than 0x96. When rewriting the Misc registers has finished, PROT0[7:0] should be set to other than 0x96 to prevent accidental writing to the Misc registers.

Misc Protect Register 1 (MISC_PROT1)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Misc Protect Register 1 (MISC_ PROT1)	**0x300022** (8 bits)	D7–0	**PROT1[7:0]**	Misc register write-protect flag	Writing 10010110 (0x96) removes the write protection of the Misc register (0x300024). Writing another value set the write protection.	0x0	R/W	

D[7:0] PROT1[7:0]: Misc Register Write-Protect Flag Bits

Enables or disables write protection of the MISC_RAMCFG register (0x300024).
0x96 (R/W): Disable write protection
Other than 0x96 (R/W): Write-protect the register (default: 0x0)

Before altering the MISC_RAMCFG register (0x300024), write data 0x96 to PROT1[7:0] to disable write protection. If PROT1[7:0] is set to other than 0x96, even if an attempt is made to alter the MISC_ RAMCFG register by executing a write instruction, the content of the register will not be altered even though the instruction may have been executed without an error. Once PROT1[7:0] is set to 0x96, the MISC_RAMCFG register can be rewritten any number of times until being reset to other than 0x96. When rewriting the MISC_RAMCFG register has finished, PROT1[7:0] should be set to other than 0x96 to prevent accidental writing to the MISC_RAMCFG register.

RAM Configuration Register (MISC_RAMCFG)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
RAM Configuration Register (MISC_ RAMCFG)	**0x300024** (8 bits)	D7–3	**–**	reserved	–		–	–	0 when being read.
		D2–0	**RAMCFG [2:0]**	DSTRAM/IVRAM configuration	RAMCFG[2:0]	DSTRAM	0x0	R/W	Write-protected
					0x7–0x4	DSTRAM			
					0x3	CALCRAM			
					0x2	DSTRAM			
					0x1	CALCRAM			
					0x0	DSTRAM			
					RAMCFG[2:0]	IVRAM			
					0x7–0x4	CALCRAM			
					0x3	IRAM			
					0x2	IRAM			
					0x1	VRAM			
					0x0	VRAM			

D[7:3] Reserved

D[2:0] RAMCFG[2:0]: DSTRAM/IVRAM Configuration Bits

Configures DSTRAM and IVRAM located in Area 3 by default.

Table 30.6.6 IVRAM/DSTRAM Configuration

RAMCFG[2:0]	IVRAM	DSTRAM
0x7–0x4	CALCRAM (CALC)	DSTRAM (Area 3)
0x3	IRAM (Area 0)	CALCRAM (CALC)
0x2	IRAM (Area 0)	DSTRAM (Area 3)
0x1	VRAM (Area 3)	CALCRAM (CALC)
0x0	VRAM (Area 3)	DSTRAM (Area 3)

(Default: 0x0)

Notes: • When IVRAM is configured as CALCRAM, the first 8KB can only be used for the CALC module.

- A read buffer is provided for IRAM (Area 0) to increase the read speed. After IRAM is relocated to Area 0, a write operation to IRAM is required to flush the read buffer.
- After a RAM is switched to CALCRAM, it can only be accessed by the CALC module. Other bus master (CPU, DMAC, and LCDC) cannot access the CALCRAM.

31 Calculation Module (CALC)

31.1 CALC Module Overview

The S1C33L27 is equipped with a calculation (CALC) module to reduce MP3 decoding and multimedia processing loads. The CALC module provides an API for various calculations.
The following shows the calculation functions provided by the CALC module:

- Multiplications
- Multiply-accumulate (MAC) operations
- Subtract-multiply operations
- Matrix multiplications
- Affine transformations
- Butterfly operations

31.2 Resources

31.2.1 Reserved Resources

The CALC module uses the resources shown below.

- CALCRAM (IVRAM or DSTRAM, see Section 31.2.2.)
- Stack 4 bytes / General-purpose registers (r4–r12)

31.2.2 Calculator RAM (CALCRAM)

To execute a multiply-accumulate operation in one cycle, the multiplicand and multiplier must be stored in different memories that can be accessed simultaneously with Harvard architecture. One of the arguments should be stored in the internal memory, which is configured as the calculator RAM (CALCRAM), while the other is stored in other memory. Either DSTRAM or IVRAM can be selected as CALCRAM using RAMCFG[2:0]/MISC_RAMCFG register. For more information on the MISC_RAMCFG register, see the "Misc Registers (MISC)" chapter.

Table 31.2.2.1 CALCRAM Selection

RAMCFG[2:0]	IVRAM	DSTRAM
0x7–0x4	CALCRAM (CALC)	DSTRAM (Area 3)
0x3	IRAM (Area 0)	CALCRAM (CALC)
0x2	IRAM (Area 0)	DSTRAM (Area 3)
0x1	VRAM (Area 3)	CALCRAM (CALC)
0x0	VRAM (Area 3)	DSTRAM (Area 3)

31.3 Description of CALC Functions

31.3.1 List of Functions

Table 31.3.1.1 List of CALC Functions

Function name	Description
CalcSetMode	Set numerical format
CalcInit	Set initial value
CalcMul1	Execute multiplication
CalcMul4	Execute multiplication 4 times
CalcMul6	Execute multiplication 6 times
CalcMul8	Execute multiplication 8 times
CalcMul16	Execute multiplication 16 times
CalcMac2_6	Execute 2-element multiply-accumulate operation 6 times
CalcMac4	Execute 4-element multiply-accumulate operation

Function name	Description
CalcMac6	Execute 6-element multiply-accumulate operation
CalcMac8	Execute 8-element multiply-accumulate operation
CalcMac16	Execute 16-element multiply-accumulate operation
CalcMacCircular8	Execute 8-element multiply-accumulate operation with circular pointer
CalcSubMul1	Execute 3-element subtract-multiply operation
CalcSubMul8	Execute 3-element subtract-multiply operation 8 times
CalcMatrix2	Execute second-order matrix multiplication ((2 × 2) × (2 × 1))
CalcMatrix3	Execute third-order matrix multiplication ((3 × 3) × (3 × 1))
CalcMatrix4	Execute fourth-order matrix multiplication ((4 × 4) × (4 × 1))
CalcAffine2	Execute second-order affine transformation ((2 × 2) × (2 × 1) + (2 × 1))
CalcAffine3	Execute third-order affine transformation ((3 × 3) × (3 × 1) + (3 × 1))
CalcButterfly	Execute butterfly operation
CalcButterflyConst	Execute butterfly operation by constant
CalcButterfly2_1	Execute butterfly operation
CalcButterfly3_1	Execute 1-pair butterfly operation
CalcButterfly3_2	Execute 2-pair butterfly operation
CalcButterfly3_4	Execute 4-pair butterfly operation
CalcButterfly3_8	Execute 8-pair butterfly operation
CalcButterfly3_16	Execute 16-pair butterfly operation

Notes: • The CALC functions should be accessed exclusively. For example, when an interrupt is occurred during executing a CALC function, another CALC function cannot be called in the interrupt handler.

- One of the multiplication arguments must be stored in CALCRAM (IVRAM or DSTRAM) before using the CALC functions. CALCRAM should be configured using a MISC register in the application program.
- Before first use of the CALC functions, the CALC module must be initialized using the “CalcInit” function.
- If an input value or a result exceeds the effective range according to the specified calculation mode while a CALC function is being executed, incorrect output value will results. Use numerical values that do not exceed the range.

31.3.2 Numerical Formats

This module can handle numbers in the following numerical formats. The numerical format to be used should be specified using the CalcSetMode function. Since each calculation function defines numbers as long integers, apply the appropriate data type conversion before actual use.

32-bit unsigned integer (CALC_UINT32)

The module can handle 32-bit unsigned integers as ordinary unsigned long integers. However, since it does not perform saturation processing, if values during calculations or calculation results fall outside the range of 0 to 4294967295, the result will be incorrect. Use numerical values that do not exceed this range.

32-bit signed integer (CALC_SINT32)

The module can handle 32-bit signed integers as ordinary signed long integers. However, since it does not perform saturation processing, if the value falls outside the range of -2147483648 to 2147483647 during calculations, the result will be incorrect. Use numerical values that do not exceed this range.

Q31 notation 32-bit signed fixed-point (CALC_FIXED32_Q31)

The module can handle Q31 notation 32-bit signed fixed-point numbers with a 1-bit integer part and a 31-bit decimal part. The two's complement representation is used to express the sign.

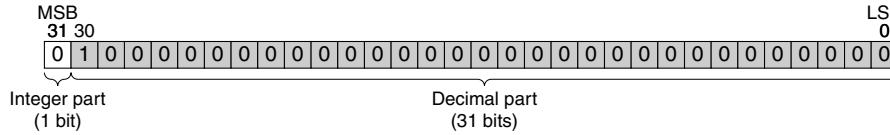


Figure 31.3.2.1 Q31 Notation 32-bit Signed Fixed-Point Format

The Q31 notation 32-bit signed fixed-point number is obtained by multiplying the original number by 2147483648 and rounding.
Examples: 0.5 → 0x40000000
-0.25 → 0xe0000000

However, since it does not perform saturation processing, if the value falls outside the range of -1 to 0.9999999995343387 during calculations, the result will be incorrect. Use numerical values that do not exceed this range.

Q12 notation 16-bit signed fixed-point (CALC_FIXED16_Q12)

The module can handle Q28 notation 32-bit signed fixed-point numbers with a 4-bit integer part and a 28-bit decimal part. It converts these numbers into Q12 notation 16-bit signed fixed-point numbers with a 4-bit integer part and a 12-bit decimal part. The two's complement representation is used to express the sign.
This makes it possible to obtain higher precision results than normal 16-bit computations.

[Input]

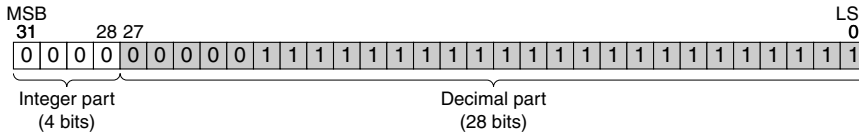


Figure 31.3.2.2 Q28 Notation 32-bit Signed Fixed-Point Format

The Q28 notation 32-bit signed fixed-point number is obtained by multiplying the original number by 268435456 and rounding.
Examples: 0.0312499962 → 0x007fffff
-0.0312499962→ 0xff800001

[Output]

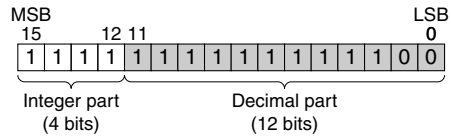


Figure 31.3.2.3 Q12 Notation 16-bit Signed Fixed-Point Format

The Q12 notation 16-bit signed fixed-point number is obtained by multiplying the original number by 4096 and rounding.
Example: -0.000976562 → 0xfffffffc

Since the saturation processing involves numbers in this numeric format, values greater than 7.999755859375 (0x7fff) will be clipped to 7.999755859375 (0x7fff). Values smaller than -8 (0x8000) will be clipped to -8 (0x8000).

The module always handles numbers as 32-bit data. Note that the upper 16 bits of output results are used to express the sign.

Q13 notation 16-bit signed fixed-point (CALC_FIXED16_Q13)

The module can handle Q13 notation 16-bit signed fixed-point numbers with a 3-bit integer part and a 13-bit decimal part. The two's complement representation is used to express the sign.

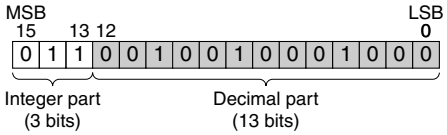


Figure 31.3.2.4 Q13 Notation 16-bit Signed Fixed-Point Format

The Q13 notation 16-bit signed fixed-point number is obtained by multiplying the original number by 8192 and rounding.

Examples: 3.1416015625 → 0x00006488

-2.71826171875 → 0xffffa904

Since the saturation processing involves numbers in this numeric format, values greater than 3.9998779296875 (0x7fff) will be clipped to 3.9998779296875 (0x7fff). Values smaller than -4 (0x8000) will be clipped to -4 (0x8000).

The module always handles numbers as 32-bit data. Note that the upper 16 bits are used to express the sign.

Q14 notation 16-bit signed fixed-point (CALC_FIXED16_Q14)

The module can handle Q14 notation 16-bit signed fixed-point numbers with a 2-bit integer part and a 14-bit decimal part. The two's complement representation is used to express the sign.

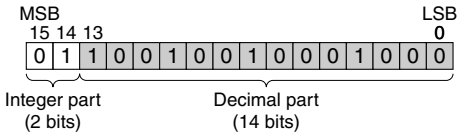


Figure 31.3.2.5 Q14 Notation 16-bit Signed Fixed-Point Format

The Q14 notation 16-bit signed fixed-point number is obtained by multiplying the original number by 16384 and rounding.

Examples: 1.57080078125 → 0x00006488

-1.359130859375 → 0xffffa904

Since the saturation processing involves numbers in this numeric format, values greater than 1.99993896484375 (0x7fff) will be clipped to 1.99993896484375 (0x7fff). Values smaller than -2 (0x8000) will be clipped to -2 (0x8000).

The module always handles numbers as 32-bit data. Note that the upper 16 bits are used to express the sign.

31.3.3 CalcSetMode

Function	Set numerical format		
Format	void CalcSetMode(long Mode);		
Argument	Mode	Numerical format	(Input)
Return value	None		
Description	The CalcSetMode function sets the numerical formats to be used. "Mode" is used to select a numerical mode from six types shown below.		

Table 31.3.3.1 Numerical Mode Selections

Argument "Mode" *	Numerical Mode
CALC_UINT32	32-bit unsigned integer
CALC_SINT32	32-bit signed integer
CALC_FIXED32_Q31	Q31 notation 32-bit signed fixed-point
CALC_FIXED16_Q12	Q12 notation 16-bit signed fixed-point
CALC_FIXED16_Q13	Q13 notation 16-bit signed fixed-point
CALC_FIXED16_Q14	Q14 notation 16-bit signed fixed-point

* These constants (CALC_*x*) are defined in the header file "calc.h."

Description	Call this function before using the CALC module or to change the numerical format. Since the CALC functions handle numbers in long integer, an appropriate data type conversion is required before being used.
Note	If an input value or a result exceeds the effective range set by the CalcSetMode function while a CALC function is being executed, incorrect output value will results. Use numerical values that do not exceed the range.

31.3.4 CalcInit

Function	Set initial value		
Format	void CalcInit(long Value);		
Argument	Value	Initial value	(Input)
Return value	None		
Description	The CalcInit function sets the specified initial value.		
Note	The CalcInit function must be called at the first time a CALC function is used after a power-on. After that the result value will be cleared at the end of each function.		

31.3.5 CalcMul1

Function	Execute multiplication		
Format	long CalcMul1(long *C, long X);		
Argument	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Multiplier	(Input)
Return value	Calculation results (scalar)		
Description	The CalcMul1 function performs multiplication once, as expressed by the following equation: $R = C[0] * X$ Prepare argument C as a long-type 1-element array regardless of the numerical format specified. Arguments C and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.6 CalcMul4

Function	Execute multiplication 4 times		
Format	void CalcMul4(long *Y, long *C, long* X);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	None		
Description	The CalcMul4 function performs multiplication 4 times, as expressed by the following equation: $Y[i] = (C[i] * X[i])$ for i = 0, 1, 2, 3 Prepare arguments Y, C, and X as long-type 4-element arrays regardless of the numerical format specified. Arguments Y, C, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.7 CalcMul6

Function	Execute multiplication 6 times
Format	void CalcMul6(long *Y, long *C, long* X);

Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	None		
Description	The CalcMul6 function performs multiplication 6 times, as expressed by the following equation: Y[i] = (C[i] ∗ X[i]) for i = 0, 1, 2, 3, 4, 5 Prepare arguments Y, C, and X as long-type 6-element arrays regardless of the numerical format specified. Arguments Y, C, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.8 CalcMul8

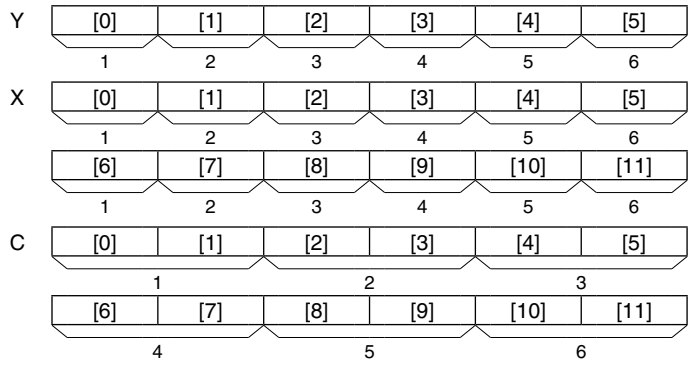
Function	Execute multiplication 8 times		
Format	void CalcMul8(long *Y, long *C, long* X);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	None		
Description	The CalcMul8 function performs multiplication 8 times, as expressed by the following equation: Y[i] = (C[i] ∗ X[i]) for i = 0, 1, 2, 3, ... , 7 Prepare arguments Y, C, and X as long-type 8-element arrays regardless of the numerical format specified. Arguments Y, C, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.9 CalcMul16

Function	Execute multiplication 16 times		
Format	void CalcMul16(long *Y, long *C, long* X);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	None		
Description	The CalcMul16 function performs multiplication 16 times, as expressed by the following equation: Y[i] = (C[i] ∗ X[i]) for i = 0, 1, 2, 3, ... , 15 Prepare arguments Y, C, and X as long-type 16-element arrays regardless of the numerical format specified. Arguments Y, C, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.10 CalcMac2_6

Function	Execute 2-element multiply-accumulate operation 6 times		
Format	void CalcMac2_6(long *Y, long *C, long *X);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	None		

Description	The CalcMac2_6 function performs 2-element multiply-accumulate operations 6 times, as expressed by the following equation: Y[0] = C[0] * X[0] + C[1] * X[6] Y[1] = C[2] * X[1] + C[3] * X[7] Y[2] = C[4] * X[2] + C[5] * X[8] Y[3] = C[6] * X[3] + C[7] * X[9] Y[4] = C[8] * X[4] + C[9] * X[10] Y[5] = C[10] * X[5] + C[11] * X[11] Prepare arguments C and X as long-type 12-element arrays, and Y as a long-type 6-element array, regardless of the numerical format specified. Arguments Y, C, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM). Data layout 

31.3.11 CalcMac4

Function	Execute 4-element multiply-accumulate operation		
Format	long CalcMac4(long *C, long *X);		
Argument	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	Calculation results (scalar)		
Description	The CalcMac4 function performs 4-element multiply-accumulate operations, as expressed by the following equation: $R = \sum_{i=0}^{3}(C[i] * X[i])$ Prepare arguments C and X as long-type 4-element arrays regardless of the numerical format specified. Arguments C and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.12 CalcMac6

Function	Execute 6-element multiply-accumulate operation		
Format	long CalcMac6(long *C, long *X);		
Argument	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	Calculation results (scalar)		

Description	The CalcMac6 function performs 6-element multiply-accumulate operations, as expressed by the following equation: $R = \sum_{i=0}^{5}(C[i] * X[i])$ Prepare arguments C and X as long-type 6-element arrays regardless of the numerical format specified. Arguments C and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

31.3.13 CalcMac8

Function	Execute 8-element multiply-accumulate operation		
Format	long CalcMac8(long *C, long *X);		
Argument	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	Calculation results (scalar)		
Description	The CalcMac8 function performs 8-element multiply-accumulate operations, as expressed by the following equation: $R = \sum_{i=0}^{7}(C[i] * X[i])$ Prepare arguments C and X as long-type 8-element arrays regardless of the numerical format specified. Arguments C and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.14 CalcMac16

Function	Execute 16-element multiply-accumulate operation		
Format	long CalcMac16(long *C, long *X);		
Argument	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	Calculation results (scalar)		
Description	The CalcMac16 function performs 16-element multiply-accumulate operations, as expressed by the following equation: $R = \sum_{i=0}^{15}(C[i] * X[i])$ Prepare arguments C and X as long-type 16-element arrays regardless of the numerical format specified. Arguments C and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.15 CalcMacCircular8

Function	Execute 8-element multiply-accumulate operation with circular pointer		
Format	long CalcMacCircular8(long *C, long *X);		
Argument	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
Return value	Calculation results (scalar)		
Description	The CalcMacCircular8 function performs 8-element multiply-accumulate operations, as expressed by the following equation: $R = \sum_{i=0}^{7}(C[i] * X[ptr])$		

Description

The 8-element array specified by argument X is manipulated as a circular buffer with a software pointer. The elements are pointed in the sequence shown below.

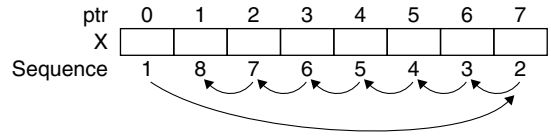


Figure 31.3.15.1 Circular Buffer Addressing

Prepare arguments C and X as long-type 8-element arrays regardless of the numerical format specified. Arguments C and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

31.3.16 CalcSubMul1

Function	Execute 3-element subtract-multiply operation		
Format	long CalcSubMul1(long A, long B, long *C);		
Argument	C	Pointer to the multiplier (CALCRAM)	(Input)
	A	Minuend	(Input)
	B	Subtrahend	(Input)
Return value	Calculation results (scalar)		
Description	The CalcSubMul1 function evaluates the following equation: R = (A - B) ∗ C[0] Prepare argument C as a long-type 1-element array regardless of the numerical format specified. Arguments C, A, and B must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.17 CalcSubMul8

Function	Execute 3-element subtract-multiply operation 8 times		
Format	void CalcSubMul8(long *Y, long *A, long *B, long *C);		
Argument	Y	Pointer to the calculation results	(Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
	A	Pointer to the minuend vector	(Input)
	B	Pointer to the subtrahend vector	(Input)
Return value	None		
Description	The CalcSubMul8 function evaluates the following equation: Y[i] = (A[i] - B[-i]) ∗ C[i] for i = 0, 1, 2, 3, ... , 7 Prepare arguments Y, C, A, and B as long-type 8-element arrays regardless of the numerical format specified. Arguments Y, C, A, and B must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM). Data layout		

Y	[0]	[1]	[2]	[3]	. . .	[6]	[7]
A	[0]	[1]	[2]	[3]	. . .	[6]	[7]
B	[-7]	[-6]	[-5]	[-4]	. . .	[-1]	[-0]
C	[0]	[1]	[2]	[3]	. . .	[6]	[7]

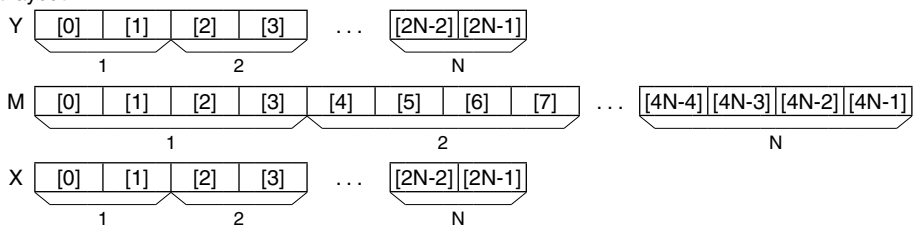
31.3.18 CalcMatrix2

Function	Execute second-order matrix multiplication N times		
Format	void CalcMatrix2(long *Y, long *M, long* X, unsigned long N);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	M	Pointer to the multiplicand matrix (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		
Description	The CalcMatrix2 function multiplies a second-order matrix by a second-order vector, as expressed by the following equation:		

$$\begin{bmatrix} Y[0] \\ Y[1] \end{bmatrix} = \begin{bmatrix} M[0] & M[1] \\ M[2] & M[3] \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \end{bmatrix}$$

Prepare arguments Y, M, and X as long-type arrays arranged continuously as shown below. Arguments Y, M, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument M must be located in CALCRAM (IVRAM or DSTRAM).

Data layout



31.3.19 CalcMatrix3

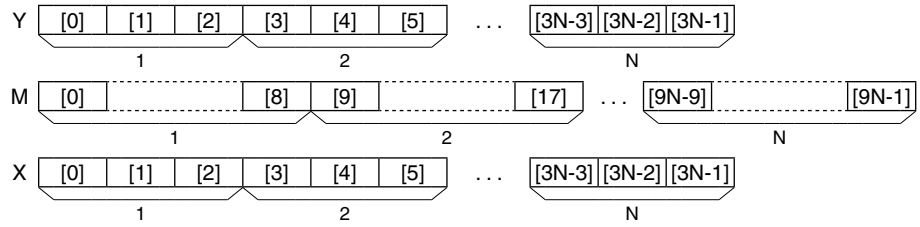
Function	Execute third-order matrix multiplication N times		
Format	void CalcMatrix3(long *Y, long *M, long* X, unsigned long N);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	M	Pointer to the multiplicand matrix (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		
Description	The CalcMatrix3 function multiplies a third-order matrix by a third-order vector, as expressed by the following equation:		

$$\begin{bmatrix} Y[0] \\ Y[1] \\ Y[2] \end{bmatrix} = \begin{bmatrix} M[0] & M[1] & M[2] \\ M[3] & M[4] & M[5] \\ M[6] & M[7] & M[8] \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \\ X[2] \end{bmatrix}$$

Prepare arguments Y, M, and X as long-type arrays arranged continuously as shown below. Arguments Y, M, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument M must be located in CALCRAM (IVRAM or DSTRAM).

Description

Data layout



31.3.20 CalcMatrix4

Function	Execute fourth-order matrix multiplication N times		
Format	void CalcMatrix4(long *Y, long *M, long* X, unsigned long N);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	M	Pointer to the multiplicand matrix (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		

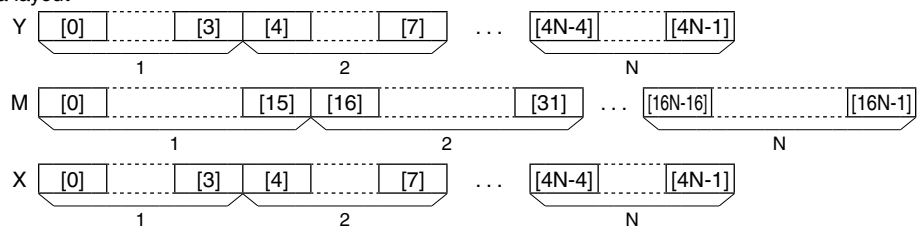
Description

The CalcMatrix4 function multiplies a fourth-order matrix by a fourth-order vector, as expressed by the following equation:

$$\begin{bmatrix} Y[0] \\ Y[1] \\ Y[2] \\ Y[3] \end{bmatrix} = \begin{bmatrix} M[0] & M[1] & M[2] & M[3] \\ M[4] & M[5] & M[6] & M[7] \\ M[8] & M[9] & M[10] & M[11] \\ M[12] & M[13] & M[14] & M[15] \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix}$$

Prepare arguments Y, M, and X as long-type arrays arranged continuously as shown below. Arguments Y, M, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument M must be located in CALCRAM (IVRAM or DSTRAM).

Data layout



31.3.21 CalcAffine2

Function	Execute second-order affine transformation N times		
Format	void CalcAffine2(long *Y, long *M, long* X, long* A, unsigned long N);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	M	Pointer to the multiplicand matrix (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
	A	Pointer to the addend vector	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		

Description	The CalcAffine2 function multiplies a second-order matrix by a second-order vector and adds a second-order vector to the product, as expressed by the following equation: $\begin{bmatrix} Y[0] \\ Y[1] \end{bmatrix} = \begin{bmatrix} M[0] & M[1] \\ M[2] & M[3] \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \end{bmatrix} + \begin{bmatrix} A[0] \\ A[1] \end{bmatrix}$ Prepare arguments Y, M, X, and A as long-type arrays arranged continuously. Arguments Y, M, X, and A must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument M must be located in CALCRAM (IVRAM or DSTRAM).

31.3.22 CalcAffine3

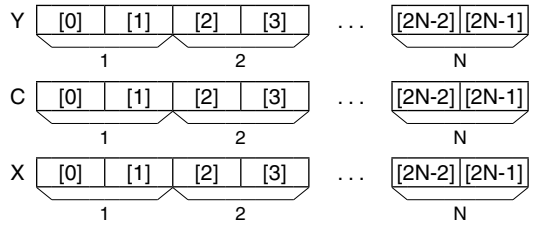
Function	Execute third-order affine transformation N times		
Format	void CalcAffine3(long *Y, long *M, long* X, long* A, unsigned long N);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	M	Pointer to the multiplicand matrix (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
	A	Pointer to the addend vector	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		
Description	The CalcAffine3 function multiplies a third-order matrix by a third-order vector and adds a third-order vector to the product, as expressed by the following equation: $\begin{bmatrix} Y[0] \\ Y[1] \\ Y[2] \end{bmatrix} = \begin{bmatrix} M[0] & M[1] & M[2] \\ M[3] & M[4] & M[5] \\ M[6] & M[7] & M[8] \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \\ X[2] \end{bmatrix} + \begin{bmatrix} A[0] \\ A[1] \\ A[2] \end{bmatrix}$ Prepare arguments Y, M, X, and A as long-type arrays arranged continuously. Arguments Y, M, X, and A must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument M must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.23 CalcButterfly

Function	Execute butterfly operation N times		
Format	void CalcButterfly(long *Y, long *C, long* X, unsigned long N);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplicand vector (CALCRAM)	(Input)
	X	Pointer to the multiplier vector	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		
Description	The CalcButterfly function performs matrix multiplication using the second-order vectors specified, as expressed by the following equation: $\begin{bmatrix} Y[0] \\ Y[1] \end{bmatrix} = \begin{bmatrix} C[0] & C[1] \\ -C[1] & C[0] \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \end{bmatrix}$ Prepare arguments Y, C, and X as long-type arrays arranged continuously as shown below. Arguments Y, C, and X must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

Description

Data layout



31.3.24 CalcButterflyConst

Function	Execute butterfly operation by constant N times		
Format	void CalcButterflyConst(long *Y1, long *Y2, long* C, unsigned long N);		
Argument	Y1	Pointer to the multiplicand/calculation results	(Input/Output)
	Y2	Pointer to the multiplicand/calculation results	(Input/Output)
	C	Pointer to the multiplier constant (CALCRAM)	(Input)
	N	Number of operations to be executed	(Input)
Return value	None		

Description

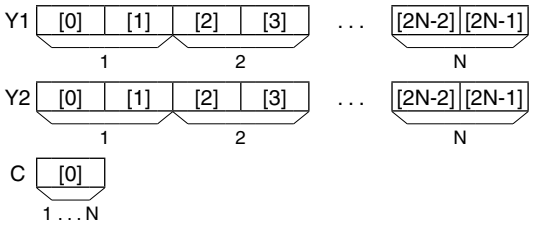
The CalcButterflyConst function performs butterfly operation using the second-order vectors specified, as expressed by the following equation:

A = Y1[0]
B = Y2[0]

$$\begin{bmatrix} Y1[0] \\ Y2[0] \end{bmatrix} = \begin{bmatrix} A & B \\ -B & A \end{bmatrix} \begin{bmatrix} C[0] \\ C[0] \end{bmatrix}$$

Prepare arguments Y1 and Y2 as long-type arrays arranged continuously as shown below, and C as a long-type 1-element array. Arguments Y1, Y2, and C must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

Data layout



31.3.25 CalcButterfly2_1

Function	Execute butterfly operation		
Format	void CalcButterfly2_1(long *Y1, long *Y2, long* C);		
Argument	Y1	Pointer to the multiplicand/calculation results	(Input/Output)
	Y2	Pointer to the multiplicand/calculation results	(Input/Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
Return value	None		

Description	The CalcButterfly2_1 function performs butterfly operation using the second-order vectors specified, as expressed by the following equation: A = Y1[0] B = Y2[0] $\begin{bmatrix} Y1[0] \\ Y2[0] \end{bmatrix} = \begin{bmatrix} A & B \\ B & -A \end{bmatrix} \begin{bmatrix} C[0] \\ C[1] \end{bmatrix}$ Prepare arguments Y1 and Y2 as long-type 1-element arrays, and C as a long-type 2-element array. Arguments Y1, Y2, and C must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

31.3.26 CalcButterfly3_1

Function	Execute 1-pair butterfly operation		
Format	void CalcButterfly3_1(long *Y, long *C, long A, long B);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
	A	Augend/minuend	(Input)
	B	Addend/subtrahend	(Input)
Return value	None		
Description	The CalcButterfly3_1 function performs butterfly operation as expressed by the following equation: Y[0] = A + B Y[1] = (A - B) ∗ C[0] Prepare argument Y as a long-type 2-element array and C as a long-type 1-element array regardless of the numerical format specified. Arguments Y, C, A, and B must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).		

31.3.27 CalcButterfly3_2

Function	Execute 2-pair butterfly operation		
Format	void CalcButterfly3_2(long *Y, long *C, long* A);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
	A	Pointer to the augend/minuend/addend/subtrahend vector	(Input)
Return value	None		
Description	The CalcButterfly3_2 function performs butterfly operation as expressed by the following equation: Y[0] = A[0] + A[3] Y[1] = A[1] + A[2] Y[2] = (A[0] - A[3]) ∗ C[0] Y[3] = (A[1] - A[2]) ∗ C[1] Prepare arguments Y, A, and C as long-type arrays arranged continuously as shown below. Arguments Y, A, and C must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM). Data layout Y [0] [1] [2] [3] A [0] [1] [2] [3] C [0] [1]		

31.3.28 CalcButterfly3_4

Function	Execute 4-pair butterfly operation		
Format	void CalcButterfly3_4(long *Y, long *C, long* A);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
	A	Pointer to the augend/minuend/addend/subtrahend vector	(Input)
Return value	None		

Description

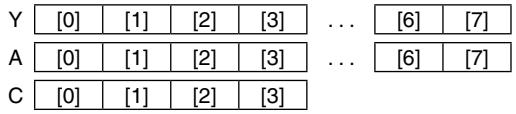
The CalcButterfly3_4 function performs butterfly operation as expressed by the following equation:

```
Y[0] = A[0] + A[7]
Y[1] = A[1] + A[6]
Y[2] = A[2] + A[5]
Y[3] = A[3] + A[4]

Y[4] = (A[0] - A[7]) * C[0]
Y[5] = (A[1] - A[6]) * C[1]
Y[6] = (A[2] - A[5]) * C[2]
Y[7] = (A[3] - A[4]) * C[3]
```

Prepare arguments Y, A, and C as long-type arrays arranged continuously as shown below. Arguments Y, A, and C must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

Data layout



31.3.29 CalcButterfly3_8

Function	Execute 8-pair butterfly operation		
Format	void CalcButterfly3_8(long *Y, long *C, long* A);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
	A	Pointer to the augend/minuend/addend/subtrahend vector	(Input)
Return value	None		

Description

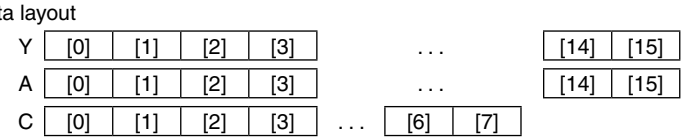
The CalcButterfly3_8 function performs butterfly operation as expressed by the following equation:

```
Y[0] = A[0] + A[15]
Y[1] = A[1] + A[14]
         :
Y[7] = A[7] + A[8]

Y[8] = (A[0] - A[15]) * C[0]
Y[9] = (A[1] - A[14]) * C[1]
         :
Y[15] = (A[7] - A[8]) * C[7]
```

Prepare arguments Y, A, and C as long-type arrays arranged continuously as shown below. Arguments Y, A, and C must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

Data layout



31.3.30 CalcButterfly3_16

Function	Execute 16-pair butterfly operation		
Format	void CalcButterfly3_16(long *Y, long *C, long* A);		
Argument	Y	Pointer to the calculation results (vector)	(Output)
	C	Pointer to the multiplier vector (CALCRAM)	(Input)
	A	Pointer to the augend/minuend/addend/subtrahend vector	(Input)
Return value	None		
Description	The CalcButterfly3_16 function performs butterfly operation as expressed by the following equation:		

$$\left.\begin{array}{l} Y[0] = A[0] + A[31] \\ Y[1] = A[1] + A[30] \\ \quad\vdots \\ Y[15] = A[15] + A[16] \end{array}\right]$$

$$\left.\begin{array}{l} Y[16] = (A[0] - A[31]) * C[0] \\ Y[17] = (A[1] - A[30]) * C[1] \\ \quad\vdots \\ Y[31] = (A[15] - A[16]) * C[15] \end{array}\right]$$

Prepare arguments Y, A, and C as long-type arrays arranged continuously as shown below. Arguments Y, A, and C must be placed at a word boundary (normally, the compiler automatically aligns them with a word boundary). Argument C must be located in CALCRAM (IVRAM or DSTRAM).

Data layout

Y	[0]	[1]	[2]	[3]	...			[30]	[31]	
A	[0]	[1]	[2]	[3]	...			[30]	[31]	
C	[0]	[1]	[2]	[3]	...	[14]	[15]			

31.4 Examples of Use

The following introduces several examples of use of the CALC module.

Example 1: Multiply-accumulate operation with unsigned integers

```
#include " calc.h"

void Sample1(void)
{
    long    *C = (long*)0x00084000;     // DSTRAM
    long    X[4] = {30, 60, 20, 90};
    long    R;

    C[0] = 1;
    C[1] = 2;
    C[2] = 3;
    C[3] = 4;

    // DSTRAM -> CALCRAM
    *(volatile unsigned char *)0x300022 = 0x96;
    *(volatile unsigned char *)0x300023 = 0x1;
    *(volatile unsigned char *)0x300022 = 0x00;

    CalcSetMode(CALC_UINT32);
    CalcInit(0)
    R = CalcMac4(C, X);

    // CALCRAM -> DSTRAM
    *(volatile unsigned char *)0x300022 = 0x96;
    *(volatile unsigned char *)0x300023 = 0x0;
    *(volatile unsigned char *)0x300022 = 0x00;
}
```

<result> R = 570 (= 1 * 30 + 2 * 60 + 3 * 20 + 4 * 90)

Example 2: Coordinate transformation with fixed-point numbers (45° rotation)

```
#include " calc.h"

void Sample3(void)
{
    //  2-D coordinate transform of rotation
    //
    //  X' = cos(n) * X - sin(n) * Y;
    //  Y' = sin(n) * X + cos(n) * Y;
    long    *C = (long*)0x84000;    // DSTRAM
    long    src[2] = {30, 60};      // [0] = Y pos, [1] = X pos
    long    dst[2];                 // [0] = Y'pos, [1] = X'pos

    // make table by q14 format for 1/4 pi radian
    C[0] = 0x2D41;          // cos(1/4pi) = 0.707107
    C[1] = 0x2D41;          // sin(1/4pi) = 0.707107

    // DSTRAM -> CALCRAM
    *(volatile unsigned char *)0x300022 = 0x96;
    *(volatile unsigned char *)0x300023 = 0x1;
    *(volatile unsigned char *)0x300022 = 0x00;

    CalcSetmode(CALC_FIXED16_Q14);
    CalcInit(0)
    CalcButterfly(dst, C, src, 1);

    // CALCRAM -> DSTRAM
    *(volatile unsigned char *)0x300022 = 0x96;
    *(volatile unsigned char *)0x300023 = 0x0;
    *(volatile unsigned char *)0x300022 = 0x00;
}
```

<result> (X, Y) = (60, 30) → (X', Y') = (21, 64)

Example 3: Color conversion with fixed-point numbers (RGB → YCbCr)

```
#include " calc.h"

void Sample2(void)
{
    //  RGB-YCbCr Color Convert
    //
    //  |Y |   | 0.299,   0.587 ,  0.114  | |R| |  0|
    //  |Cb| = |-0.1687, -0.3313,  0.5    |*|G|+|128|
    //  |Cr|   | 0.5,    -0.4187, -0.08113| |B| |128|

    long    *M      = (long*)0x84000;     // DSTRAM
    long    RGB[3]  = {140, 120,  80};
    long    A[3]    = {  0, 128, 128};
    long    YCbCr[3];

    // make table by q14 format
    M[0] = 0x00001322;      //  0.299
    M[1] = 0x00002591;      //  0.587
    M[2] = 0x0000074B;      //  0.114
    M[3] = 0xFFFFF535;      // -0.687
    M[4] = 0xFFFFEACC;      // -0.3313
    M[5] = 0x00002000;      //  0.5
    M[6] = 0x00002000;      //  0.5
    M[7] = 0xFFFFE535;      // -0.4187
    M[8] = 0xFFFFFACF;      // -0.08113

    // DSTRAM -> CALCRAM
    *(volatile unsigned char *)0x300022 = 0x96;
    *(volatile unsigned char *)0x300023 = 0x1;
    *(volatile unsigned char *)0x300022 = 0x00;

    // Color convert (RGB-YCbCr)
    CalcSetMode(CALC_FIXED16_Q14);
    CalcInit(0)
```

```
    CalcAffine3(YCbCr, M, RGB, A, 1);

    // CALCRAM -> DSTRAM
    *(volatile unsigned char *)0x300022 = 0x96;
    *(volatile unsigned char *)0x300023 = 0x0;
    *(volatile unsigned char *)0x300022 = 0x00;

    // saturation 0-255
    if (YCbCr[0] > 0xFF) YCbCr[0] = 0xFF;
    if (YCbCr[0] < 0x00) YCbCr[0] = 0x00;
    if (YCbCr[1] > 0xFF) YCbCr[1] = 0xFF;
    if (YCbCr[1] < 0x00) YCbCr[1] = 0x00;
    if (YCbCr[2] > 0xFF) YCbCr[2] = 0xFF;
    if (YCbCr[2] < 0x00) YCbCr[2] = 0x00;
}
```

<result> RGB = (140, 120, 80) → YCbCr = (121, 105, 141)

Header File (calc.h)

```
/*****************************************************************************
 *
 *      File Name   : calc.h
 *      Description : Calculation module interface
 *
 *****************************************************************************/

#ifndef ___CALC_H___
#define ___CALC_H___

#ifdef __cplusplus
extern" C" {
#endif

/*****************************************************************************
 *      Define macro
 *****************************************************************************/

/* Function table address */
#define CALC_API_BASE           ((unsigned long*)0x00023100)

// Calculation mode
#define CALC_FIXED32_Q31          0x00000000
#define CALC_SINT32               0x00000002
#define CALC_UINT32               0x00000002
#define CALC_FIXED16_Q12          0x00000010
#define CALC_FIXED16_Q13          0x00000012
#define CALC_FIXED16_Q14          0x00000112


/*****************************************************************************
 *      Function
 *****************************************************************************/
#define CalcSetMode \
            (*((void (*)(long))CALC_API_BASE[0]))
#define CalcInit \
            (*((void (*)(long))CALC_API_BASE[1]))
#define CalcMul1 \
            (*((long (*)(long*, long))CALC_API_BASE[2]))
#define CalcMul4 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[3]))
#define CalcMul6 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[4]))
#define CalcMul8 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[5]))
#define CalcMul16 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[6]))
#define CalcMac2_6 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[7]))
#define CalcMac4 \
```

```
            (*((long (*)(long*, long*))CALC_API_BASE[8]))
#define CalcMac6 \
            (*((long (*)(long*, long*))CALC_API_BASE[9]))
#define CalcMac8 \
            (*((long (*)(long*, long*))CALC_API_BASE[10]))
#define CalcMac16 \
            (*((long (*)(long*, long*))CALC_API_BASE[11]))
#define CalcMacCircular8 \
            (*((long (*)(long*, long*))CALC_API_BASE[12]))
#define CalcSubMul1 \
            (*((long (*)(long, long, long*))CALC_API_BASE[13]))
#define CalcSubMul8 \
            (*((void (*)(long*, long*, long*, long*))CALC_API_BASE[14]))
#define CalcMatrix2 \
            (*((void (*)(long*, long*, long*, unsigned long))CALC_API_BASE[15]))
#define CalcMatrix3 \
            (*((void (*)(long*, long*, long*, unsigned long))CALC_API_BASE[16]))
#define CalcMatrix4 \
            (*((void (*)(long*, long*, long*, unsigned long))CALC_API_BASE[17]))
#define CalcAffine2 \
            (*((void (*)(long*, long*, long*, long*, unsigned long))CALC_API_BASE[18]))
#define CalcAffine3 \
            (*((void (*)(long*, long*, long*, long*, unsigned long))CALC_API_BASE[19]))
#define CalcButterfly \
            (*((void (*)(long*, long*, long*, unsigned long))CALC_API_BASE[20]))
#define CalcButterflyConst \
            (*((void (*)(long*, long*, long*, unsigned long))CALC_API_BASE[21]))
#define CalcButterfly2_1 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[22]))
#define CalcButterfly3_1 \
            (*((void (*)(long*, long*, long, long))CALC_API_BASE[24]))
#define CalcButterfly3_2 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[25]))
#define CalcButterfly3_4 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[26]))
#define CalcButterfly3_8 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[27]))
#define CalcButterfly3_16 \
            (*((void (*)(long*, long*, long*))CALC_API_BASE[28]))

#ifdef __cplusplus
}
#endif
#endif                          //___CALC_H___
```

32 Electrical Characteristics

32.1 Absolute Maximum Rating

(V_{SS} = 0V)

Item	Symbol	Condition	Rated value	Unit
Internal logic and 1.8 V system I/O power supply voltage ∗1	LV_{DD}		V_{SS} - 0.3 to 2.5	V
RTC power supply voltage ∗1	$RTCV_{DD}$		V_{SS} - 0.3 to 2.5	V
PLL power supply voltage ∗1	$PLLV_{DD}$		V_{SS} - 0.3 to 2.5	V
3.3 V system I/O power supply voltage ∗1	HV_{DD}		V_{SS} - 0.3 to 4.0	V
ADC power supply voltage ∗1	AV_{DD}		V_{SS} - 0.3 to 4.0	V
3.3 V system input voltage	HV_I		V_{SS} - 0.3 to HV_{DD} + 0.5	V
1.8 V system input voltage	LV_I		V_{SS} - 0.3 to LV_{DD} + 0.5	V
Analog input voltage	AV_I		V_{SS} - 0.3 to AV_{DD} + 0.5	V
3.3 V system output voltage	HV_O		V_{SS} - 0.3 to HV_{DD} + 0.5	V
1.8 V system output voltage	LV_O		V_{SS} - 0.3 to LV_{DD} + 0.5	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
Storage temperature	Tstg		-65 to 150	°C

∗1) $HV_{DD}/AV_{DD} \geq LV_{DD}/RTCV_{DD}/PLLV_{DD}$
$LV_{DD} = RTCV_{DD} = PLLV_{DD}$
∗2) The maximum input voltage range of the #STBY pin is V_{SS} - 0.3 V to 4.0 V.

32.2 Recommended Operating Conditions

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Power supply voltage (High) ∗1	HV_{DD}	When USB is not used	2.70	3.30	3.60	V
	AV_{DD}	When USB is used	3.00	3.30	3.60	V
Power supply voltage (Low) ∗1	LV_{DD} $RTCV_{DD}$ $PLLV_{DD}$	Crystal oscillator or external clock input	1.65	1.80	1.95	V
		Ceramic oscillator	1.70	1.80	1.90	V
Input voltage	HV_I		-0.3	–	HV_{DD} + 0.3	V
	LV_I		-0.3	–	LV_{DD} + 0.3	V
	AV_I		-0.3	–	AV_{DD} + 0.3	V
Ambient temperature	Ta	Crystal oscillator or external clock input	-40	25	85	°C
		Ceramic oscillator	0	25	70	°C
CPU operating frequency	f_{CPU}		–	–	60	MHz
Bus operating frequency	f_{BUS}		–	–	60	MHz
MCLKI oscillation frequency	f_{OSC3}		–	–	48	MHz
RTCCLKI oscillation frequency	f_{OSC1}		–	32.768	–	kHz
Input rise time (normal input)	t_{ri1}		–	–	50	ns
Input fall time (normal input)	t_{fi1}		–	–	50	ns
Input rise time (Schmitt input)	t_{ri2}		–	–	5	ms
Input fall time (Schmitt input)	t_{fi2}		–	–	5	ms

∗1) $HV_{DD}/AV_{DD} \geq LV_{DD}/RTCV_{DD}/PLLV_{DD}$
$LV_{DD} = RTCV_{DD} = PLLV_{DD}$
∗2) The recommended input voltage range of the #STBY pin is V_{SS} - 0.3 V to 3.6 V.

32.3 DC Characteristics

HV_{DD} = AV_{DD} = 3.0 to 3.6 V

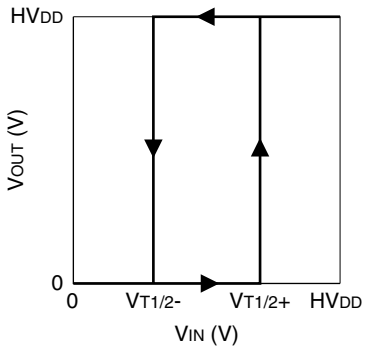
Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Input leakage current	I_{LI}	HV_{DD} = 3.6V, LV_{DD} = 1.95V, HV_{IN} = HV_{DD}, LV_{IN} = LV_{DD}, V_{IL} = 0V	-5	–	5	µA
Off-state leakage current	I_{OZ}	HV_{DD} = 3.6V, LV_{DD} = 1.95V, HV_{IN} = HV_{DD}, LV_{IN} = LV_{DD}, V_{IL} = 0V	-5	–	5	µA
HV_{DD}, AV_{DD} system I/O						
High level output voltage (TYPE1)	V_{OH1H}	HV_{DD} = 3.0V, I_{OH} = -2mA	HV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE1)	V_{OL1H}	HV_{DD} = 3.0V, I_{OL} = 2mA	–	–	0.4	V
High level output voltage (TYPE2)	V_{OH2H}	HV_{DD} = 3.0V, I_{OH} = -4mA	HV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE2)	V_{OL2H}	HV_{DD} = 3.0V, I_{OL} = 4mA	–	–	0.4	V
High level output voltage (TYPE3)	V_{OH3H}	HV_{DD} = 3.0V, I_{OH} = -8mA	HV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE3)	V_{OL3H}	HV_{DD} = 3.0V, I_{OL} = 8mA	–	–	0.4	V
High level input voltage (LVTTL)	V_{IH1H}	HV_{DD} = 3.6V	2	–	HV_{DD} + 0.3	V
Low level input voltage (LVTTL)	V_{IL1H}	HV_{DD} = 3.0V	-0.3	–	0.8	V
High level input voltage (LVCMOS)	V_{IH2H}	HV_{DD} = 3.6V	2.2	–	HV_{DD} + 0.3	V
Low level input voltage (LVCMOS)	V_{IL2H}	HV_{DD} = 3.0V	-0.3	–	0.8	V
Positive trigger input voltage (LVCMOS Schmitt)	V_{T1+}	HV_{DD} = 3.6V, LV_{DD} = 1.95V	1.4	–	2.7	V
Negative trigger input voltage (LVCMOS Schmitt)	V_{T1-}	HV_{DD} = 3.0V, LV_{DD} = 1.65V	0.6	–	1.8	V
Hysteresis voltage (LVCMOS Schmitt)	ΔV_1	HV_{DD} = 3.0V, LV_{DD} = 1.65V	0.3	–	–	V
Pull-up resistor (TYPE1)	R_{PLU1H}	V_I = 0V	25	50	120	kΩ
Pull-down resistor (TYPE1)	R_{PLD1H}	V_I = HV_{DD}	25	50	120	kΩ
Pull-up resistor (TYPE2)	R_{PLU2H}	V_I = 0V	50	100	240	kΩ
Pull-down resistor (TYPE2)	R_{PLD2H}	V_I = HV_{DD}	50	100	240	kΩ
High level holding current (Bus hold latch)	HI_{BHH}	HV_{DD} = 3.0V, V_I = 2.0V	–	–	-20	µA
Low level holding current (Bus hold latch)	HI_{BHL}	HV_{DD} = 3.0V, V_I = 0.8V	–	–	17	µA
High level inverting current (Bus hold latch)	HI_{BHH0}	HV_{DD} = 3.6V, V_I = 0.8V	-350	–	–	µA
Low level inverting current (Bus hold latch)	HI_{BHL0}	HV_{DD} = 3.6V, V_I = 2.0V	300	–	–	µA
LV_{DD}, $RTCV_{DD}$ system I/O						
High level output voltage (TYPE1)	V_{OH1L}	LV_{DD} = 1.65V, I_{OH} = -1mA	LV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE1)	V_{OL1L}	LV_{DD} = 1.65V, I_{OL} = 1mA	–	–	0.4	V
High level input voltage (LVCMOS)	V_{IH1L}	LV_{DD} = 1.95V	1.27	–	LV_{DD} + 0.3	V
Low level input voltage (LVCMOS)	V_{IL1L}	LV_{DD} = 1.65V	-0.3	–	0.57	V
Positive trigger input voltage (LVCMOS Schmitt)	V_{T2+}	HV_{DD} = 3.6V, LV_{DD} = 1.95V	0.6	–	1.4	V
Negative trigger input voltage (LVCMOS Schmitt)	V_{T2-}	HV_{DD} = 3.0V, LV_{DD} = 1.65V	0.3	–	1.1	V
Hysteresis voltage (LVCMOS Schmitt)	ΔV_2	HV_{DD} = 3.0V, LV_{DD} = 1.65V	0.02	–	–	V
Pull-down resistor (TYPE2)	R_{PLD2L}	V_I = LV_{DD}	48	120	300	kΩ
Input pin capacitance	C_I	f = 1MHz, HV_{DD} = 0V	–	–	8	pF
Output pin capacitance	C_O	f = 1MHz, HV_{DD} = 0V	–	–	8	pF
I/O pin capacitance	C_{IO}	f = 1MHz, HV_{DD} = 0V	–	–	8	pF

HV_{DD} = AV_{DD} = 2.7 to 3.6 V

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Input leakage current	I_{LI}	HV_{DD} = 3.6V, LV_{DD} = 1.95V, HV_{IN} = HV_{DD}, LV_{IN} = LV_{DD}, V_{IL} = 0V	-5	–	5	µA
Off-state leakage current	I_{OZ}	HV_{DD} = 3.6V, LV_{DD} = 1.95V, HV_{IN} = HV_{DD}, LV_{IN} = LV_{DD}, V_{IL} = 0V	-5	–	5	µA
HV_{DD}, AV_{DD} system I/O						
High level output voltage (TYPE1)	V_{OH1H}	HV_{DD} = 2.7V, I_{OH} = -1.8mA	HV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE1)	V_{OL1H}	HV_{DD} = 2.7V, I_{OL} = 1.8mA	–	–	0.4	V
High level output voltage (TYPE2)	V_{OH2H}	HV_{DD} = 2.7V, I_{OH} = -3.6mA	HV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE2)	V_{OL2H}	HV_{DD} = 2.7V, I_{OL} = 3.6mA	–	–	0.4	V
High level output voltage (TYPE3)	V_{OH3H}	HV_{DD} = 2.7V, I_{OH} = -7.2mA	HV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE3)	V_{OL3H}	HV_{DD} = 2.7V, I_{OL} = 7.2mA	–	–	0.4	V
High level input voltage (LVTTL)	V_{IH1H}	HV_{DD} = 3.6V	2	–	HV_{DD} + 0.3	V
Low level input voltage (LVTTL)	V_{IL1H}	HV_{DD} = 2.7V	-0.3	–	0.7	V
High level input voltage (LVCMOS)	V_{IH2H}	HV_{DD} = 3.6V	2.2	–	HV_{DD} + 0.3	V
Low level input voltage (LVCMOS)	V_{IL2H}	HV_{DD} = 2.7V	-0.3	–	0.7	V
Positive trigger input voltage (LVCMOS Schmitt)	V_{T1+}	HV_{DD} = 3.6V, LV_{DD} = 1.95V	1.4	–	2.7	V

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Negative trigger input voltage (LVCMOS Schmitt)	V_{T1-}	HV_{DD} = 2.7V, LV_{DD} = 1.65V	0.5	–	1.8	V
Hysteresis voltage (LVCMOS Schmitt)	ΔV_1	HV_{DD} = 2.7V, LV_{DD} = 1.65V	0.27	–	–	V
Pull-up resistor (TYPE1)	R_{PLU1H}	V_I = 0V	25	50	150	kΩ
Pull-down resistor (TYPE1)	R_{PLD1H}	V_I = HV_{DD}	25	50	150	kΩ
Pull-up resistor (TYPE2)	R_{PLU2H}	V_I = 0V	50	100	300	kΩ
Pull-down resistor (TYPE2)	R_{PLD2H}	V_I = HV_{DD}	50	100	300	kΩ
High level holding current (Bus hold latch)	HI_{BHH}	HV_{DD} = 2.7V, V_I = 2.0V	–	–	-15.7	µA
Low level holding current (Bus hold latch)	HI_{BHL}	HV_{DD} = 2.7V, V_I = 0.7V	–	–	12.2	µA
High level inverting current (Bus hold latch)	HI_{BHH0}	HV_{DD} = 3.6V, V_I = 0.7V	-350	–	–	µA
Low level inverting current (Bus hold latch)	HI_{BHL0}	HV_{DD} = 3.6V, V_I = 2.0V	300	–	–	µA
LV_{DD}, $RTCV_{DD}$ system I/O						
High level output voltage (TYPE1)	V_{OH1L}	LV_{DD} = 1.65V, I_{OH} = -1mA	LV_{DD} - 0.4	–	–	V
Low level output voltage (TYPE1)	V_{OL1L}	LV_{DD} = 1.65V, I_{OL} = 1mA	–	–	0.4	V
High level input voltage (LVCMOS)	V_{IH1L}	LV_{DD} = 1.95V	1.27	–	LV_{DD} + 0.3	V
Low level input voltage (LVCMOS)	V_{IL1L}	LV_{DD} = 1.65V	-0.3	–	0.57	V
Positive trigger input voltage (LVCMOS Schmitt)	V_{T2+}	HV_{DD} = 3.6V, LV_{DD} = 1.95V	0.6	–	1.4	V
Negative trigger input voltage (LVCMOS Schmitt)	V_{T2-}	HV_{DD} = 2.7V, LV_{DD} = 1.65V	0.3	–	1.1	V
Hysteresis voltage (LVCMOS Schmitt)	ΔV_2	HV_{DD} = 2.7V, LV_{DD} = 1.65V	0.02	–	–	V
Pull-down resistor (TYPE2)	R_{PLD2L}	V_I = LV_{DD}	48	120	300	kΩ
Input pin capacitance	C_I	f = 1MHz, HV_{DD} = 0V	–	–	8	pF
Output pin capacitance	C_O	f = 1MHz, HV_{DD} = 0V	–	–	8	pF
I/O pin capacitance	C_{IO}	f = 1MHz, HV_{DD} = 0V	–	–	8	pF

Schmitt input voltage



32.4 Current Consumption

Operating current

Item	Symbol	Condition	Min.	Typ.	Max.	Unit	Power source
Battery backup current	I_{BKUP}	OSC1: Off, RTC: Stop LV_{DD}/HV_{DD}/AV_{DD}: On #STBY = High	–	0.1	–	µA	$RTCV_{DD}$
		OSC1: 32kHz, RTC: Run LV_{DD}/HV_{DD}/AV_{DD}: Off #STBY = Low	–	1.3	–	µA	
Current consumption during execution (CPU/CCU/A0RAM) SYSCLK = OSC1	I_{DD1}	SYSCLK = 32kHz ∗2, ∗3	–	15	–	µA	LV_{DD}
Current consumption during execution (CPU/CCU/A0RAM) SYSCLK = OSC3	I_{DD2}	SYSCLK = 5MHz ∗1, ∗2	–	3	–	mA	LV_{DD}
		SYSCLK = 16MHz ∗1, ∗2	–	7	–	mA	
		SYSCLK = 24MHz ∗1, ∗2	–	10	–	mA	
		SYSCLK = 33MHz ∗1, ∗2	–	13	–	mA	
		SYSCLK = 48MHz ∗1, ∗2	–	18	–	mA	
Current consumption during execution (CPU/CCU/A0RAM) OSC3 = 48MHz, SYSCLK = PLL out (60MHz)	I_{DD3}	SYSCLK = 60MHz ∗1	–	23	–	mA	LV_{DD}
Current consumption in HALT mode (CPU/CCU/A0RAM) SYSCLK = OSC1	I_{HALT1}	SYSCLK = 32kHz ∗2, ∗3	–	5	–	µA	LV_{DD}

Item	Symbol	Condition	Min.	Typ.	Max.	Unit	Power source
Current consumption in HALT mode (CPU/CCU/A0RAM) SYSCLK = OSC3	I_{HALT2}	SYSCLK = 5MHz ∗1, ∗2	–	0.6	–	mA	LV_{DD}
		SYSCLK = 16MHz ∗1, ∗2	–	1.6	–	mA	
		SYSCLK = 24MHz ∗1, ∗2	–	2.3	–	mA	
		SYSCLK = 33MHz ∗1, ∗2	–	3	–	mA	
		SYSCLK = 48MHz ∗1, ∗2	–	4.3	–	mA	
Current consumption in HALT mode (CPU/CCU/A0RAM) OSC3 = 48MHz, SYSCLK = PLL out (60MHz)	I_{HALT3}	SYSCLK = 60MHz ∗1	–	6.3	–	mA	LV_{DD}
Current consumption in SLEEP mode SYSCLK = OSC3	I_{SLP}	OSC3: Off, OSC1: Off	–	1.0	–	µA	LV_{DD}
RTC current consumption	I_{RTC}	RTC: Run, RTCCLK = 32kHz PCLK = 48MHz, #STBY = Low	–	1.3	–	µA	$RTCV_{DD}$
		RTC: Run, RTCCLK = 32kHz PCLK = 5MHz, #STBY = High	–	40	–	µA	
		RTC: Run, RTCCLK = 32kHz PCLK = 16MHz, #STBY = High	–	110	–	µA	
		RTC: Run, RTCCLK = 32kHz PCLK = 24MHz, #STBY = High	–	165	–	µA	
		RTC: Run, RTCCLK = 32kHz PCLK = 33MHz, #STBY = High	–	225	–	µA	
		RTC: Run, RTCCLK = 32kHz PCLK = 48MHz, #STBY = High	–	325	–	µA	
		RTC: Run, RTCCLK = 32kHz PCLK = 60MHz, #STBY = High	–	405	–	µA	
USB current consumption	I_{USB}	USBCLK = 48MHz ∗4	–	3.8	–	mA	LV_{DD}
LCDC current consumption	I_{LCDC}	LCDC_CLK = 48MHz	–	0.2	–	mA	LV_{DD}
SD_MMC current consumption	I_{SDMMC}	PCLK = 48MHz	–	1.6	–	mA	LV_{DD}
I^2S current consumption	I_{I2S}	PCLK = 48MHz	–	1.2	–	mA	LV_{DD}
SDRAM + SRAM current consumption	I_{BCU}	PCLK = 48MHz	–	1.5	–	mA	LV_{DD}
BCLK current consumption	I_{BCLK}	PCLK = 48MHz	–	1.9	–	mA	LV_{DD}
T16A6 + PSC current consumption	I_{T16A}	PCLK = 48MHz	–	1.9	–	mA	LV_{DD}
USI + PSC + T8F current consumption	I_{USI}	PCLK = 48MHz	–	2.5	–	mA	LV_{DD}
USIL + PSC + T8F current consumption	I_{USIL}	PCLK = 48MHz	–	1.8	–	mA	LV_{DD}
UART + PSC current consumption	I_{UART}	PCLK = 48MHz	–	1	–	mA	LV_{DD}
REMC + PSC current consumption	I_{REMC}	PCLK = 48MHz	–	1	–	mA	LV_{DD}
T16P + PSC current consumption	I_{T16P}	PCLK = 48MHz	–	1.3	–	mA	LV_{DD}
ADC10 current consumption	I_{ADC10}	On, Conversion clock = 2MHz	–	260	–	µA	AV_{DD}
		Off	–	0.5	–	µA	
PLL current consumption	I_{PLL}	48MHz input → 60MHz output	–	2	–	mA	$PLLV_{DD}$
		Off	–	0.1	–	µA	
LCD display on	I_{LCDON}	Display: On, CPU: HALT ∗5 LCDC_CLK = 8MHz	–	8	–	mA	LV_{DD}

∗1) • The program is executed in A0RAM in Area 0.
- The OSC1 and peripheral module clocks are all disabled (off).
- The SSCG circuit is disabled (off).
- The GPIO ports are all configured to input with pulled up (no floating input).
- The current consumption is measured by executing a test program that consists of 51% load instructions, 21% arithmetic operation instructions, 10% branch instructions and 18% ext instructions.

∗2) The PLL circuit is disabled (off).

∗3) • The program is executed in A0RAM in Area 0.
- The OSC3 and peripheral module clocks are all disabled (off).
- The SSCG circuit is disabled (off).
- The GPIO ports are all configured to input with pulled up (no floating input).
- The current consumption is measured by executing a test program that consists of 51% load instructions, 21% arithmetic operation instructions, 10% branch instructions and 18% ext instructions.

∗4) USBREG_CLK and USB_CLK are enabled.

∗5) LCDC_SAPB_CLK, LCDC_AHB_CLK, LCDC_CLK, BCU_CLK, BCLK, IVRAM_ARB_CLK, and A3RAM_CLK are enabled.

32.5 A/D Converter Characteristics

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = AV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C, ADST[2:0] = 0x7

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Resolution	–		–	10	–	bits
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) Condition for Max. value: A/D clock = 2MHz. Condition for Min. value: A/D clock = 16kHz.

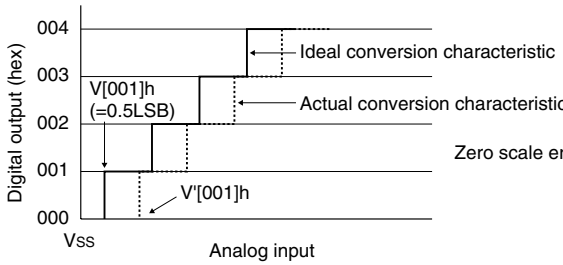
A/D conversion error

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

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

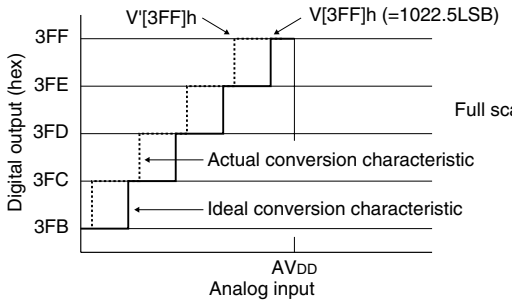
$$1LSB' = \frac{V'[3FF]h - V'[001]h}{2^{10} - 2}$$

■ Zero scale error



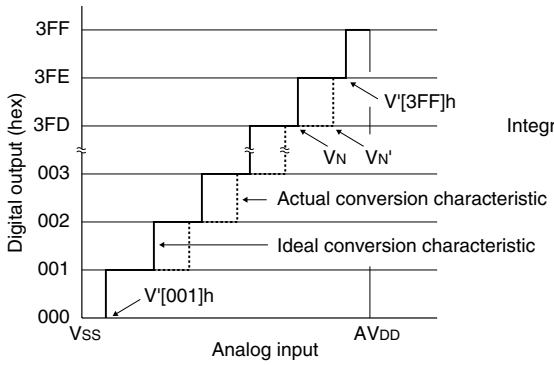
$$\text{Zero scale error } E_{ZS} = \frac{(V'[001]h - 0.5LSB') - (V[001]h - 0.5LSB)}{1LSB} \text{ [LSB]}$$

■ Full scale error



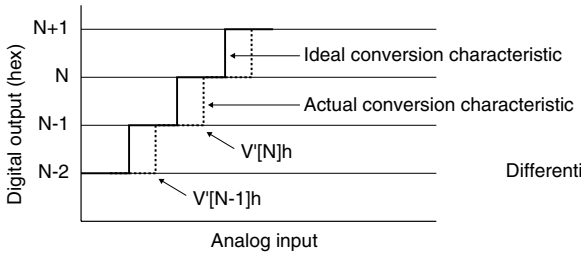
$$\text{Full scale error } E_{FS} = \frac{(V'[3FF]h + 0.5LSB') - (V[3FF]h + 0.5LSB)}{1LSB} \text{ [LSB]}$$

■ Integral linearity error



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

■ **Differential linearity error**



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

32.6 Oscillation Characteristics

Oscillation characteristics vary 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 resonator 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: LV_{DD} = $RTCV_{DD}$ = 1.65 to 1.95V, 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: LV_{DD} = $RTCV_{DD}$ = 1.65 to 1.95V, V_{SS} = 0V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{STA3}	∗1	–	–	25	ms

OSC3 ceramic oscillation

Unless otherwise specified: LV_{DD} = $RTCV_{DD}$ = 1.65 to 1.95V, V_{SS} = 0V, Ta = 25°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Oscillation start time	t_{STA3}	∗1	–	–	25	ms

∗1) When the recommended parts shown in the “Basic External Wiring Diagram” chapter are used

32.7 PLL Characteristics

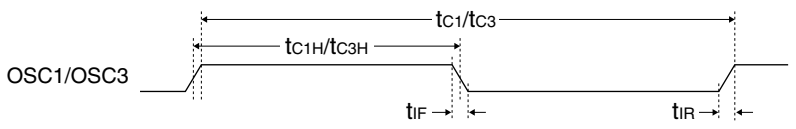
Unless otherwise specified: $PLLV_{DD}$ = 1.65 to 1.95V, $PLLV_{SS}$ = 0V, Ta = -40 to 85°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Input frequency ∗1	f_{PLLIN}		5	–	48	MHz
Output frequency ∗2	f_{PLLOUT}		20	–	60	MHz
Output stabilization time	t_{PLL}		–	–	200	µs

∗1) Input clock source divider: OSC3 ×1, ×1/2, ×1/3, ×1/4, ×1/5, ×1/6, ×1/7, ×1/8, ×1/9, ×1/10
∗2) Multiplication rate: ×1, ×2, ×3, ×4, ×5, ×6, ×7, ×8, ×9, ×10, ×11, ×12, ×13, ×14, ×15, ×16

32.8 AC Characteristics

32.8.1 External Clock Input Characteristics



$t_{C1ED} = t_{C1H}/t_{C1}$
$t_{C3ED} = t_{C3H}/t_{C3}$

OSC1 external clock

Unless otherwise specified: LV_{DD} = $RTCV_{DD}$ = 1.65 to 1.95V, 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

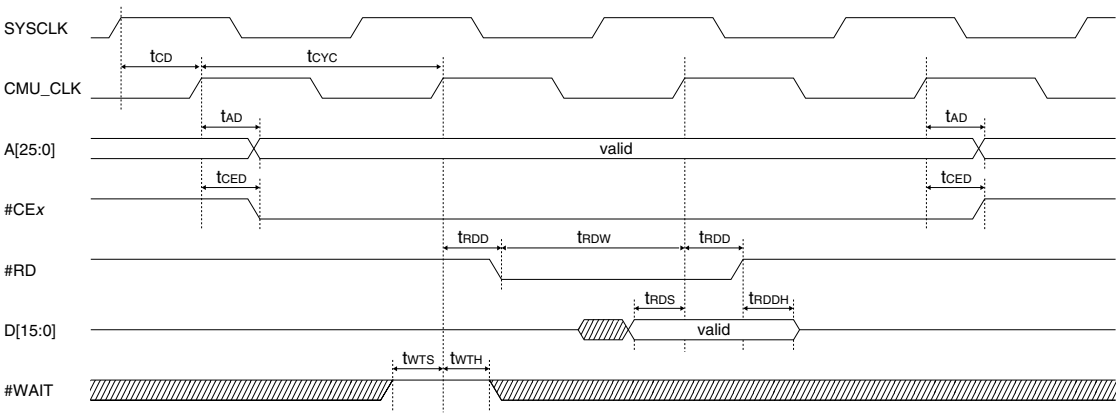
OSC3 external clock

Unless otherwise specified: LV_{DD} = $RTCV_{DD}$ = 1.65 to 1.95V, V_{SS} = 0V, Ta = -40 to 85°C

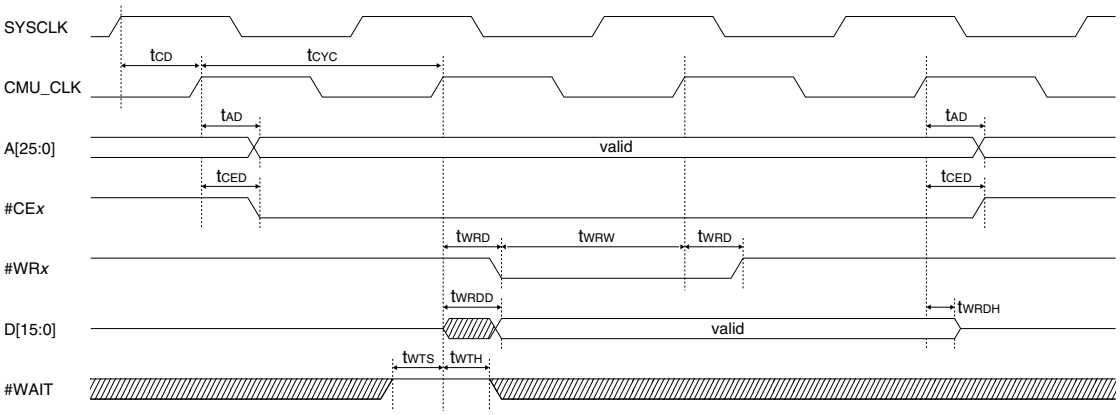
Item	Symbol	Min.	Typ.	Max.	Unit
OSC3 external clock cycle time	t_{C3}	20.83	–	500	ns
OSC3 external clock input duty	t_{C3ED}	45	–	55	%
OSC3 external clock input rise time	t_{IF}	–	–	5	ns
OSC3 external clock input fall time	t_{IR}	–	–	5	ns

32.8.2 SRAMC AC Characteristics

SRAM read cycle



SRAM write cycle



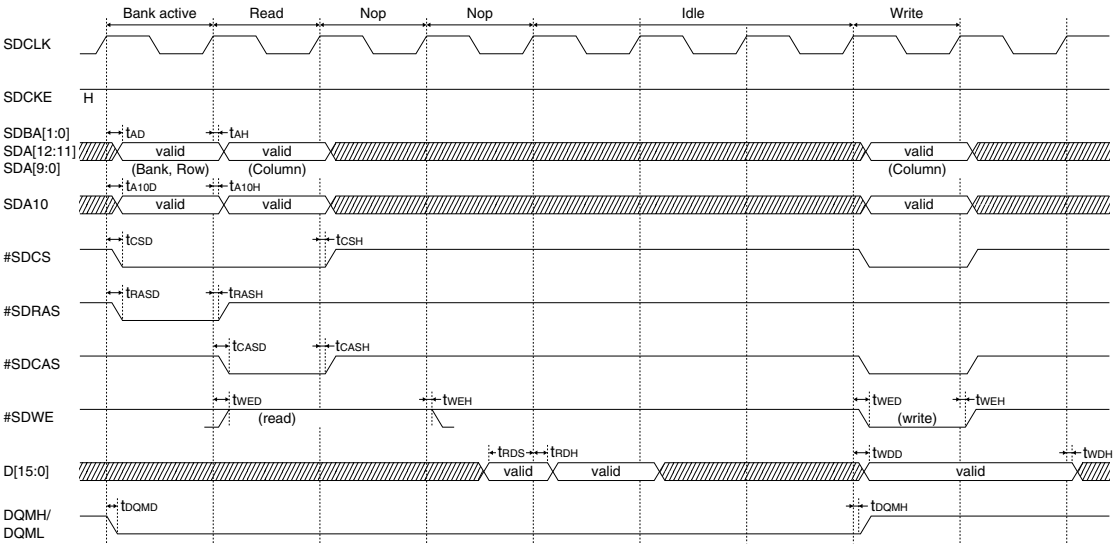
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, external load = 50pF, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
CMU_CLK output delay time	t_{CD}	–	–	24	ns
Address delay time	t_{AD}	–	–	7	ns
#CE*x* delay time	t_{CED}	–	–	7	ns
Write delay time	t_{WRD}	–	–	7	ns
Write data delay time	t_{WRDD}	–	–	7	ns
Write data hold time	t_{WRDH}	1	–	–	ns
Read delay time	t_{RDD}	–	–	7	ns
Read data setup time	t_{RDS}	7	–	–	ns
Read data hold time	t_{RDDH}	0	–	–	ns
Write signal pulse width	t_{WRW}	t_{CYC}(1 + WC) - 7	–	–	ns
Read signal pulse width	t_{RDW}	t_{CYC}(1 + WC) - 7	–	–	ns
Read address access time	t_{ACC}	–	–	t_{CYC}(1 + WC) - 14	ns
Chip enable access time	t_{CEAC}	–	–	t_{CYC}(1 + WC) - 14	ns
Read signal access time	t_{RDAC}	–	–	t_{CYC}(1 + WC) - 14	ns
#WAIT setup time	t_{WTS}	48	–	–	ns
#WAIT hold time	t_{WTH}	0	–	–	ns

WC: Number of wait cycles

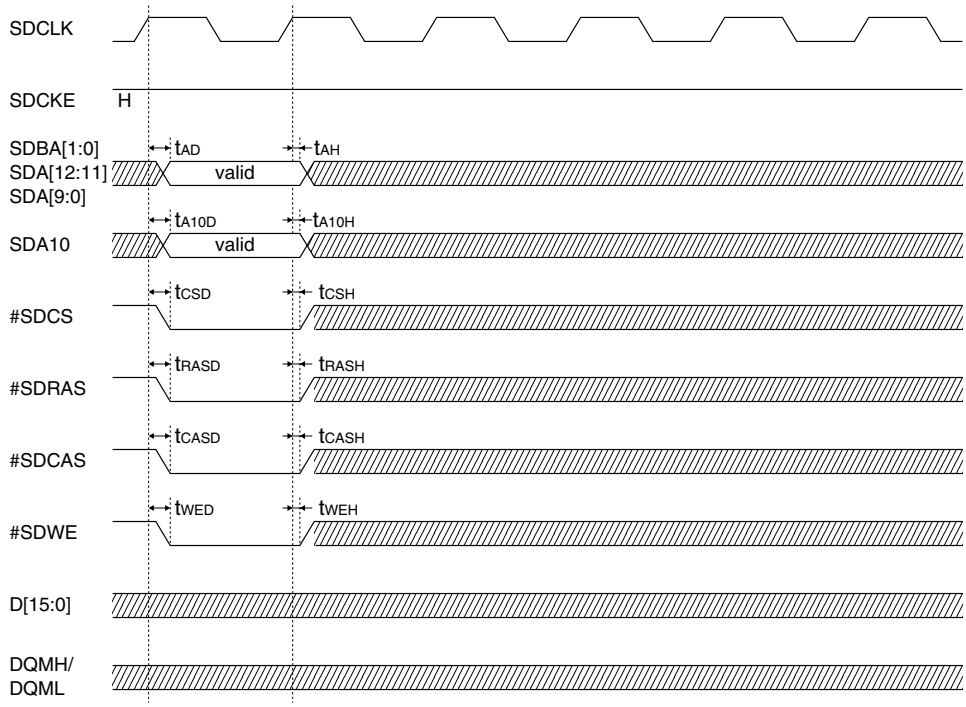
32.8.3 SDRAMC AC Characteristics

SDRAM access cycle

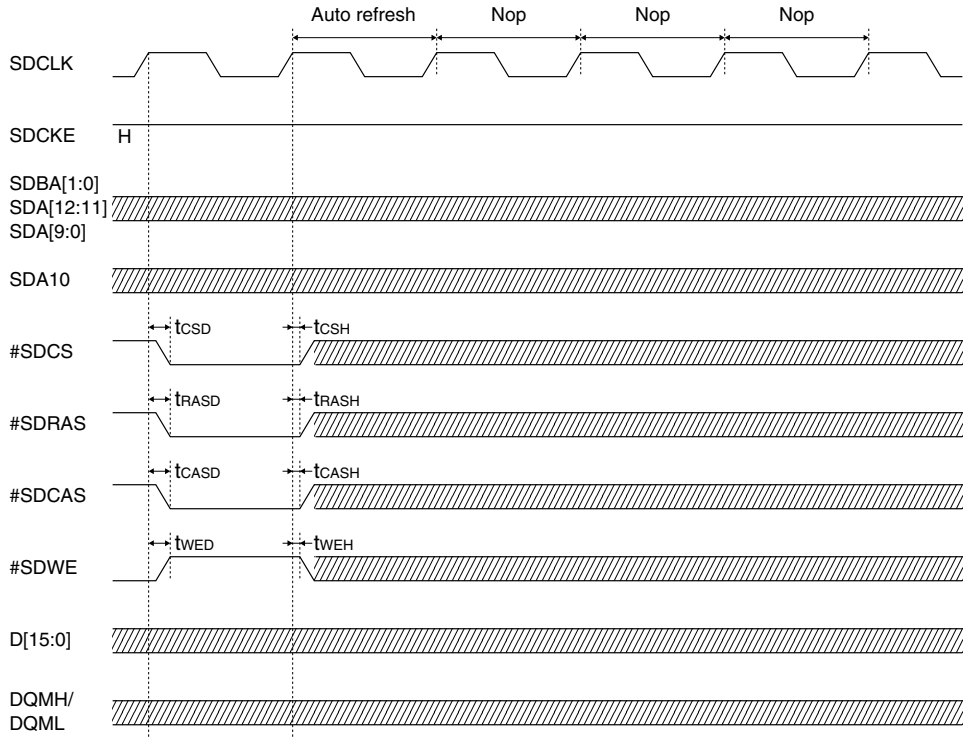


∗ Read: CAS latency = 2, burst length = 2 Write: single write

SDRAM mode-register-set cycle

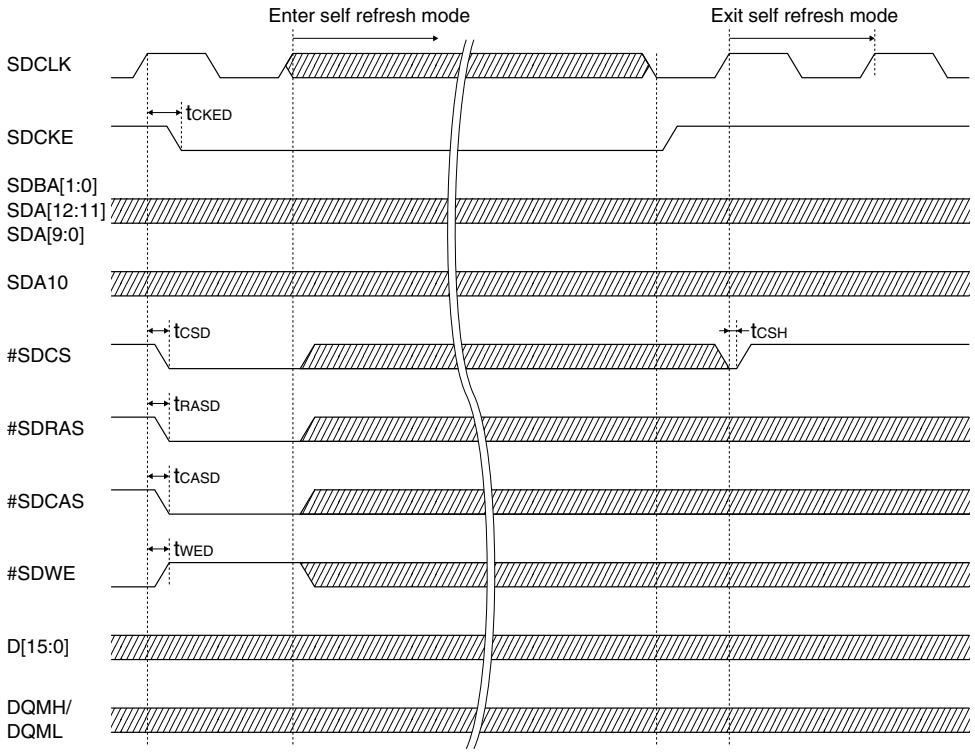


SDRAM auto-refresh cycle



∗ A precharge cycle is necessary before entering the auto refresh mode.

SDRAM self-refresh cycle



∗ A precharge cycle is necessary before entering the self refresh mode.

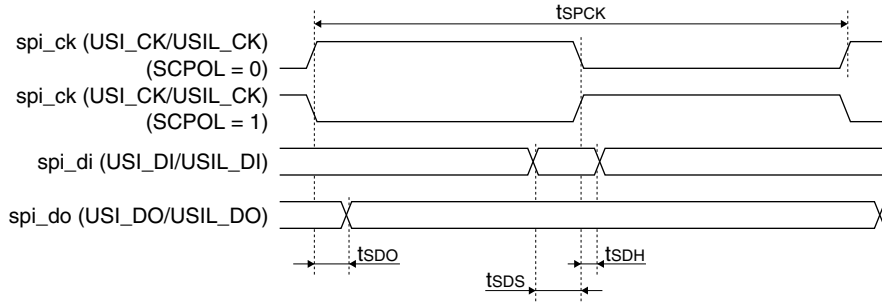
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C
External load conditions: Address bus/data bus = 50pF, SDCLK/#SDCS signals = 20pF

Item	Symbol	Min.	Typ.	Max.	Unit
Address delay time	t_{AD}	–	–	13	ns
Address hold time	t_{AH}	1.2	–	–	ns
SDA10 delay time	t_{A10D}	–	–	13	ns
SDA10 hold time	t_{A10H}	1.2	–	–	ns
#SDCS delay time	t_{CSD}	–	–	13	ns
#SDCS hold time	t_{CSH}	1.2	–	–	ns
#SDRAS signal delay time	t_{RASD}	–	–	13	ns
#SDRAS signal hold time	t_{RASH}	1.2	–	–	ns
#SDCAS signal delay time	t_{CASD}	–	–	13	ns
#SDCAS signal hold time	t_{CASH}	1.2	–	–	ns
DQMH, DQML signal delay time	t_{DQMD}	–	–	13	ns
DQMH, DQML signal hold time	t_{DQMH}	1.2	–	–	ns
SDCKE signal delay time	t_{CKED}	–	–	13	ns
SDCKE signal hold time	t_{CKEH}	1.2	–	–	ns
#SDWE signal delay time	t_{WED}	–	–	13	ns
#SDWE signal hold time	t_{WEH}	1.2	–	–	ns
Read data setup time	t_{RDS}	9	–	–	ns
Read data hold time	t_{RDH}	0	–	–	ns
Write data delay time	t_{WDD}	–	–	13	ns
Write data hold time	t_{WDH}	1.2	–	–	ns

Note: All the signals change at the rising edge of the SDRAM clock.

32.8.4 USI/USIL AC Characteristics

SPI master/slave mode (USI/USIL)



SPI master mode (normal mode)

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
spi_ck cycle time	t_{SPCK}	42 + t_{PCLK}	–	–	ns
spi_di setup time	t_{SDS}	42 + t_{PCLK}	–	–	ns
spi_di hold time	t_{SDH}	0	–	–	ns
spi_do output delay time	t_{SDO}	–	–	5	ns

SPI master mode (fast mode)

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
spi_ck cycle time	t_{SPCK}	44	–	–	ns
spi_di setup time	t_{SDS}	44	–	–	ns
spi_di hold time	t_{SDH}	0	–	–	ns
spi_do output delay time	t_{SDO}	–	–	7	ns

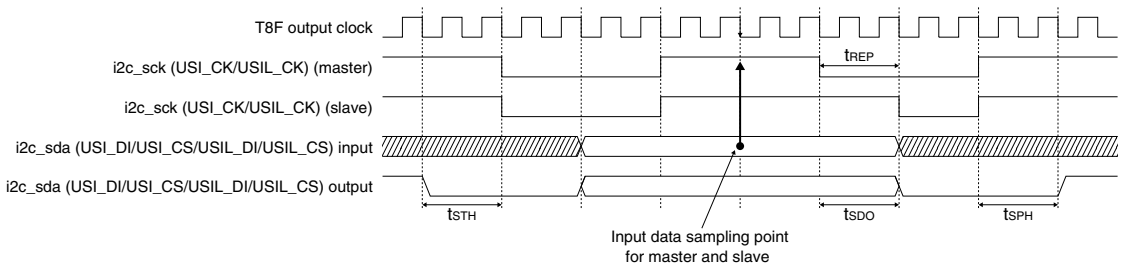
SPI slave mode

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
spi_ck cycle time	t_{SPCK}	29 + t_{PCLK}	–	–	ns
spi_di setup time	t_{SDS}	20 + t_{PCLK}	–	–	ns
spi_di hold time	t_{SDH}	9	–	–	ns
spi_do output delay time	t_{SDO}	–	–	42	ns
spi_do output hold time	t_{SDOH}	–	–	2 ∗ t_{PCLK} + 40.552	ns

t_{PCLK}: PCLK (peripheral module clock supplied from the CMU) clock cycle time

I²C master/slave mode (USI/USIL)



I²C master mode

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
i2c_sck cycle time	t_{SCL}	8 ∗ t_{T8F}	–	–	ns
i2c_sda output delay time	t_{SDO}	–	–	2 ∗ t_{T8F} + 11	ns
Start condition hold time	t_{STH}	4 ∗ t_{T8F} - 5	–	–	ns
Stop condition hold time	t_{SPH}	3 ∗ t_{T8F} - 4	–	–	ns

I²C slave mode

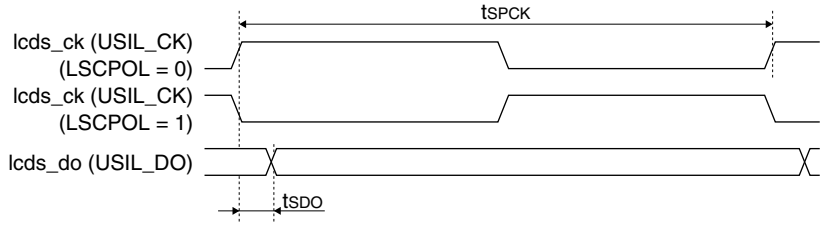
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
i2c_sck cycle time	t_{SCL}	8 * t_{T8F}	–	–	ns
i2c_sck input clock response delay time	t_{REP}	4 * t_{T8F}	–	–	ns
i2c_sda output delay time	t_{SDO}	–	–	2 * t_{T8F} + 11	ns
Start condition hold time	t_{STH}	7 * t_{PCLK} + 16	–	–	ns
Stop condition hold time	t_{SPH}	7 * t_{PCLK} + 17	–	–	ns

t_{PCLK}: PCLK (peripheral module clock supplied from the CMU) clock cycle time

t_{T8F}: T8F output clock cycle time

LCD SPI mode (USIL)



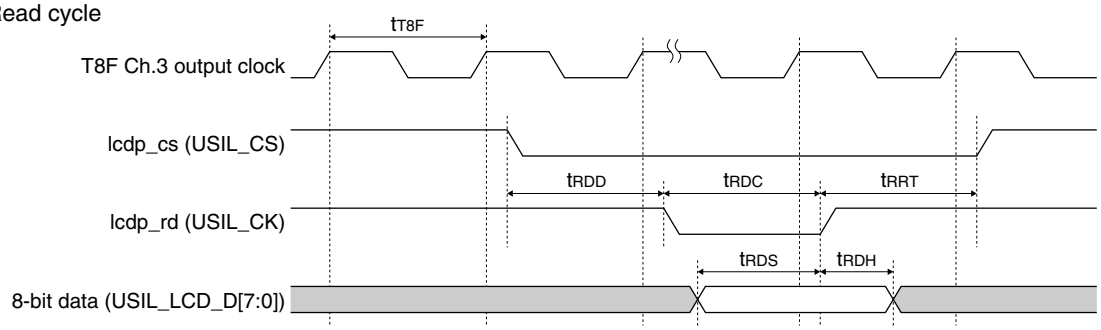
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
spi_ck cycle time	t_{SPCK}	t_{T8F}	–	–	ns
spi_do output delay time	t_{SDO}	–	–	1	ns

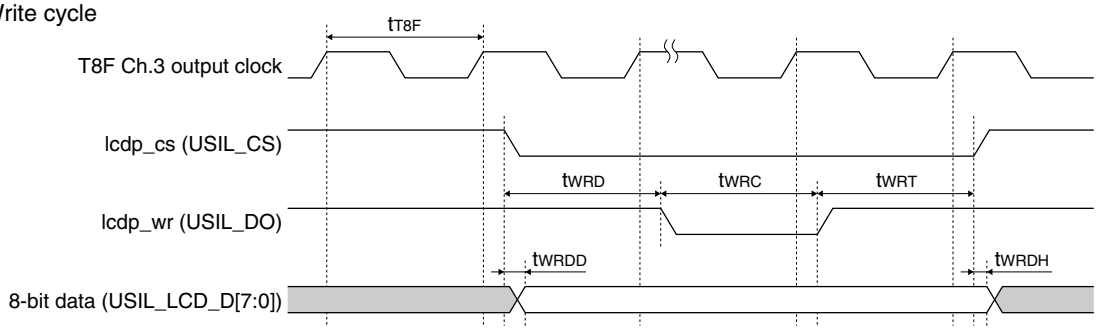
t_{T8F}: T8F output clock cycle time

LCD parallel mode (USIL)

Read cycle



Write cycle



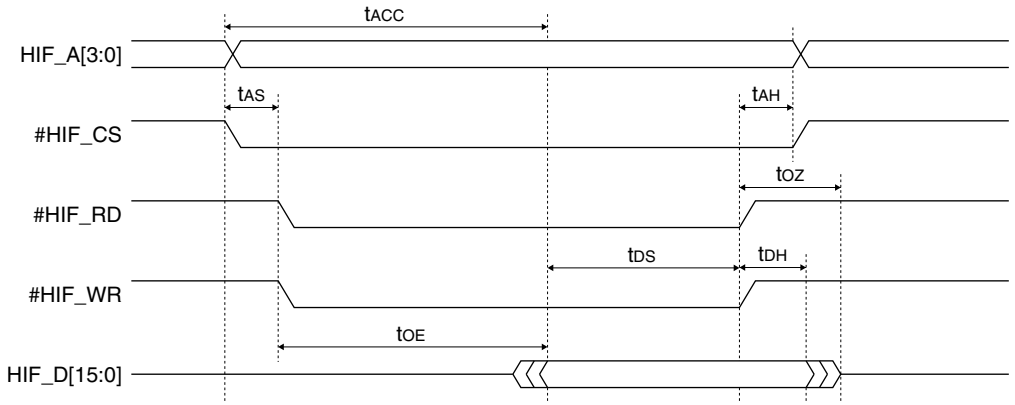
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
lcdp_rd output delay time	t_{RDD}	–	–	t_S + 1	ns
lcdp_rd period time	t_{RDC}	–	–	t_W + t_{T8F}	ns
lcdp_rd to lcdp_cs rising time	t_{RRT}	–	–	t_H - 1	ns
Read data setup time	t_{RDS}	22	–	–	ns
Read data hold time	t_{RDH}	0	–	–	ns
lcdp_wr output delay time	t_{WRD}	–	–	t_S + 1	ns
lcdp_wr period time	t_{WRC}	–	–	t_W + t_{T8F}	ns
lcdp_wr to lcdp_cs rising time	t_{WRT}	–	–	t_H + 1	ns
Write data output delay time	t_{WRDD}	–	–	3	ns
Write data output hold time	t_{WRDH}	–	–	0	ns

t_S: Setup cycle time set using LPST[1:0]/USIL_LPAC register
t_W: Wait cycle time set using LPWT[3:0]/USIL_LPAC register
t_H: Hold cycle time set using LPHD[1:0]/USIL_LPAC register
t_{T8F}: T8F output clock cycle time

32.8.5 HIF AC Characteristics

Asynchronous mode

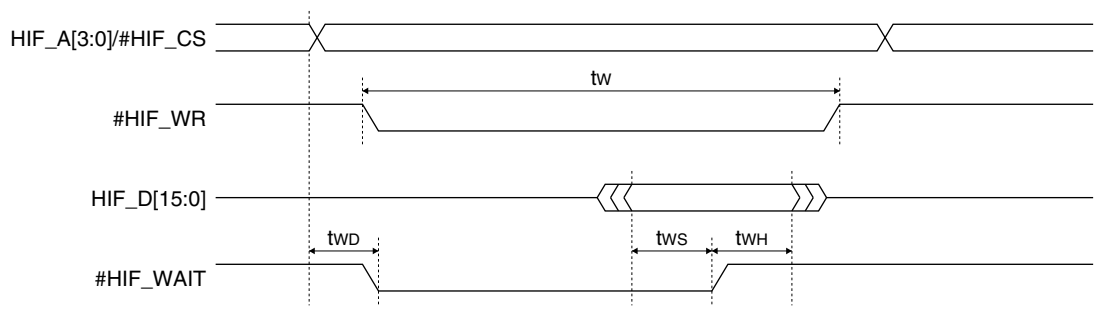


Asynchronous I/F timing

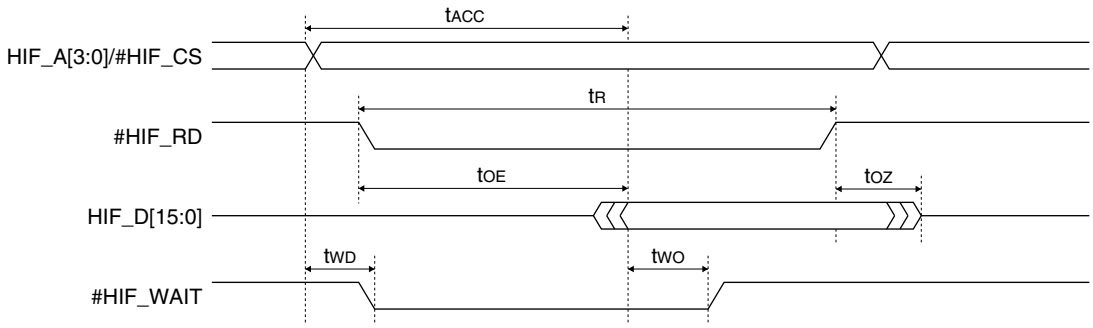
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
Data setup time	t_{DS}	–	–	25	ns
Data hold time	t_{DH}	1	–	–	ns
Address setup time	t_{AS}	–	–	25	ns
Address hold time	t_{AH}	1	–	–	ns
Address to output delay	t_{ACC}	–	–	25	ns
Read to output delay	t_{OE}	–	–	25	ns
Read to high impedance	t_{OZ}	–	–	25	ns

Synchronous mode



Synchronous I/F write timing



Synchronous I/F read timing

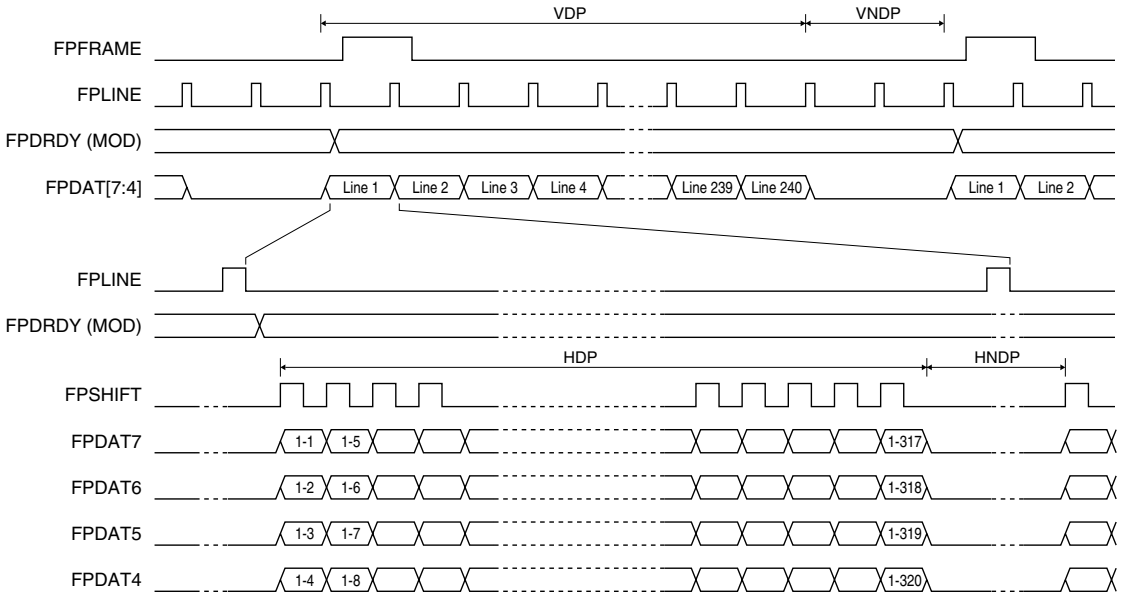
Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 2.7 to 3.6V, V_{SS} = 0V, Ta = -40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
Data setup time	t_{WS}	–	–	25	ns
Data hold time	t_{WH}	0	–	–	ns
Wait signal delay time	t_{WD}	–	–	25	ns
Read data valid time	t_{WO}	t_{PCLK} - 8	–	–	ns
Address to output delay	t_{ACC}	–	–	25	ns
Read to output delay	t_{OE}	–	–	25	ns
Read to high impedance	t_{OZ}	–	–	25	ns
Write pulse width	t_W	4 * t_{PCLK}	–	–	ns
Read pulse width	t_R	5 * t_{PCLK}	–	–	ns

t_{PCLK}: PCLK (peripheral module clock supplied from the CMU) clock cycle time

32.8.6 LCDC AC Characteristics

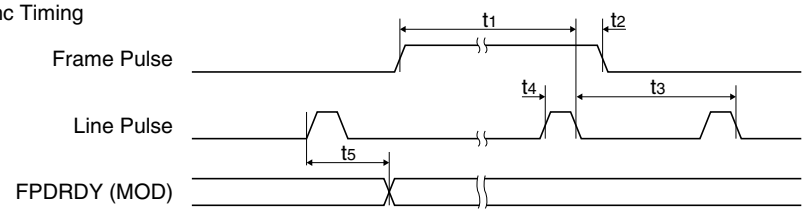
4-bit single monochrome panel timing



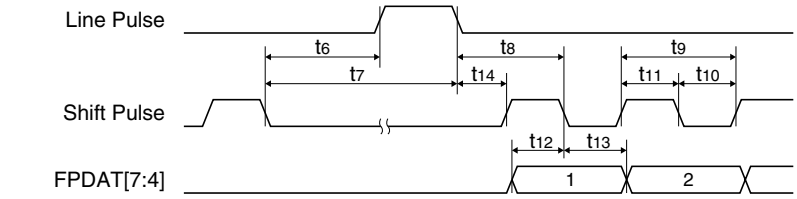
* Diagram drawn with 2 FPLINE vertical blank period
  Example timing for a 320 × 240 panel
  For this timing diagram FPSHIFT_MSK is set to 1

HDP (Horizontal Display Period) = (HDPCNT[6:0] + 1) × 8 (Ts)
HNDP (Horizontal Non-Display Period) = (HTCNT[6:0] - HDPCNT[6:0]) × 8 (Ts)
VDP (Vertical Display Period) = VDPCNT[9:0] + 1 (lines)
VNDP (Vertical Non-Display Period) = VTCNT[9:0] - VDPCNT[9:0] (lines)

Sync Timing



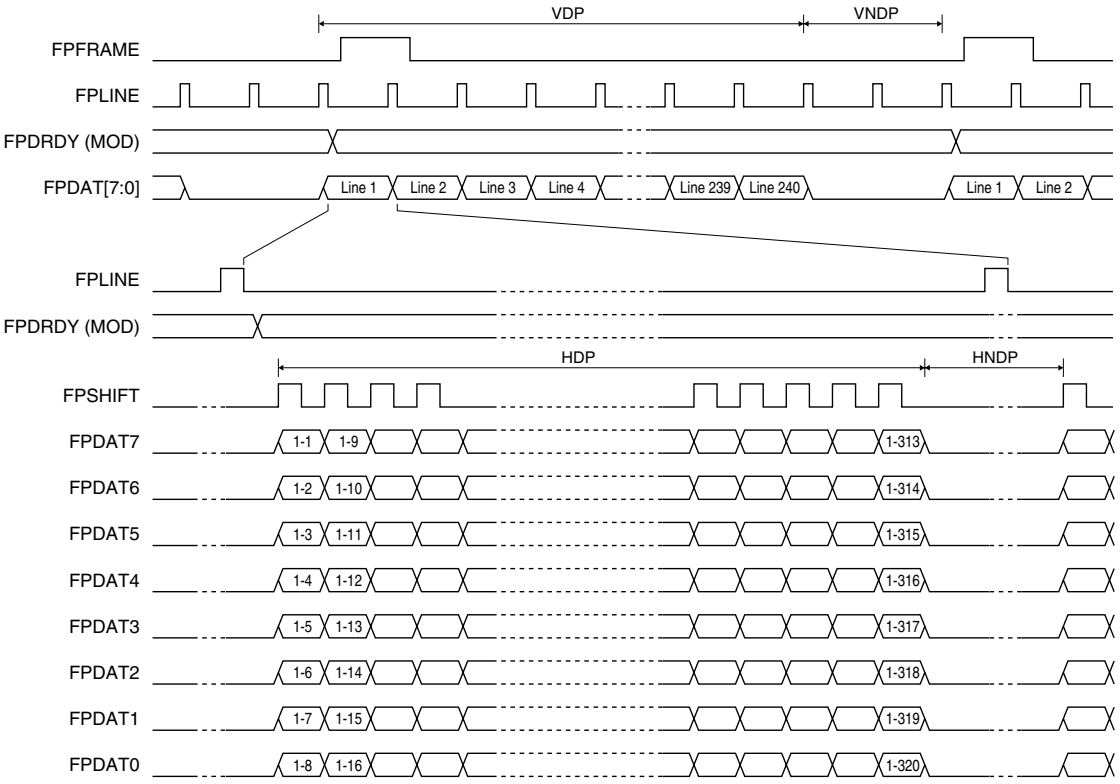
Data Timing



Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	Frame Pulse setup to Line Pulse falling edge	note 2	–	–	(note 1)
t_2	Frame Pulse hold from Line Pulse falling edge	9	–	–	Ts
t_3	Line Pulse period	note 3	–	–	–
t_4	Line Pulse width	9	–	–	Ts
t_5	MOD delay from Line Pulse rising edge	1	–	–	Ts
t_6	Shift Pulse falling edge to Line Pulse rising edge	note 4	–	–	–
t_7	Shift Pulse falling edge to Line Pulse falling edge	note 5	–	–	–
t_8	Line Pulse falling edge to Shift Pulse falling edge	t_{14} + 2	–	–	Ts
t_9	Shift Pulse period	4	–	–	Ts
t_{10}	Shift Pulse width low	2	–	–	Ts
t_{11}	Shift Pulse width high	2	–	–	Ts
t_{12}	FPDAT[7:4] setup to Shift Pulse falling edge	2	–	–	Ts
t_{13}	FPDAT[7:4] hold from Shift Pulse falling edge	2	–	–	Ts
t_{14}	Line Pulse falling edge to Shift Pulse rising edge	23	–	–	Ts

note) 1. Ts = pixel clock period
2. t_{1min} = t_{3min} - 9 (Ts)
3. t_{3min} = HDP + HNDP (Ts)
4. t_{6min} = HNDP + 2 (Ts)
5. t_{7min} = HNDP + 11 (Ts)

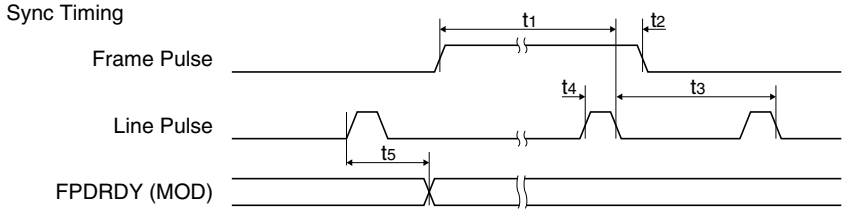
8-bit single monochrome panel timing



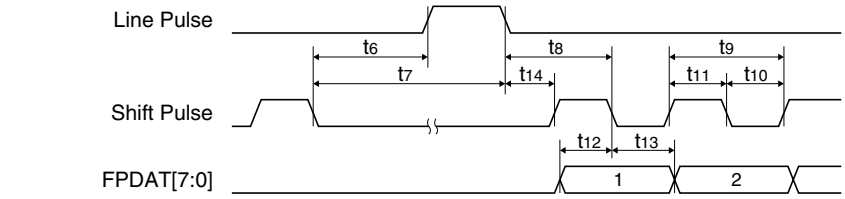
* Diagram drawn with 2 FPLINE vertical blank period
Example timing for a 320 × 240 panel
For this timing diagram FPSHIFT_MSK is set to 1

HDP (Horizontal Display Period) = (HDPCNT[6:0] + 1) × 8 (Ts)
HNDP (Horizontal Non-Display Period) = (HTCNT[6:0] - HDPCNT[6:0]) × 8 (Ts)
VDP (Vertical Display Period) = VDPCNT[9:0] + 1 (lines)
VNDP (Vertical Non-Display Period) = VTCNT[9:0] - VDPCNT[9:0] (lines)

Sync Timing



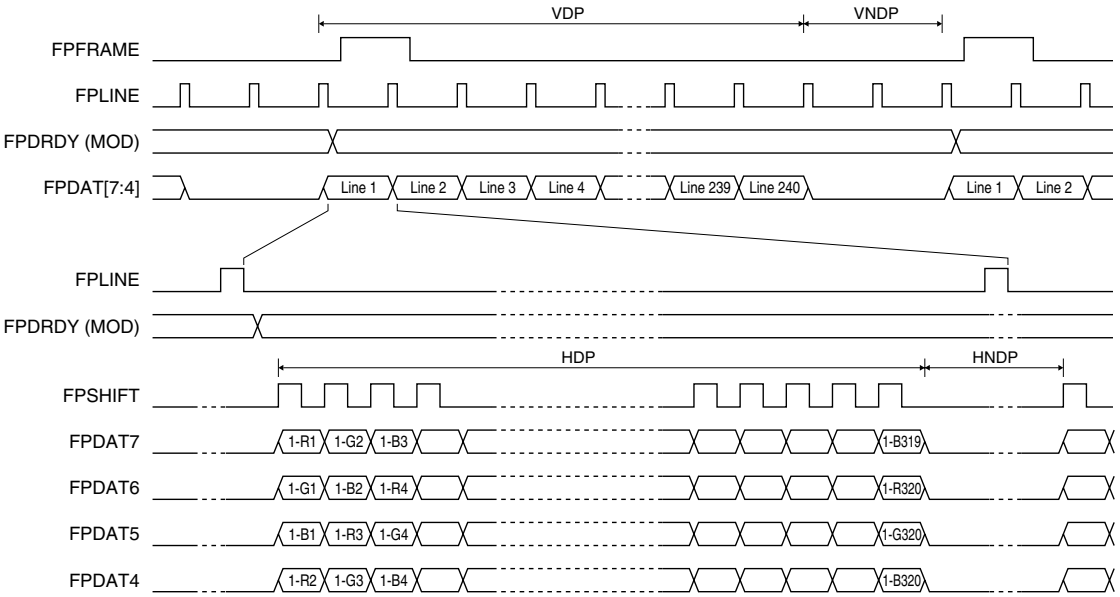
Data Timing



Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	Frame Pulse setup to Line Pulse falling edge	note 2	–	–	(note 1)
t_2	Frame Pulse hold from Line Pulse falling edge	9	–	–	Ts
t_3	Line Pulse period	note 3	–	–	–
t_4	Line Pulse width	9	–	–	Ts
t_5	MOD delay from Line Pulse rising edge	1	–	–	Ts
t_6	Shift Pulse falling edge to Line Pulse rising edge	note 4	–	–	–
t_7	Shift Pulse falling edge to Line Pulse falling edge	note 5	–	–	–
t_8	Line Pulse falling edge to Shift Pulse falling edge	t_{14} + 4	–	–	Ts
t_9	Shift Pulse period	8	–	–	Ts
t_{10}	Shift Pulse width low	4	–	–	Ts
t_{11}	Shift Pulse width high	4	–	–	Ts
t_{12}	FPDAT[7:0] setup to Shift Pulse falling edge	4	–	–	Ts
t_{13}	FPDAT[7:0] hold from Shift Pulse falling edge	4	–	–	Ts
t_{14}	Line Pulse falling edge to Shift Pulse rising edge	23	–	–	Ts

note) 1. Ts = pixel clock period
2. t_{1min} = t_{3min} - 9 (Ts)
3. t_{3min} = HDP + HNDP (Ts)
4. t_{6min} = HNDP + 4 (Ts)
5. t_{7min} = HNDP + 13 (Ts)

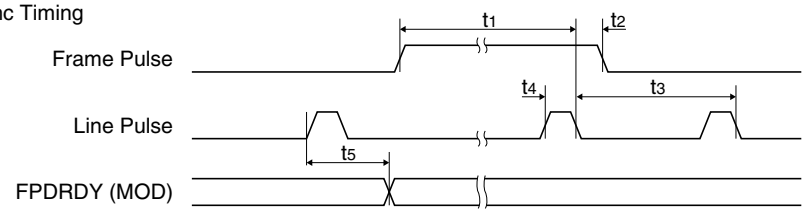
4-bit single color panel timing



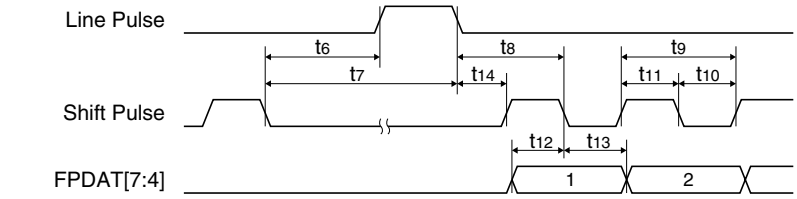
* Diagram drawn with 2 FPLINE vertical blank period
Example timing for a 320 × 240 panel

HDP (Horizontal Display Period) = (HDPCNT[6:0] + 1) × 8 (Ts)
HNDP (Horizontal Non-Display Period) = (HTCNT[6:0] - HDPCNT[6:0]) × 8 (Ts)
VDP (Vertical Display Period) = VDPCNT[9:0] + 1 (lines)
VNDP (Vertical Non-Display Period) = VTCNT[9:0] - VDPCNT[9:0] (lines)

Sync Timing



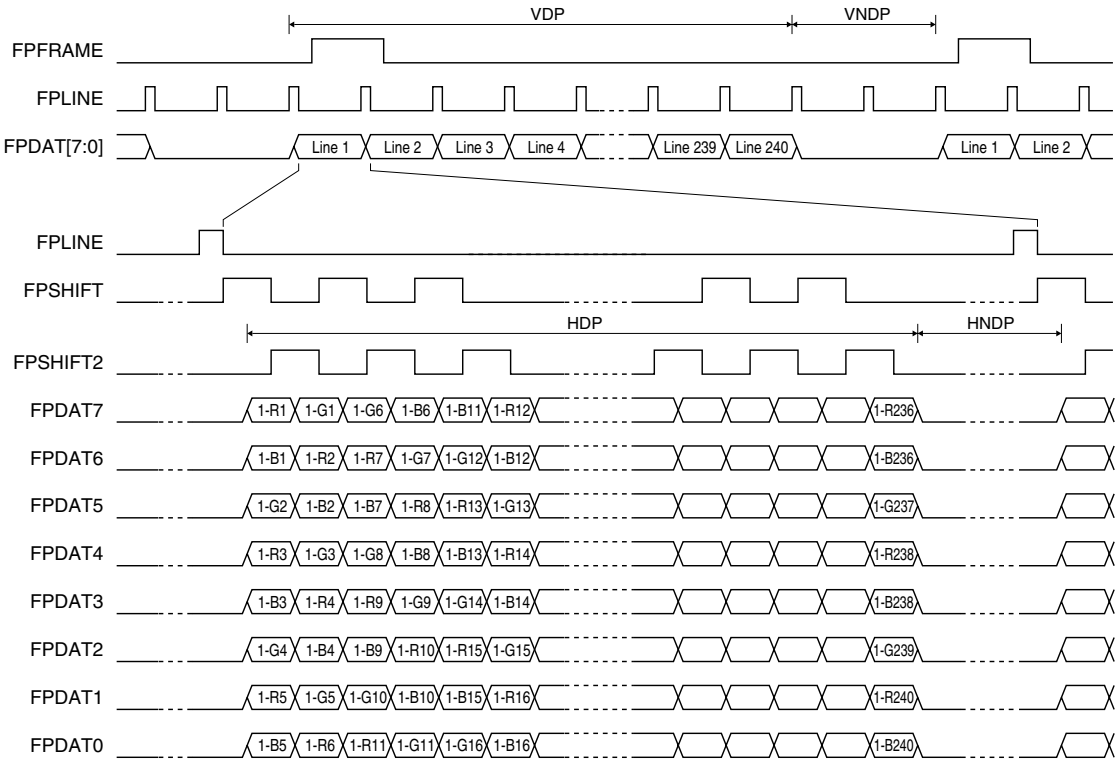
Data Timing



Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	Frame Pulse setup to Line Pulse falling edge	note 2	–	–	(note 1)
t_2	Frame Pulse hold from Line Pulse falling edge	9	–	–	Ts
t_3	Line Pulse period	note 3	–	–	–
t_4	Line Pulse width	9	–	–	Ts
t_5	MOD delay from Line Pulse rising edge	1	–	–	Ts
t_6	Shift Pulse falling edge to Line Pulse rising edge	note 4	–	–	–
t_7	Shift Pulse falling edge to Line Pulse falling edge	note 5	–	–	–
t_8	Line Pulse falling edge to Shift Pulse falling edge	t_{14} + 0.5	–	–	Ts
t_9	Shift Pulse period	1	–	–	Ts
t_{10}	Shift Pulse width low	0.5	–	–	Ts
t_{11}	Shift Pulse width high	0.5	–	–	Ts
t_{12}	FPDAT[7:4] setup to Shift Pulse falling edge	0.5	–	–	Ts
t_{13}	FPDAT[7:4] hold from Shift Pulse falling edge	0.5	–	–	Ts
t_{14}	Line Pulse falling edge to Shift Pulse rising edge	23 (24)	–	–	Ts

note) 1. Ts = pixel clock period
2. t_{1min} = t_{3min} - 9 (Ts)
3. t_{3min} = HDP + HNDP (Ts)
4. t_{6min} = HNDP + 1.5 (Ts)
5. t_{7min} = HNDP + 10.5 (Ts)

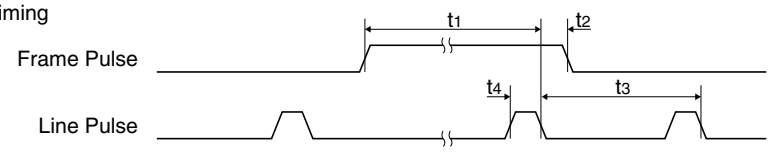
8-bit single color panel timing (Format 1)



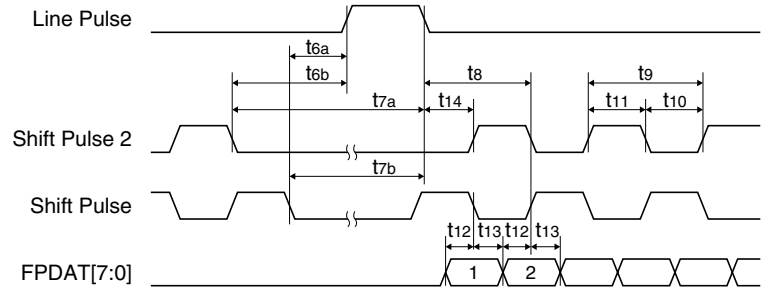
* Diagram drawn with 2 FPLINE vertical blank period
Example timing for a 320 × 240 panel

HDP (Horizontal Display Period) = (HDPCNT[6:0] + 1) × 8 (Ts)
HNDP (Horizontal Non-Display Period) = (HTCNT[6:0] - HDPCNT[6:0]) × 8 (Ts)
VDP (Vertical Display Period) = VDPCNT[9:0] + 1 (lines)
VNDP (Vertical Non-Display Period) = VTCNT[9:0] - VDPCNT[9:0] (lines)

Sync Timing



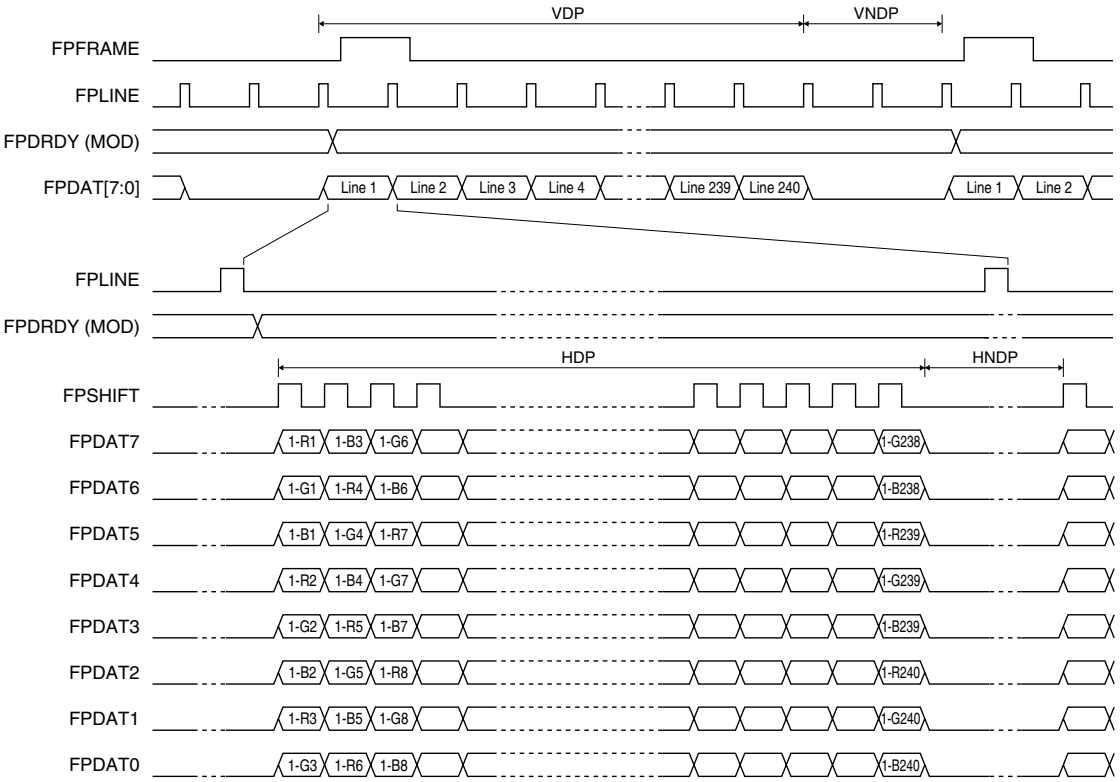
Data Timing



Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	Frame Pulse setup to Line Pulse falling edge	note 2	–	–	(note 1)
t_2	Frame Pulse hold from Line Pulse falling edge	9	–	–	Ts
t_3	Line Pulse period	note 3	–	–	–
t_4	Line Pulse width	9	–	–	Ts
t_{6a}	Shift Pulse falling edge to Line Pulse rising edge	note 4	–	–	–
t_{6b}	Shift Pulse 2 falling edge to Line Pulse rising edge	note 5			–
t_{7a}	Shift Pulse 2 falling edge to Line Pulse falling edge	note 6	–	–	–
t_{7b}	Shift Pulse falling edge to Line Pulse falling edge	note 7			–
t_8	Line Pulse falling edge to Shift Pulse rising, Shift Pulse 2 falling edge	t_{14} + 2	–	–	Ts
t_9	Shift Pulse 2, Shift Pulse period	4	–	–	Ts
t_{10}	Shift Pulse 2, Shift Pulse width low	2	–	–	Ts
t_{11}	Shift Pulse 2, Shift Pulse width high	2	–	–	Ts
t_{12}	FPDAT[7:0] setup to Shift Pulse 2, Shift Pulse falling edge	1	–	–	Ts
t_{13}	FPDAT[7:0] hold from Shift Pulse 2, Shift Pulse falling edge	1	–	–	Ts
t_{14}	Line Pulse falling edge to Shift Pulse rising edge	23 (25)	–	–	Ts

note) 1. Ts = pixel clock period
2. t_{1min} = t_{3min} - 9 (Ts)
3. t_{3min} = HDP + HNDP (Ts)
4. t_{6amin} = HNDP + t_{13} - t_{10} + 1 (Ts)
5. t_{6bmin} = HNDP + t_{13} + 1 (Ts)
6. t_{7amin} = HNDP + 11 (Ts)
7. t_{7bmin} = HNDP + 11 - t_{10} (Ts)

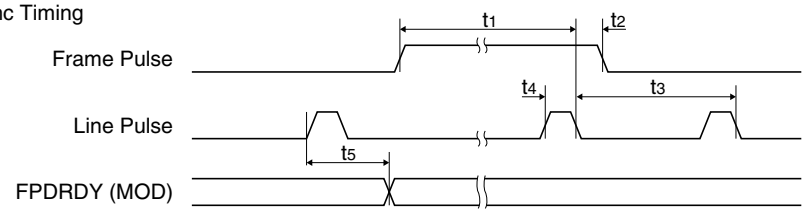
8-bit single color panel timing (Format 2)



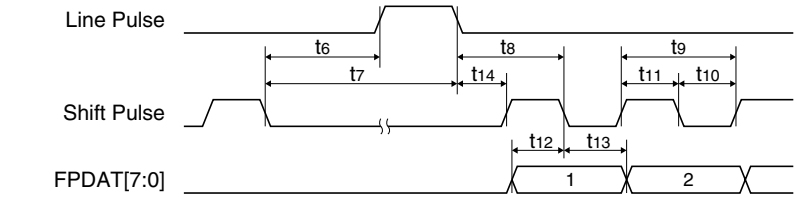
* Diagram drawn with 2 FPLINE vertical blank period
Example timing for a 320 × 240 panel

HDP (Horizontal Display Period) = (HDPCNT[6:0] + 1) × 8 (Ts)
HNDP (Horizontal Non-Display Period) = (HTCNT[6:0] - HDPCNT[6:0]) × 8 (Ts)
VDP (Vertical Display Period) = VDPCNT[9:0] + 1 (lines)
VNDP (Vertical Non-Display Period) = VTCNT[9:0] - VDPCNT[9:0] (lines)

Sync Timing



Data Timing

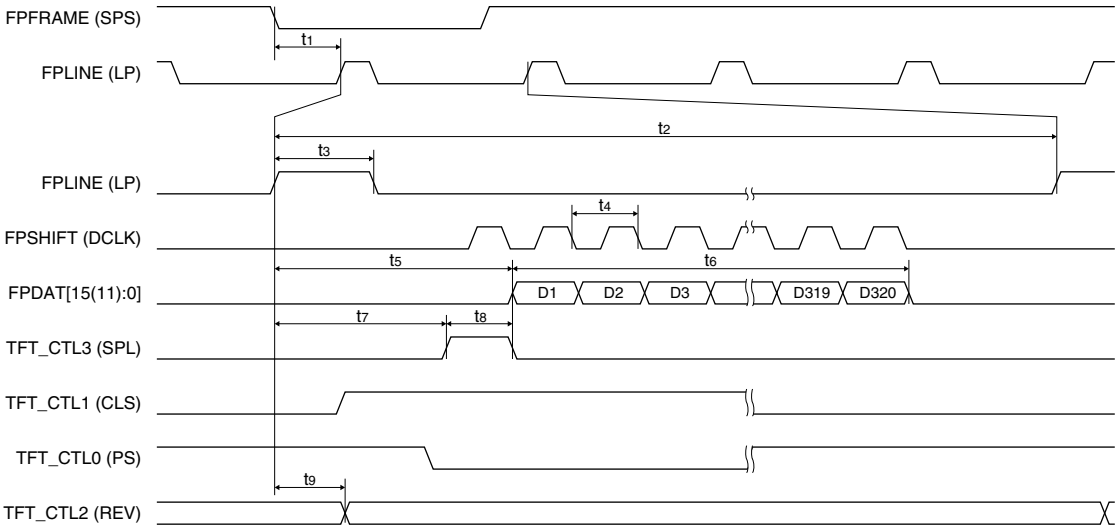


Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	Frame Pulse setup to Line Pulse falling edge	note 2	–	–	(note 1)
t_2	Frame Pulse hold from Line Pulse falling edge	9	–	–	Ts
t_3	Line Pulse period	note 3	–	–	–
t_4	Line Pulse width	9	–	–	Ts
t_5	MOD delay from Line Pulse rising edge	1	–	–	Ts
t_6	Shift Pulse falling edge to Line Pulse rising edge	note 4	–	–	–
t_7	Shift Pulse falling edge to Line Pulse falling edge	note 5	–	–	–
t_8	Line Pulse falling edge to Shift Pulse falling edge	t_{14} + 2	–	–	Ts
t_9	Shift Pulse period	2 (3)	–	–	Ts
t_{10}	Shift Pulse width low	1	–	–	Ts
t_{11}	Shift Pulse width high	1	–	–	Ts
t_{12}	FPDAT[7:0] setup to Shift Pulse falling edge	1	–	–	Ts
t_{13}	FPDAT[7:0] hold from Shift Pulse falling edge	1	–	–	Ts
t_{14}	Line Pulse falling edge to Shift Pulse rising edge	23	–	–	Ts

note) 1. Ts = pixel clock period
2. t_{1min} = t_{3min} - 9 (Ts)
3. t_{3min} = HDP + HNDP (Ts)
4. t_{6min} = HNDP + 1 (Ts)
5. t_{7min} = HNDP + 10 (Ts)

TFT panel timing

(1) TFT panel horizontal timing

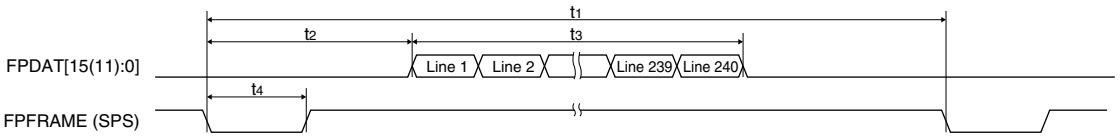


∗ Example timing for a 320 × 240 panel

Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	FPLINE start position	–	note 2	–	(note 1)
t_2	Total horizontal period	400	note 3	440	Ts
t_3	FPLINE pulse width	–	note 4	–	Ts
t_4	FPSHIFT period	–	1	–	Ts
t_5	Horizontal display start position	–	note 5	–	Ts
t_6	Horizontal display period	–	note 6	–	Ts
t_7	FPLINE rising edge to TFT_CTL3 rising edge	–	59	–	Ts
t_8	TFT_CTL3 pulse width	–	1	–	Ts
t_9	FPLINE rising edge to TFT_CTL2 change	–	11	–	Ts

note) 1. Ts = pixel clock period
2. t_{1typ} = FPLINE_ST[9:0] + 1 (Ts)
3. t_{2typ} = (HTCNT[6:0] + 1) × 8 (Ts)
4. t_{3typ} = FPLINE_WD[6:0] + 1 (Ts)
5. t_{5typ} = HDPSCNT[9:0] + 1 (Ts)
6. t_{6typ} = (HDPCNT[6:0] + 1) × 8 (Ts)

(2) TFT panel vertical timing

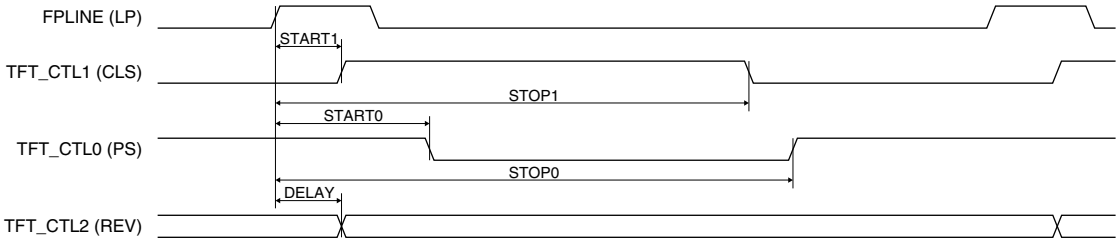


∗ Example timing for a 320 × 240 panel

Symbol	Parameter	Min.	Typ.	Max.	Unit
t_1	Total vertical period	245	note 1	330	Lines
t_2	Vertical display start position	–	note 2	–	Lines
t_3	Vertical display period	–	note 3	–	Lines
t_4	Vertical sync pulse width	–	2	–	Lines

note) 1. t_{1typ} = VTCNT[9:0] + 1 (Lines)
2. t_{2typ} = VDPSCNT[9:0] (Lines)
3. t_{3typ} = VDPCNT[9:0] + 1 (Lines)

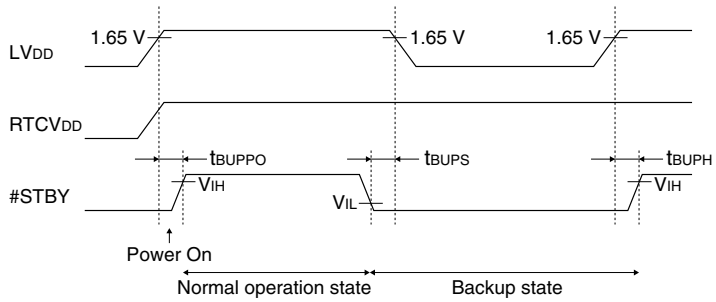
(3) TFT panel control signal offset timings



∗ When FPLINE_ST[9:0] = 0x0

START1 = CTL1ST[7:0] (Ts)
STOP1 = CTL1STP[7:0] + 1 (Ts)
START0 = CTL0ST[7:0] (Ts)
STOP0 = CTL0STP[7:0] + 1 (Ts)
DELAY = CTL2DLY[7:0] (Ts)

32.8.7 #STBY AC Characteristics



Unless otherwise specified: LV_{DD} = $RTCV_{DD}$ = $PLLV_{DD}$ = 1.65 to 1.95V, HV_{DD} = AV_{DD} = 2.7 to 3.6V, V_{SS} = 0V

Item	Symbol	Min.	Typ.	Max.	Unit
#STBY delay time at start of power supply	t_{BUPPO}	0	–	–	ns
Power shutdown time when entering backup status ∗1	t_{BUPS}	1	–	–	ms
Power stabilization time when exiting from backup status ∗2	t_{BUPH}	t_{STA3} + t_{RST}	–	–	ms

∗1) After setting the #STBY pin to low to place the IC into backup state, keep LV_{DD} at 1.65 V or more for at least t_{BUPS}.
∗2) Before setting the #STBY pin to high to cancel backup state, wait at least t_{BUPH} after LV_{DD} goes 1.65 V or higher.

32.9 USB DC and AC Characteristics

Input levels

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = 0 to 70°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
VBUS input voltage ∗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 V_{DI} 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: LV_{DD} = 1.65 to 1.95V, HV_{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: LV_{DD} = 1.65 to 1.95V, HV_{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.

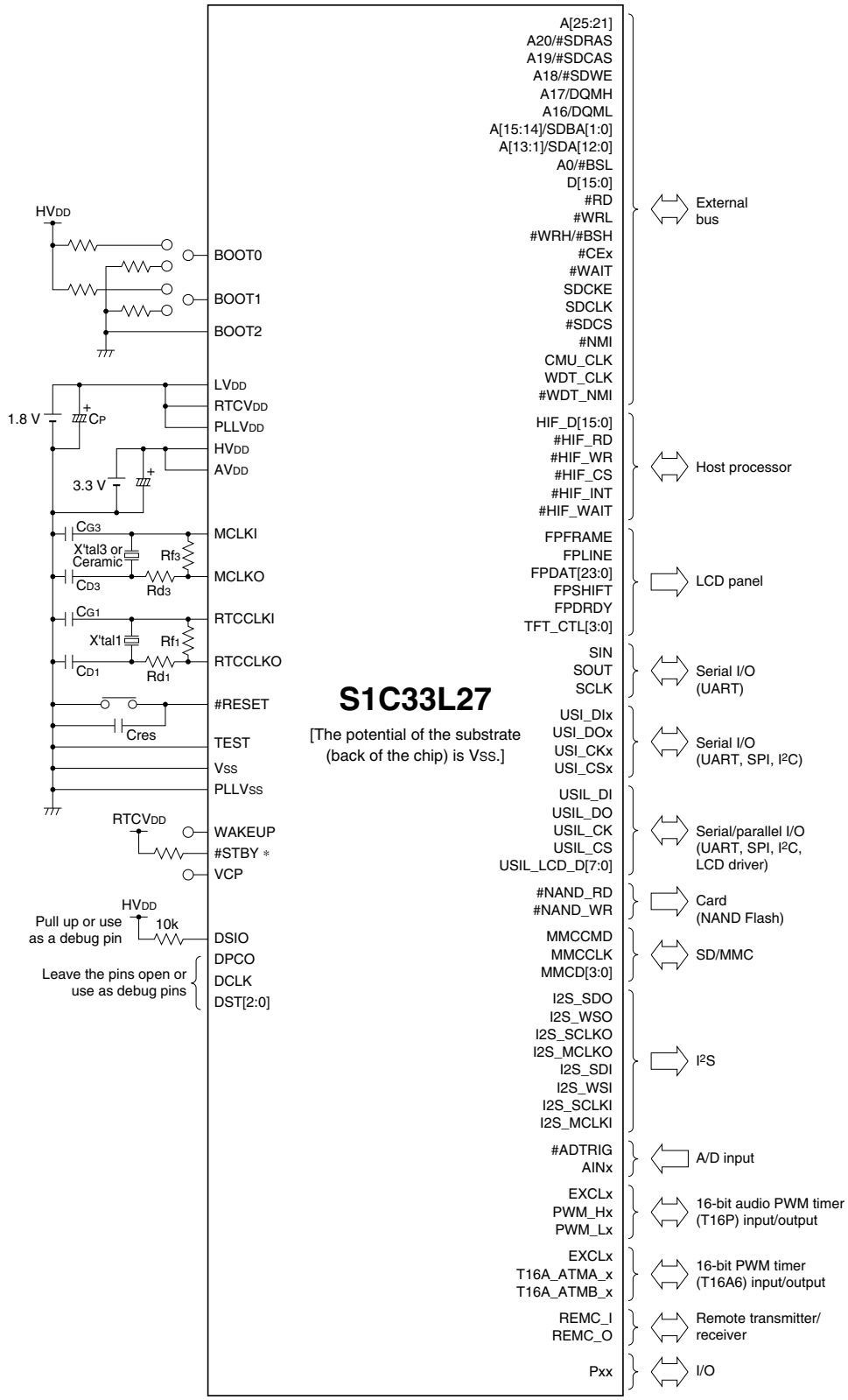
Driver characteristics

Unless otherwise specified: LV_{DD} = 1.65 to 1.95V, HV_{DD} = 3.0 to 3.6V, V_{SS} = 0V, Ta = 0 to 70°C

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Rise time ∗6	T_{FR}		4	–	20	ns
Fall time ∗6	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)			–

∗6) Refer to Figures 7-8 and 7-9 in the USB2.0 Specification for the conditions.

33 Basic External Connection Diagram



* The #STBY pin should be fixed at the RTCVDD level if it is not used for power control.

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_{D3} [pF]	C_{G3} [pF]	R_{f3} [Ω]	R_{d3} [Ω]	Temperature range [°C]
X'tal3	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_{D3} and C_{G3} 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.

Appendix A List of I/O Registers

Peripheral	Address	Register name		Function
MISC registers (MISC) (8-bit device)	0x300010	MISC_RTCWT	RTC Wait Control Register	Configure RTC access cycles
	0x300012	MISC_USB	USB Configuration Register	Enable USB interrupt, snooze, and configure access cycle
	0x300014	MISC_RAMWT	Internal RAM Wait Control Register	Configure internal RAM access cycles
	0x300016	MISC_BOOT	Boot Register	Indicate/set boot conditions
	0x300018	MISC_TRACE	Debug Trace Enable Register	Enable PC trace debugging
	0x300020	MISC_PROT0	Misc Protect Register 0	Enable/disable Misc register write protection
	0x300022	MISC_PROT1	Misc Protect Register 1	
	0x300024	MISC_RAMCFG	RAM Configuration Register	Configure internal RAM
Clock management unit (CMU) (8-bit device)	0x300100	CMU_OSCSEL	Clock Source Select Register	Select system clock source
	0x300101	CMU_OSCCTL	Oscillation Control Register	Control oscillators
	0x300103	CMU_LCLKDIV	LCDC Clock Division Ratio Select Register	Set LCDC_CLK frequency
	0x300105	CMU_SYSCLKDIV	System Clock Division Ratio Select Register	Set system clock frequency
	0x300106	CMU_CMUCLK	CMU_CLK Select Register	Select CMU_CLK output clock
	0x300107	CMU_PLLINDIV	PLL Input Clock Division Ratio Select Register	Set PLL input clock frequency
	0x300108	CMU_PLLCTL0	PLL Control Register 0	Set PLL multiplication rate and enable PLL
	0x300109	CMU_PLLCTL1	PLL Control Register 1	Set PLL parameters
	0x30010a	CMU_PLLCTL2	PLL Control Register 2	
	0x30010c	CMU_SSCG0	SSCG Macro Control Register 0	Enable SSCG
	0x30010d	CMU_SSCG1	SSCG Macro Control Register 1	Set SSCG parameters
	0x300110	CMU_PROTECT	CMU Write Protect Register	Enable/disable CMU register write protection
	0x300114	CMU_CLKCTL0	Clock Control Register 0	Control clock supply to peripheral/bus modules
	0x300115	CMU_CLKCTL1	Clock Control Register 1	
	0x300116	CMU_CLKCTL2	Clock Control Register 2	
	0x300117	CMU_CLKCTL3	Clock Control Register 3	
	0x300118	CMU_CLKCTL4	Clock Control Register 4	
Interrupt controller (ITC) (8-bit device)	0x300210	ITC_PINT0_LV	Port Input 0 Interrupt Level Register	Set port input 0 interrupt level
	0x300211	ITC_PINT1_LV	Port Input 1 Interrupt Level Register	Set port input 1 interrupt level
	0x300212	ITC_KINT0_LV	Key Input 0 Interrupt Level Register	Set key input 0 interrupt level
	0x300213	ITC_KINT1_LV	Key Input 1 Interrupt Level Register	Set key input 1 interrupt level
	0x300214	ITC_DMA02_LV	DMAC Ch.0/2 Interrupt Level Register	Set DMAC Ch.0 and 2 interrupt levels
	0x300215	ITC_DMA13_LV	DMAC Ch.1/3 Interrupt Level Register	Set DMAC Ch.1 and 3 interrupt levels
	0x300216	ITC_DMA46_LV	DMAC Ch.4/6 Interrupt Level Register	Set DMAC Ch.4 and 6 interrupt levels
	0x300217	ITC_DMA57_LV	DMAC Ch.5/7 Interrupt Level Register	Set DMAC Ch.5 and 7 interrupt levels
	0x300218	ITC_T16P0_LV	T16P Ch.0 Interrupt Level Register	Set T16P Ch.0 interrupt level
	0x300219	ITC_T16P1_LV	T16P Ch.1 Interrupt Level Register	Set T16P Ch.1 interrupt level
	0x30021a	ITC_T16A0_LV	T16A6 Ch.0 Interrupt Level Register	Set T16A6 Ch.0 interrupt level
	0x30021b	ITC_T16A1_LV	T16A6 Ch.1 Interrupt Level Register	Set T16A6 Ch.1 interrupt level
	0x30021c	ITC_T16A2_LV	T16A6 Ch.2 Interrupt Level Register	Set T16A6 Ch.2 interrupt level
	0x30021d	ITC_T16A3_LV	T16A6 Ch.3 Interrupt Level Register	Set T16A6 Ch.3 interrupt level
	0x30021e	ITC_T8F024_LV	T8F Ch.0/2/4 Interrupt Level Register	Set T8F Ch.0, 2, and 4 interrupt levels
	0x30021f	ITC_T8F135_LV	T8F Ch.1/3/5 Interrupt Level Register	Set T8F Ch.1, 3, and 5 interrupt levels
	0x300220	ITC_ADC10_LV	ADC10 Interrupt Level Register	Set ADC10 interrupt level
	0x300221	ITC_RTC_LV	RTC Interrupt Level Register	Set RTC interrupt level
	0x300222	ITC_USI0_LV	USI Ch.0/UART Interrupt Level Register	Set USI Ch.0 and UART interrupt levels
	0x300223	ITC_USI1_LV	USI Ch.1 Interrupt Level Register	Set USI Ch.1 interrupt level
	0x300224	ITC_USI2_LV	USI Ch.2 Interrupt Level Register	Set USI Ch.2 interrupt level
	0x300225	ITC_USIL_LV	USIL Interrupt Level Register	Set USIL interrupt level
	0x300226	ITC_LCDC_LV	LCDC Interrupt Level Register	Set LCDC interrupt level
	0x300227	ITC_SDMMC_LV	SD_MMC Interrupt Level Register	Set SD_MMC interrupt level
	0x300228	ITC_REMC_LV	REMC Interrupt Level Register	Set REMC interrupt level
	0x300229	ITC_I2SOUT_LV	I²S Output Interrupt Level Register	Set I²S output channel interrupt level
	0x30022a	ITC_I2SIN_LV	I²S Input Interrupt Level Register	Set I²S input channel interrupt level
	0x30022b	ITC_HIF_LV	HIF Interrupt Level Register	Set HIF interrupt level
	0x30022c	ITC_USB_LV	USB Interrupt Level Register	Set USB interrupt level
GPIO & port MUX (GPIO/PMUX) (8-bit device)	0x300300	GPIO_P0_DAT	P0 Port Data Register	P0 port input/output data
	0x300301	GPIO_P0_IOC	P0 Port I/O Control Register	Control P0 port input/output direction
	0x300302	GPIO_P1_DAT	P1 Port Data Register	P1 port input/output data
	0x300303	GPIO_P1_IOC	P1 Port I/O Control Register	Control P1 port input/output direction
	0x300304	GPIO_P2_DAT	P2 Port Data Register	P2 port input/output data
	0x300305	GPIO_P2_IOC	P2 Port I/O Control Register	Control P2 port input/output direction
	0x300306	GPIO_P3_DAT	P3 Port Data Register	P3 port input/output data
	0x300307	GPIO_P3_IOC	P3 Port I/O Control Register	Control P3 port input/output direction

Peripheral	Address	Register name		Function
GPIO & port MUX (GPIO/PMUX) (8-bit device)	0x300308	GPIO_P4_DAT	P4 Port Data Register	P4 port input/output data
	0x300309	GPIO_P4_IOC	P4 Port I/O Control Register	Control P4 port input/output direction
	0x30030a	GPIO_P5_DAT	P5 Port Data Register	P5 port input/output data
	0x30030b	GPIO_P5_IOC	P5 Port I/O Control Register	Control P5 port input/output direction
	0x30030c	GPIO_P6_DAT	P6 Port Data Register	P6 port input/output data
	0x30030d	GPIO_P6_IOC	P6 Port I/O Control Register	Control P6 port input/output direction
	0x30030e	GPIO_P7_DAT	P7 Port Data Register	P7 port input data
	0x300310	GPIO_P8_DAT	P8 Port Data Register	P8 port input/output data
	0x300311	GPIO_P8_IOC	P8 Port I/O Control Register	Control P8 port input/output direction
	0x300312	GPIO_P9_DAT	P9 Port Data Register	P9 port input/output data
	0x300313	GPIO_P9_IOC	P9 Port I/O Control Register	Control P9 port input/output direction
	0x300314	GPIO_PA_DAT	PA Port Data Register	PA port input/output data
	0x300315	GPIO_PA_IOC	PA Port I/O Control Register	Control PA port input/output direction
	0x300316	GPIO_PB_DAT	PB Port Data Register	PB port input/output data
	0x300317	GPIO_PB_IOC	PB Port I/O Control Register	Control PB port input/output direction
	0x300318	GPIO_PC_DAT	PC Port Data Register	PC port input/output data
	0x300319	GPIO_PC_IOC	PC Port I/O Control Register	Control PC port input/output direction
	0x30031a	GPIO_PD_DAT	PD Port Data Register	PD port input/output data
	0x30031b	GPIO_PD_IOC	PD Port I/O Control Register	Control PD port input/output direction
	0x30031c	GPIO_PE_DAT	PE Port Data Register	PE port input/output data
	0x30031d	GPIO_PE_IOC	PE Port I/O Control Register	Control PE port input/output direction
	0x30031e	GPIO_PF_DAT	PF Port Data Register	PF port input/output data
	0x30031f	GPIO_PF_IOC	PF Port I/O Control Register	Control PF port input/output direction
	0x300340	GPIO_FPT01_SEL	FPT0–1 Interrupt Port Select Register	Select ports used for FPT0–1 interrupts
	0x300341	GPIO_FPT23_SEL	FPT2–3 Interrupt Port Select Register	Select ports used for FPT2–3 interrupts
	0x300342	GPIO_FPT45_SEL	FPT4–5 Interrupt Port Select Register	Select ports used for FPT4–5 interrupts
	0x300343	GPIO_FPT67_SEL	FPT6–7 Interrupt Port Select Register	Select ports used for FPT6–7 interrupts
	0x300344	GPIO_FPT03_POL	FPT0–3 Interrupt Polarity Select Register	Select input signal polarity for FPT0–3 interrupts
	0x300345	GPIO_FPT47_POL	FPT4–7 Interrupt Polarity Select Register	Select input signal polarity for FPT4–7 interrupts
	0x300346	GPIO_FPT03_MOD	FPT0–3 Interrupt Mode Select Register	Select edge/level mode for FPT0–3 interrupts
	0x300347	GPIO_FPT47_MOD	FPT4–7 Interrupt Mode Select Register	Select edge/level mode for FPT4–7 interrupts
	0x300348	GPIO_FPT03_MSK	FPT0–3 Interrupt Mask Register	Enable/disable FPT0–3 interrupts
	0x300349	GPIO_FPT47_MSK	FPT4–7 Interrupt Mask Register	Enable/disable FPT4–7 interrupts
	0x30034a	GPIO_FPT03_FLG	FPT0–3 Interrupt Flag Register	Indicate FPT0–3 interrupt cause status
	0x30034b	GPIO_FPT47_FLG	FPT4–7 Interrupt Flag Register	Indicate FPT4–7 interrupt cause status
	0x30034c	GPIO_FPT01_CHAT	FPT0–1 Interrupt Chattering Filter Control Register	Control FPT0–1 chattering filter
	0x30034d	GPIO_FPT23_CHAT	FPT2–3 Interrupt Chattering Filter Control Register	Control FPT2–3 chattering filter
	0x30034e	GPIO_FPT45_CHAT	FPT4–5 Interrupt Chattering Filter Control Register	Control FPT4–5 chattering filter
	0x30034f	GPIO_FPT67_CHAT	FPT6–7 Interrupt Chattering Filter Control Register	Control FPT6–7 chattering filter
	0x300350	GPIO_DMA	Port DMA Trigger Source Select Register	Select port DMA trigger source
	0x300360	GPIO_FPK0_07_SEL	FPK000–007 Interrupt Select Register	Select ports used for FPK0 interrupts
	0x300361	GPIO_FPK0_8F_SEL	FPK008–015 Interrupt Select Register	
	0x300362	GPIO_FPK1_07_SEL	FPK100–107 Interrupt Select Register	Select ports used for FPK1 interrupts
	0x300363	GPIO_FPK1_8F_SEL	FPK108–115 Interrupt Select Register	
	0x300370	GPIO_FPK01_POL	FPK0–1 Interrupt Polarity Select Register	Select input signal polarity for FPK0–1 interrupts
	0x300371	GPIO_FPK01_MOD	FPK0–1 Interrupt Mode Select Register	Select edge/level mode for FPK0–1 interrupts
	0x300372	GPIO_FPK01_MSK	FPK0–1 Interrupt Mask Register	Enable/disable FPK0–1 interrupts
	0x300373	GPIO_FPK01_FLG	FPK0–1 Interrupt Flag Register	Indicate FPK0–1 interrupt cause status
	0x300374	GPIO_FPK01_CHAT	FPK0–1 Interrupt Chattering Filter Control Register	Control FPK0–1 chattering filter
	0x300800	PMUX_P0_03	P0[3:0] Port Function Select Register	Select P0[3:0] port functions
	0x300801	PMUX_P0_47	P0[7:4] Port Function Select Register	Select P0[7:4] port functions
	0x300802	PMUX_P1_03	P1[3:0] Port Function Select Register	Select P1[3:0] port functions
	0x300803	PMUX_P1_47	P1[7:4] Port Function Select Register	Select P1[7:4] port functions
	0x300804	PMUX_P2_01	P2[1:0] Port Function Select Register	Select P2[1:0] port functions
	0x300806	PMUX_P3_03	P3[3:0] Port Function Select Register	Select P3[3:0] port functions
	0x300807	PMUX_P3_46	P3[6:4] Port Function Select Register	Select P3[6:4] port functions
	0x300808	PMUX_P4_02	P4[2:0] Port Function Select Register	Select P4[2:0] port functions
	0x30080a	PMUX_P5_03	P5[3:0] Port Function Select Register	Select P5[3:0] port functions
	0x30080b	PMUX_P5_46	P5[6:4] Port Function Select Register	Select P5[6:4] port functions
	0x30080c	PMUX_P6_0	P60 Port Function Select Register	Select P60 port function

Peripheral	Address	Register name		Function
GPIO & port MUX (GPIO/PMUX) (8-bit device)	0x30080e	PMUX_P7_03	P7[3:0] Port Function Select Register	Select P7[3:0] port functions
	0x30080f	PMUX_P7_47	P7[7:4] Port Function Select Register	Select P7[7:4] port functions
	0x300810	PMUX_P8_03	P8[3:0] Port Function Select Register	Select P8[3:0] port functions
	0x300812	PMUX_P9_03	P9[3:0] Port Function Select Register	Select P9[3:0] port functions
	0x300813	PMUX_P9_47	P9[7:4] Port Function Select Register	Select P9[7:4] port functions
	0x300814	PMUX_PA_03	PA[3:0] Port Function Select Register	Select PA[3:0] port functions
	0x300815	PMUX_PA_47	PA[7:4] Port Function Select Register	Select PA[7:4] port functions
	0x300816	PMUX_PB_03	PB[3:0] Port Function Select Register	Select PB[3:0] port functions
	0x300817	PMUX_PB_47	PB[7:4] Port Function Select Register	Select PB[7:4] port functions
	0x300818	PMUX_PC_03	PC[3:0] Port Function Select Register	Select PC[3:0] port functions
	0x300819	PMUX_PC_47	PC[7:4] Port Function Select Register	Select PC[7:4] port functions
	0x30081a	PMUX_PD_03	PD[3:0] Port Function Select Register	Select PD[3:0] port functions
	0x30081b	PMUX_PD_47	PD[7:4] Port Function Select Register	Select PD[7:4] port functions
	0x30081c	PMUX_PE_03	PE[3:0] Port Function Select Register	Select PE[3:0] port functions
	0x30081d	PMUX_PE_45	PE[5:4] Port Function Select Register	Select PE[5:4] port functions
	0x30081e	PMUX_PF_03	PF[3:0] Port Function Select Register	Select PF[3:0] port functions
	0x30081f	PMUX_PF_4	PF4 Port Function Select Register	Select PF4 port function
	0x300820	GPIO_P0_PUP	P0 Port Pull-up Control Register	Enable/disable P0 port pull-up resistors
	0x300821	GPIO_P1_PUP	P1 Port Pull-up Control Register	Enable/disable P1 port pull-up resistors
	0x300822	GPIO_P2_PUP	P2 Port Pull-up Control Register	Enable/disable P2 port pull-up resistors
	0x300823	GPIO_P3_PUP	P3 Port Pull-up Control Register	Enable/disable P3 port pull-up resistors
	0x300824	GPIO_P4_PUP	P4 Port Pull-up Control Register	Enable/disable P4 port pull-up resistors
	0x300825	GPIO_P5_PUP	P5 Port Pull-up Control Register	Enable/disable P5 port pull-up resistors
	0x300826	GPIO_P6_PUP	P6 Port Pull-up Control Register	Enable/disable P6 port pull-up resistors
	0x300827	GPIO_P7_PUP	P7 Port Pull-up Control Register	Enable/disable P7 port pull-up resistors
	0x300828	GPIO_P8_PUP	P8 Port Pull-up Control Register	Enable/disable P8 port pull-up resistors
	0x300829	GPIO_P9_PUP	P9 Port Pull-up Control Register	Enable/disable P9 port pull-up resistors
	0x30082a	GPIO_PA_PUP	PA Port Pull-up Control Register	Enable/disable PA port pull-up resistors
	0x30082b	GPIO_PB_PUP	PB Port Pull-up Control Register	Enable/disable PB port pull-up resistors
	0x30082d	GPIO_PD_PUP	PD Port Pull-up Control Register	Enable/disable PD port pull-up resistors
	0x30082e	GPIO_PE_PUP	PE Port Pull-up Control Register	Enable/disable PE port pull-up resistors
	0x30082f	GPIO_PF_PUP	PF Port Pull-up Control Register	Enable/disable PF port pull-up resistors
	0x300830	GPIO_BUS_DRV	Bus Drive Control Register	Set external data and address bus signals to low
	0x30083e	GPIO_FILTER	Port Noise Filter Control Register	Enable/disable port input noise filter
	0x30083f	GPIO_PROTECT	GPIO/PMUX Write Protect Register	Enable/disable write protection for PMUX, GPIO_BUS_DRV, GPIO_FILTER, and GPIO_P*x*_PUP registers
USI Ch.0 (8-bit device)	0x300400	USI_GCFG0	USI Ch.0 Global Configuration Register	Set interface and MSB/LSB modes
	0x300401	USI_TD0	USI Ch.0 Transmit Data Buffer Register	Transmit data buffer
	0x300402	USI_RD0	USI Ch.0 Receive Data Buffer Register	Receive data buffer
	0x300440	USI_UCFG0	USI Ch.0 UART Mode Configuration Register	Set UART transfer conditions
	0x300441	USI_UIE0	USI Ch.0 UART Mode Interrupt Enable Register	Enable/disable UART interrupts
	0x300442	USI_UIF0	USI Ch.0 UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
	0x300450	USI_SCFG0	USI Ch.0 SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
	0x300451	USI_SIE0	USI Ch.0 SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
	0x300452	USI_SIF0	USI Ch.0 SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
	0x300460	USI_IMTG0	USI Ch.0 I²C Master Mode Trigger Register	Start I²C master operations
	0x300461	USI_IMIE0	USI Ch.0 I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
	0x300462	USI_IMIF0	USI Ch.0 I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
	0x300470	USI_ISTG0	USI Ch.0 I²C Slave Mode Trigger Register	Start I²C slave operations
	0x300471	USI_ISIE0	USI Ch.0 I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
	0x300472	USI_ISIF0	USI Ch.0 I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status
USI Ch.1 (8-bit device)	0x300500	USI_GCFG1	USI Ch.1 Global Configuration Register	Set interface and MSB/LSB modes
	0x300501	USI_TD1	USI Ch.1 Transmit Data Buffer Register	Transmit data buffer
	0x300502	USI_RD1	USI Ch.1 Receive Data Buffer Register	Receive data buffer
	0x300540	USI_UCFG1	USI Ch.1 UART Mode Configuration Register	Set UART transfer conditions
	0x300541	USI_UIE1	USI Ch.1 UART Mode Interrupt Enable Register	Enable/disable UART interrupts
	0x300542	USI_UIF1	USI Ch.1 UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
	0x300550	USI_SCFG1	USI Ch.1 SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
	0x300551	USI_SIE1	USI Ch.1 SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
	0x300552	USI_SIF1	USI Ch.1 SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status

Peripheral	Address	Register name		Function
USI Ch.1 (8-bit device)	0x300560	USI_IMTG1	USI Ch.1 I²C Master Mode Trigger Register	Start I²C master operations
	0x300561	USI_IMIE1	USI Ch.1 I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
	0x300562	USI_IMIF1	USI Ch.1 I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
	0x300570	USI_ISTG1	USI Ch.1 I²C Slave Mode Trigger Register	Start I²C slave operations
	0x300571	USI_ISIE1	USI Ch.1 I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
	0x300572	USI_ISIF1	USI Ch.1 I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status
USIL (8-bit device)	0x300600	USIL_GCFG	USIL Global Configuration Register	Set interface and MSB/LSB modes
	0x300601	USIL_TD	USIL Transmit Data Buffer Register	Transmit data buffer
	0x300602	USIL_RD	USIL Receive Data Buffer Register	Receive data buffer
	0x300640	USIL_UCFG	USIL UART Mode Configuration Register	Set UART transfer conditions
	0x300641	USIL_UIE	USIL UART Mode Interrupt Enable Register	Enable/disable UART interrupts
	0x300642	USIL_UIF	USIL UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
	0x300650	USIL_SCFG	USIL SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
	0x300651	USIL_SIE	USIL SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
	0x300652	USIL_SIF	USIL SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
	0x300660	USIL_IMTG	USIL I²C Master Mode Trigger Register	Start I²C master operations
	0x300661	USIL_IMIE	USIL I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
	0x300662	USIL_IMIF	USIL I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
	0x300670	USIL_ISTG	USIL I²C Slave Mode Trigger Register	Start I²C slave operations
	0x300671	USIL_ISIE	USIL I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
	0x300672	USIL_ISIF	USIL I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status
	0x300680	USIL_LSCFG	USIL LCD SPI Mode Configuration Register	Set LCD SPI transfer conditions
	0x300681	USIL_LSIE	USIL LCD SPI Mode Interrupt Enable Register	Enable/disable LCD SPI interrupts
	0x300682	USIL_LSIF	USIL LCD SPI Mode Interrupt Flag Register	Indicate LCD SPI interrupt cause status
	0x30068f	USIL_LSDCFG	USIL LCD SPI Mode Data Configuration Register	Select display data format
	0x300690	USIL_LPCFG	USIL LCD Parallel I/F Mode Configuration Register	Set LCD parallel interface conditions
	0x300691	USIL_LPIE	USIL LCD Parallel I/F Mode Interrupt Enable Register	Enable/disable LCD parallel interface interrupts
	0x300692	USIL_LPIF	USIL LCD Parallel I/F Mode Interrupt Flag Register	Indicate LCD parallel interface interrupt cause status
	0x30069f	USIL_LPAC	USIL LCD Parallel I/F Mode Access Timing Register	Set LCD parallel interface access timing parameters
USI Ch.2 (8-bit device)	0x300700	USI_GCFG2	USI Ch.2 Global Configuration Register	Set interface and MSB/LSB modes
	0x300701	USI_TD2	USI Ch.2 Transmit Data Buffer Register	Transmit data buffer
	0x300702	USI_RD2	USI Ch.2 Receive Data Buffer Register	Receive data buffer
	0x300740	USI_UCFG2	USI Ch.2 UART Mode Configuration Register	Set UART transfer conditions
	0x300741	USI_UIE2	USI Ch.2 UART Mode Interrupt Enable Register	Enable/disable UART interrupts
	0x300742	USI_UIF2	USI Ch.2 UART Mode Interrupt Flag Register	Indicate UART interrupt cause status
	0x300750	USI_SCFG2	USI Ch.2 SPI Master/Slave Mode Configuration Register	Set SPI transfer conditions
	0x300751	USI_SIE2	USI Ch.2 SPI Master/Slave Mode Interrupt Enable Register	Enable/disable SPI interrupts
	0x300752	USI_SIF2	USI Ch.2 SPI Master/Slave Mode Interrupt Flag Register	Indicate SPI interrupt cause status
	0x300760	USI_IMTG2	USI Ch.2 I²C Master Mode Trigger Register	Start I²C master operations
	0x300761	USI_IMIE2	USI Ch.2 I²C Master Mode Interrupt Enable Register	Enable/disable I²C master interrupts
	0x300762	USI_IMIF2	USI Ch.2 I²C Master Mode Interrupt Flag Register	Indicate I²C master interrupt cause status
	0x300770	USI_ISTG2	USI Ch.2 I²C Slave Mode Trigger Register	Start I²C slave operations
	0x300771	USI_ISIE2	USI Ch.2 I²C Slave Mode Interrupt Enable Register	Enable/disable I²C slave interrupts
	0x300772	USI_ISIF2	USI Ch.2 I²C Slave Mode Interrupt Flag Register	Indicate I²C slave interrupt cause status
Host Interface (HIF) (8- or 16-bit device)	Host 0x0	HIF_DRIF_B	HIF Direct Interface Byte Data Register	Byte data direct read/write register
	Host 0x1	HIF_DRIF_HW	HIF Direct Interface Half Word Data Register	Half word data direct read/write register
	Host 0x2	HIF_DRIF_W	HIF Direct Interface Word Data Register	Word data direct read/write register
	Host 0x3	HIF_IDIF_DAT	HIF Indirect Interface Data Register	Data indirect read/write register
	Host 0x4	HIF_IDIF_CTRL	HIF Indirect Interface Control Register	Control data indirect reading/writing
	Host 0x6	HIF_CTRL	HIF Control Register	Set access conditions
	Host 0x8	HIF_ADDR0	HIF Address Register 0	Address bits A[7:0] (8-bit I/F) or A[15:0] (16-bit I/F)
	Host 0x9	HIF_ADDR1	HIF Address Register 1	Address bits A[15:8] (8-bit I/F) or A[31:16] (16-bit I/F)

Peripheral	Address	Register name		Function
Host Interface (HIF) (8- or 16-bit device)	Host 0xa	HIF_ADDR2	HIF Address Register 2	Address bits A[23:16] (8-bit I/F)
	Host 0xb	HIF_ADDR3	HIF Address Register 3	Address bits A[31:24] (8-bit I/F)
	Host 0xc	HIF_SP_IDX	HIF Special Register Index Register	Specify special register
	Host 0xd	HIF_SP_DAT	HIF Special Register Data Register	Special register data
Host Interface (HIF) (8-bit device)	0xd[0x1] /0x300901	HIF_EXTCTRL1	HIF Extended Control Register 1	Set interrupt mode
	0xd[0x8] /0x300908	HIF_SMPH0	HIF Semaphore Register 0	Semaphore 0
	0xd[0x9] /0x300909	HIF_SMPH1	HIF Semaphore Register 1	Semaphore 1
	0xd[0xa] /0x30090a	HIF_SMPH2	HIF Semaphore Register 2	Semaphore 2
	0xd[0xb] /0x30090b	HIF_SMPH3	HIF Semaphore Register 3	Semaphore 3
	0xd[0xc] /0x30090c	HIF_SMPH4	HIF Semaphore Register 4	Semaphore 4
	0xd[0xd] /0x30090d	HIF_SMPH5	HIF Semaphore Register 5	Semaphore 5
	0xd[0xe] /0x30090e	HIF_SMPH6	HIF Semaphore Register 6	Semaphore 6
	0xd[0xf] /0x30090f	HIF_SMPH7	HIF Semaphore Register 7	Semaphore 7
	0xd[0x10] /0x300910	HIF_SCRATCH0	HIF Scratchpad Register 0	Scratchpad register 0
	0xd[0x11] /0x300911	HIF_SCRATCH1	HIF Scratchpad Register 1	Scratchpad register 1
	0xd[0x12] /0x300912	HIF_SCRATCH2	HIF Scratchpad Register 2	Scratchpad register 2
	0xd[0x13] /0x300913	HIF_SCRATCH3	HIF Scratchpad Register 3	Scratchpad register 3
Host Interface (HIF) (8- or 16-bit device)	Host 0xe /0x300918	HIF_FLAG0	HIF Flag Register 0	Slave flag
	Host 0xf /0x300919	HIF_FLAG1	HIF Flag Register 1	Host flag
Real-Time Clock (RTC) (8-bit device)	0x300a00	RTC_INTSTAT	RTC Interrupt Status Register	Indicate RTC interrupt cause status
	0x300a01	RTC_INTMODE	RTC Interrupt Mode Register	Set RTC interrupt modes
	0x300a02	RTC_CNTL0	RTC Control 0 Register	Control RTC
	0x300a03	RTC_CNTL1	RTC Control 1 Register	
	0x300a04	RTC_SEC	RTC Second Register	Second counter data
	0x300a05	RTC_MIN	RTC Minute Register	Minute counter data
	0x300a06	RTC_HOUR	RTC Hour Register	Hour counter data
	0x300a07	RTC_DAY	RTC Day Register	Day counter data
	0x300a08	RTC_MONTH	RTC Month Register	Month counter data
	0x300a09	RTC_YEAR	RTC Year Register	Year counter data
	0x300a0a	RTC_WEEK	RTC Days of Week Register	Days of week counter data
	0x300a0f	RTC_WAKEUP	RTC Wakeup Configuration Register	Set RTC wakeup conditions
BBRAM (8-bit device)	0x300b00 | 0x300b0f	BBRAM_0 | BBRAM_15	BBRAM byte 0 | BBRAM byte 15	BBRAM
USB function controller (USB) (8-bit device)	0x300c00	MainIntStat	Main Interrupt Status Register	Indicate main interrupt status
	0x300c01	SIE_IntStat	SIE Interrupt Status Register	Indicate SIE interrupt status
	0x300c02	EPrIntStat	EPr Interrupt Status Register	Indicate EPr interrupt status
	0x300c03	DMA_IntStat	DMA Interrupt Status Register	Indicate DMA interrupt status
	0x300c04	FIFO_IntStat	FIFO Interrupt Status Register	Indicate FIFO interrupt status
	0x300c07	EP0IntStat	EP0 Interrupt Status Register	Indicate EP0 interrupt status
	0x300c08	EPaIntStat	EPa Interrupt Status Register	Indicate EPa interrupt status
	0x300c09	EPbIntStat	EPb Interrupt Status Register	Indicate EPb interrupt status
	0x300c0a	EPcIntStat	EPc Interrupt Status Register	Indicate EPc interrupt status
	0x300c0b	EPdIntStat	EPd Interrupt Status Register	Indicate EPd interrupt status
	0x300c10	MainIntEnb	Main Interrupt Enable Register	Enable main interrupts
	0x300c11	SIE_IntEnb	SIE Interrupt Enable Register	Enable SIE interrupts
	0x300c12	EPrIntEnb	EPr Interrupt Enable Register	Enable EPr interrupts
	0x300c13	DMA_IntEnb	DMA Interrupt Enable Register	Enable DMA interrupts
	0x300c14	FIFO_IntEnb	FIFO Interrupt Enable Register	Enable FIFO interrupts
	0x300c17	EP0IntEnb	EP0 Interrupt Enable Register	Enable EP0 interrupts
	0x300c18	EPaIntEnb	EPa Interrupt Enable Register	Enable EPa interrupts
	0x300c19	EPbIntEnb	EPb Interrupt Enable Register	Enable EPb interrupts
	0x300c1a	EPcIntEnb	EPc Interrupt Enable Register	Enable EPc interrupts
	0x300c1b	EPdIntEnb	EPd Interrupt Enable Register	Enable EPd interrupts
	0x300c20	RevisionNum	Revision Number Register	Indicate revision number of USB controller

Peripheral	Address	Register name		Function
USB function controller (USB) (8-bit device)	0x300c21	USB_Control	USB Control Register	Control USB conditions
	0x300c22	USB_Status	USB Status Register	Indicate USB status
	0x300c23	XcvrControl	Xcvr Control Register	Control transceiver macro
	0x300c24	USB_Test	USB Test Register	Set USB test mode
	0x300c25	EPnControl	Endpoint Control Register	Clear all FIFOs and set NAK/STALL bits
	0x300c26	EPrFIFO_Clr	EPr FIFO Clear Register	Clear each FIFO
	0x300c2e	FrameNumber_H	Frame Number High Register	Frame number
	0x300c2f	FrameNumber_L	Frame Number Low Register	
	0x300c30	EP0Setup_0	EP0 Setup 0 Register	EP0 setup data (BmRequestType)
	0x300c31	EP0Setup_1	EP0 Setup 1 Register	EP0 setup data (BRequest)
	0x300c32	EP0Setup_2	EP0 Setup 2 Register	EP0 setup data (low-order Wvalue bits)
	0x300c33	EP0Setup_3	EP0 Setup 3 Register	EP0 setup data (high-order Wvalue bits)
	0x300c34	EP0Setup_4	EP0 Setup 4 Register	EP0 setup data (low-order WIndex bits)
	0x300c35	EP0Setup_5	EP0 Setup 5 Register	EP0 setup data (high-order WIndex bits)
	0x300c36	EP0Setup_6	EP0 Setup 6 Register	EP0 setup data (low-order WLength bits)
	0x300c37	EP0Setup_7	EP0 Setup 7 Register	EP0 setup data (high-order WLength bits)
	0x300c38	USB_Address	USB Address Register	Set USB address
	0x300c39	EP0Control	EP0 Control Register	Set up EP0
	0x300c3a	EP0ControlIN	EP0 Control In Register	Set EP0 IN transaction conditions
	0x300c3b	EP0ControlOUT	EP0 Control Out Register	Set EP0 OUT transaction conditions
	0x300c3f	EP0MaxSize	EP0 Max Packet Size Register	Set the EP0 max packet size
	0x300c40	EPaControl	EPa Control Register	Set up EPa
	0x300c41	EPbControl	EPb Control Register	Set up EPb
	0x300c42	EPcControl	EPc Control Register	Set up EPc
	0x300c43	EPdControl	EPd Control Register	Set up EPd
	0x300c50	EPaMaxSize_H	EPa Max Packet Size High Register	Set EPa max packet size
	0x300c51	EPaMaxSize_L	EPa Max Packet Size Low Register	
	0x300c52	EPaConfig_0	EPa Configuration 0 Register	Configure EPa
	0x300c53	EPaConfig_1	EPa Configuration 1 Register	
	0x300c54	EPbMaxSize_H	EPb Max Packet Size High Register	Set EPb max packet size
	0x300c55	EPbMaxSize_L	EPb Max Packet Size Low Register	
	0x300c56	EPbConfig_0	EPb Configuration 0 Register	Configure EPb
	0x300c57	EPbConfig_1	EPb Configuration 1 Register	
	0x300c58	EPcMaxSize_H	EPc Max Packet Size High Register	Set EPc max packet size
	0x300c59	EPcMaxSize_L	EPc Max Packet Size Low Register	
	0x300c5a	EPcConfig_0	EPc Configuration 0 Register	Configure EPc
	0x300c5b	EPcConfig_1	EPc Configuration 1 Register	
	0x300c5c	EPdMaxSize_H	EPd Max Packet Size High Register	Set EPd max packet size
	0x300c5d	EPdMaxSize_L	EPd Max Packet Size Low Register	
	0x300c5e	EPdConfig_0	EPd Configuration 0 Register	Configure EPd
	0x300c5f	EPdConfig_1	EPd Configuration 1 Register	
	0x300c70	EPaStartAdrs_H	EPa FIFO Start Address High Register	Set FIFO start address for EPa
	0x300c71	EPaStartAdrs_L	EPa FIFO Start Address Low Register	
	0x300c72	EPbStartAdrs_H	EPb FIFO Start Address High Register	Set FIFO start address for EPb
	0x300c73	EPbStartAdrs_L	EPb FIFO Start Address Low Register	
	0x300c74	EPcStartAdrs_H	EPc FIFO Start Address High Register	Set FIFO start address for EPc
	0x300c75	EPcStartAdrs_L	EPc FIFO Start Address Low Register	
	0x300c76	EPdStartAdrs_H	EPd FIFO Start Address High Register	Set FIFO start address for EPd
	0x300c77	EPdStartAdrs_L	EPd FIFO Start Address Low Register	
	0x300c80	CPU_JoinRd	CPU Join FIFO Read Register	Set up FIFO data read conditions
	0x300c81	CPU_JoinWr	CPU Join FIFO Write Register	Set up FIFO data write conditions
	0x300c82	EnEPnFIFO _Access	EPn FIFO Access Enable Register	Enable CPU_JoinRd and CPU_JoinWr registers
	0x300c83	EPnFIFOforCPU	EPn FIFO for CPU Register	EPn FIFO for accessing by CPU
	0x300c84	EPnRdRemain_H	EPn FIFO Read Remain High Register	Indicate remained data quantity in FIFO
	0x300c85	EPnRdRemain_L	EPn FIFO Read Remain Low Register	
	0x300c86	EPnWrRemain_H	EPn FIFO Write High Register	Indicate free space capacity in FIFO
	0x300c87	EPnWrRemain_L	EPn FIFO Write Low Register	
	0x300c88	DescAdrs_H	Descriptor Address High Register	Specify FIFO start address for the descriptor reply function
	0x300c89	DescAdrs_L	Descriptor Address Low Register	
	0x300c8a	DescSize_H	Descriptor Size High Register	Specify number of data for the descriptor reply function
	0x300c8b	DescSize_L	Descriptor Size Low Register	
	0x300c8f	DescDoor	Descriptor Door Register	Read/write descriptors
	0x300c90	DMA_FIFO_Control	DMA FIFO Control Register	Control DMA FIFO
	0x300c91	DMA_Join	DMA Join FIFO	Enable endpoint to perform DMA transfer
	0x300c92	DMA_Control	DMA Control Register	Control DMA transfer and indicate DMA status
	0x300c94	DMA_Config_0	DMA Configuration 0 Register	Configure DMA interface signals
	0x300c95	DMA_Config_1	DMA Configuration 1 Register	Set DMA interface operating modes

Peripheral	Address	Register name		Function
USB function controller (USB) (8-bit device)	0x300c97	DMA_Latency	DMA Latency Register	Set data transfer latency
	0x300c98	DMA_Remain_H	DMA FIFO Remain High Register	Indicate remained data quantity in FIFO or free space capacity in FIFO
	0x300c99	DMA_Remain_L	DMA FIFO Remain Low Register	
	0x300c9c	DMA_Count_HH	DMA Transfer Byte Counter High/High Register	Specify/indicate data length in DMA transfer
	0x300c9d	DMA_Count_HL	DMA Transfer Byte Counter High/Low Register	
	0x300c9e	DMA_Count_LH	DMA Transfer Byte Counter Low/High Register	
	0x300c9f	DMA_Count_LL	DMA Transfer Byte Counter Low/Low Register	
Prescaler (PSC) (8-bit device)	0x300e00	PSC_CTL0	PSC Ch.0 Control Register	Control prescaler Ch.0
	0x300e01	PSC_CTL12	PSC Ch.1/2 Control Register	Control prescalers Ch.1 and 2
UART (with IrDA) (8-bit device)	0x300e10	UART_ST	UART Status Register	Indicates transfer, buffer and error statuses.
	0x300e11	UART_TXD	UART Transmit Data Register	Transmit data
	0x300e12	UART_RXD	UART Receive Data Register	Receive data
	0x300e13	UART_MOD	UART Mode Register	Sets transfer data format.
	0x300e14	UART_CTL	UART Control Register	Controls data transfer.
	0x300e15	UART_EXP	UART Expansion Register	Sets IrDA mode.
Card interface (CARD) (8-bit device)	0x300f00	CARD_EDC_TRG	CARD EDC Trigger Register	Resets the hardware accelerator registers and terminates 10-symbol EDC.
	0x300f01	CARD_EDC_ CTRL	CARD EDC Control Register	Sets the operating mode and enables operation.
	0x300f02	CARD_EDC_RGN	CARD EDC Region Register	Selects the card area.
	0x300f10	CARD_EDC_FLG	CARD EDC Flag Register	Indicates error detection status.
	0x300f20 : 0x300f38	CARD_EDC10_ COD0 : CARD_EDC10_ COD24	CARD 10-Symbol EDC Code Register 0 : CARD 10-Symbol EDC Code Register 24	10-symbol EDC RS code
	0x300f40 : 0x300f43	CARD_EDC2_ COD0 : CARD_EDC2_ COD3	CARD 2-Symbol EDC Code Register 0 : CARD 2-Symbol EDC Code Register 3	2-symbol EDC RS code
Watchdog timer (WDT) (16-bit device)	0x301000	WD_PROTECT	WDT Write Protect Register	Enable WDT register write protection
	0x301002	WD_EN	WDT Enable and Setup Register	Configure and start watchdog timer
	0x301004	WD_CMP_L	WDT Comparison Data L Register	Comparison data
	0x301006	WD_CMP_H	WDT Comparison Data H Register	
	0x301008	WD_CNT_L	WDT Count Data L Register	Watchdog timer counter data
	0x30100a	WD_CNT_H	WDT Count Data H Register	
	0x30100c	WD_CTL	WDT Control Register	Reset watchdog timer
Fine mode 8-bit timer (T8F) Ch.0 (16-bit device)	0x301100	T8F_CLK0	T8F Ch.0 Input Clock Select Register	Select prescaler output clock
	0x301102	T8F_TR0	T8F Ch.0 Reload Data Register	Set reload data
	0x301104	T8F_TC0	T8F Ch.0 Counter Data Register	Counter data
	0x301106	T8F_CTL0	T8F Ch.0 Control Register	Set timer mode and start/stop timer
	0x301108	T8F_INT0	T8F Ch.0 Interrupt Control Register	Control interrupt
Fine mode 8-bit timer (T8F) Ch.1 (16-bit device)	0x301110	T8F_CLK1	T8F Ch.1 Input Clock Select Register	Select prescaler output clock
	0x301112	T8F_TR1	T8F Ch.1 Reload Data Register	Set reload data
	0x301114	T8F_TC1	T8F Ch.1 Counter Data Register	Counter data
	0x301116	T8F_CTL1	T8F Ch.1 Control Register	Set timer mode and start/stop timer
	0x301118	T8F_INT1	T8F Ch.1 Interrupt Control Register	Control interrupt
Fine mode 8-bit timer (T8F) Ch.2 (16-bit device)	0x301120	T8F_CLK2	T8F Ch.2 Input Clock Select Register	Select prescaler output clock
	0x301122	T8F_TR2	T8F Ch.2 Reload Data Register	Set reload data
	0x301124	T8F_TC2	T8F Ch.2 Counter Data Register	Counter data
	0x301126	T8F_CTL2	T8F Ch.2 Control Register	Set timer mode and start/stop timer
	0x301128	T8F_INT2	T8F Ch.2 Interrupt Control Register	Control interrupt
Fine mode 8-bit timer (T8F) Ch.3 (16-bit device)	0x301130	T8F_CLK3	T8F Ch.3 Input Clock Select Register	Select prescaler output clock
	0x301132	T8F_TR3	T8F Ch.3 Reload Data Register	Set reload data
	0x301134	T8F_TC3	T8F Ch.3 Counter Data Register	Counter data
	0x301136	T8F_CTL3	T8F Ch.3 Control Register	Set timer mode and start/stop timer
	0x301138	T8F_INT3	T8F Ch.3 Interrupt Control Register	Control interrupt
Fine mode 8-bit timer (T8F) Ch.4 (16-bit device)	0x301140	T8F_CLK4	T8F Ch.4 Input Clock Select Register	Select prescaler output clock
	0x301142	T8F_TR4	T8F Ch.4 Reload Data Register	Set reload data
	0x301144	T8F_TC4	T8F Ch.4 Counter Data Register	Counter data
	0x301146	T8F_CTL4	T8F Ch.4 Control Register	Set timer mode and start/stop timer
	0x301148	T8F_INT4	T8F Ch.4 Interrupt Control Register	Control interrupt
Fine mode 8-bit timer (T8F) Ch.5 (16-bit device)	0x301150	T8F_CLK5	T8F Ch.5 Input Clock Select Register	Select prescaler output clock
	0x301152	T8F_TR5	T8F Ch.5 Reload Data Register	Set reload data
	0x301154	T8F_TC5	T8F Ch.5 Counter Data Register	Counter data
	0x301156	T8F_CTL5	T8F Ch.5 Control Register	Set timer mode and start/stop timer
	0x301158	T8F_INT5	T8F Ch.5 Interrupt Control Register	Control interrupt

Peripheral	Address	Register name		Function
16-bit PWM timer (T16A6) Ch.0 (16-bit device)	0x301160	T16A_CTL0	T16A6 Ch.0 Counter Control Register	Control counter
	0x301162	T16A_TC0	T16A6 Ch.0 Counter Data Register	Counter data
	0x301164	T16A_CCCTL0	T16A6 Ch.0 Comparator/Capture Control Register	Control comparator/capture block and TOUT
	0x301166	T16A_CCA0	T16A6 Ch.0 Comparator/Capture A Data Register	Compare A/capture A data
	0x301168	T16A_CCB0	T16A6 Ch.0 Comparator/Capture B Data Register	Compare B/capture B data
	0x30116a	T16A_IEN0	T16A6 Ch.0 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
	0x30116c	T16A_IFLG0	T16A6 Ch.0 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
16-bit PWM timer (T16A6) Ch.1 (16-bit device)	0x301170	T16A_CTL1	T16A6 Ch.1 Counter Control Register	Control counter
	0x301172	T16A_TC1	T16A6 Ch.1 Counter Data Register	Counter data
	0x301174	T16A_CCCTL1	T16A6 Ch.1 Comparator/Capture Control Register	Control comparator/capture block and TOUT
	0x301176	T16A_CCA1	T16A6 Ch.1 Comparator/Capture A Data Register	Compare A/capture A data
	0x301178	T16A_CCB1	T16A6 Ch.1 Comparator/Capture B Data Register	Compare B/capture B data
	0x30117a	T16A_IEN1	T16A6 Ch.1 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
	0x30117c	T16A_IFLG1	T16A6 Ch.1 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
16-bit PWM timer (T16A6) Ch.2 (16-bit device)	0x301180	T16A_CTL2	T16A6 Ch.2 Counter Control Register	Control counter
	0x301182	T16A_TC2	T16A6 Ch.2 Counter Data Register	Counter data
	0x301184	T16A_CCCTL2	T16A6 Ch.2 Comparator/Capture Control Register	Control comparator/capture block and TOUT
	0x301186	T16A_CCA2	T16A6 Ch.2 Comparator/Capture A Data Register	Compare A/capture A data
	0x301188	T16A_CCB2	T16A6 Ch.2 Comparator/Capture B Data Register	Compare B/capture B data
	0x30118a	T16A_IEN2	T16A6 Ch.2 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
	0x30118c	T16A_IFLG2	T16A6 Ch.2 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
16-bit PWM timer (T16A6) Ch.3 (16-bit device)	0x301190	T16A_CTL3	T16A6 Ch.3 Counter Control Register	Control counter
	0x301192	T16A_TC3	T16A6 Ch.3 Counter Data Register	Counter data
	0x301194	T16A_CCCTL3	T16A6 Ch.3 Comparator/Capture Control Register	Control comparator/capture block and TOUT
	0x301196	T16A_CCA3	T16A6 Ch.3 Comparator/Capture A Data Register	Compare A/capture A data
	0x301198	T16A_CCB3	T16A6 Ch.3 Comparator/Capture B Data Register	Compare B/capture B data
	0x30119a	T16A_IEN3	T16A6 Ch.3 Comparator/Capture Interrupt Enable Register	Enable/disable T16A6 interrupts
	0x30119c	T16A_IFLG3	T16A6 Ch.3 Comparator/Capture Interrupt Flag Register	Indicate T16A6 interrupt cause status
16-bit audio PWM timer (T16P) Ch.0 (16-bit device)	0x301200	T16P_A0	T16P Ch.0 Compare A Buffer Register	Compare A data
	0x301202	T16P_B0	T16P Ch.0 Compare B Buffer Register	Compare B data
	0x301204	T16P_CNT_DATA0	T16P Ch.0 Counter Data Register	Counter data
	0x301206	T16P_VOL_CTL0	T16P Ch.0 Volume Control Register	Enable volume control and set volume level
	0x301208	T16P_CTL0	T16P Ch.0 Control Register	Set timer operating conditions
	0x30120a	T16P_RUN0	T16P Ch.0 Running Control Register	Start/stop timer
	0x30120c	T16P_CLK0	T16P Ch.0 Internal Clock Control Register	Select internal count clock
	0x30120e	T16P_INT0	T16P Ch.0 Interrupt Control Register	Control T16P interrupts
16-bit audio PWM timer (T16P) Ch.1 (16-bit device)	0x301210	T16P_A1	T16P Ch.1 Compare A Buffer Register	Compare A data
	0x301212	T16P_B1	T16P Ch.1 Compare B Buffer Register	Compare B data
	0x301214	T16P_CNT_DATA1	T16P Ch.1 Counter Data Register	Counter data
	0x301216	T16P_VOL_CTL1	T16P Ch.1 Volume Control Register	Enable volume control and set volume level
	0x301218	T16P_CTL1	T16P Ch.1 Control Register	Set timer operating conditions
	0x30121a	T16P_RUN1	T16P Ch.1 Running Control Register	Start/stop timer
	0x30121c	T16P_CLK1	T16P Ch.1 Internal Clock Control Register	Select internal count clock
	0x30121e	T16P_INT1	T16P Ch.1 Interrupt Control Register	Control T16P interrupts
A/D converter (ADC10) (16-bit device)	0x301300	ADC10_ADD	A/D Conversion Result Register	A/D converted data
	0x301302	ADC10_TRG	A/D Trigger/Channel Select Register	Set start/end channels and conversion mode
	0x301304	ADC10_CTL	A/D Control/Status Register	Control A/D converter and indicate conversion status
	0x301306	ADC10_CLK	A/D Clock Control Register	Control A/D converter clock
Remote controller (REMC) (16-bit device)	0x301500	REMC_CFG	REMC Configuration Register	Control clock and data transfer
	0x301502	REMC_CAR	REMC Carrier Length Setup Register	Set carrier H/L section lengths
	0x301504	REMC_LCNT	REMC Length Counter Register	Set transmit/receive data length
	0x301506	REMC_INT	REMC Interrupt Control Register	Control REMC interrupts

Peripheral	Address	Register name		Function
LCD controller (LCDC) (32-bit device)	0x302000	LCDC_INT	LCDC Interrupt Enable Register	Enable/disable LCDC interrupts
	0x302004	LCDC_PSAVE	Status and Power Save Configuration Register	Indicate LCDC status and set power save mode
	0x302010	LCDC_HDISP	Horizontal Display Register	Set horizontal display period
	0x302014	LCDC_VDISP	Vertical Display Register	Set vertical display period
	0x302018	LCDC_MODR	MOD Rate Register	Set MOD rate
	0x302020	LCDC_HDPS	Horizontal Display Start Position Register	Set horizontal display start position for TFT
	0x302024	LCDC_VDPS	Vertical Display Start Position Register	Set vertical display start position for TFT
	0x302028	LCDC_FPLINE	FPLINE Pulse Setting Register	Configure FPLINE pulse for TFT
	0x30202c	LCDC_FPFR	FPFRAME Pulse Setting Register	Configure FPFRAME pulse for TFT
	0x302030	LCDC_FPFROFS	FPFRAME Pulse Offset Register	Adjust FPLINE pulse timings for TFT
	0x302040	LCDC_TFTSO	TFT Special Output Register	Set TFT control signals
	0x302044	LCDC_TFT_CTL1	TFT_CTL1 Pulse Register	Set TFT_CTL1 pulse timings
	0x302048	LCDC_TFT_CTL0	TFT_CTL0 Pulse Register	Set TFT_CTL0 pulse timings
	0x30204c	LCDC_TFT_CTL2	TFT_CTL2 Register	Set TFT_CTL2 signal timing
	0x302050	LCDC_RLDCTL	LCDC Reload Control Register	Control reloading
	0x302054	LCDC_RLDADR	LCDC Reload Table Base Address Register	Set reload table base address
	0x302060	LCDC_DISPMOD	LCDC Display Mode Register	Set display conditions
	0x302070	LCDC_MAINADR	Main Window Display Start Address Register	Set main window display start address
	0x302074	LCDC_MAINOFS	Main Screen Address Offset Register	Set virtual main screen width
	0x302080	LCDC_SUBADR	Sub-window Display Start Address Register	Set sub-window display start address
	0x302084	LCDC_SUBOFS	Sub-screen Address Offset Register	Set virtual sub-screen width
	0x302088	LCDC_SUBSP	Sub-window Start Position Register	Set sub-window start position
	0x30208c	LCDC_SUBEP	Sub-window End Position Register	Set sub-window end position
	0x302090	LCDC_MLUT0	Monochrome Look-up Table Register 0	Monochrome look-up table data entries 0–7
	0x302094	LCDC_MLUT1	Monochrome Look-up Table Register 1	Monochrome look-up table data entries 8–15
DMA controller (DMAC) (32-bit device)	0x302100	DMAC_CH_EN	DMAC Channel Enable Register	Enable DMAC channels
	0x302104	DMAC_TBL_BASE	DMAC Control Table Base Address Register	Set control table base address
	0x302108	DMAC_IE	DMAC Interrupt Enable Register	Enable/disable DMAC interrupts
	0x30210c	DMAC_TRG_SEL	DMAC Trigger Select Register	Select trigger sources
	0x302110	DMAC_TRG_FLG	DMAC Trigger Flag Register	Control software trigger and indicate trigger status
	0x302114	DMAC_END_FLG	DMAC End-of-Transfer Flag Register	Indicate DMA completed channels
	0x302118	DMAC_RUN_ STAT	DMAC Running Status Register	Indicates running channel
	0x30211c	DMAC_PAUSE_ STAT	DMAC Pause Status Register	Indicate DMA suspended channels
SDRAM controller (SDRAMC) (32-bit device)	0x302200	SDRAMC_INIT	SDRAM Initialization Register	Enable SDRAMC and control SDRAM initialization
	0x302204	SDRAMC_CFG	SDRAM Configuration Register	Set SDRAM size and timing parameters
	0x302208	SDRAMC_REF	SDRAM Refresh Control Register	Control SDRAM refresh
	0x302210	SDRAMC_APP	SDRAM Application Configuration Register	Set CAS latency
SRAM controller (SRAMC) (32-bit device)	0x302220	SRAMC_TMG47	#CE[7:4] Access Timing Configuration Register	Set #CE[7:4] access conditions
	0x302224	SRAMC_TMG810	#CE[10:8] Access Timing Configuration Register	Set #CE[10:8] access conditions
	0x302228	SRAMC_TYPE	#CE[10:4] Device Configuration Register	Set #CE[10:4] device types
Cache controller (CCU) (32-bit device)	0x302300	CCU_CFG	Cache Configuration Register	Enable instruction and data caches
	0x302304	CCU_AREA	Cacheable Area Select Register	Select cacheable areas
	0x302308	CCU_LK	Cache Lock Register	Configure cache lock function
	0x30230c	CCU_STAT	Cache Status Register	Indicate cache statuses
	0x302318	CCU_WB_STAT	Cache Write Buffer Status Register	Indicate write buffer status
I^2S (32-bit device)	0x302400	I2S_CTL0	I^2S Ch.0 Control Register	Control I^2S Ch.0 output
	0x302404	I2S_CTL1	I^2S Ch.1 Control Register	Control I^2S Ch.1 input
	0x302408	I2S_DV_MCLK	I^2S Master Clock Division ratio Register	Configure master clock
	0x30240c	I2S_DV_BCLK	I^2S Audio Clock Division ratio Register	Configure audio clock
	0x302410	I2S_START	I^2S Start/Stop Register	Control/indicate I^2S start/stop status
	0x302414	I2S_STATUS	I^2S FIFO Status Register	Indicate FIFO status
	0x302418	I2S_INT	I^2S Interrupt Control Register	Control I^2S interrupts
	0x30241c	I2S_MODE	I^2S Mode Select Register	Set DMA and pin modes
	0x302420	I2S_FIFO0	I^2S Ch.0 FIFO Register	I^2S Ch.0 output data
	0x302422			
	0x302430	I2S_FIFO1	I^2S Ch.1 FIFO Register	I^2S Ch.1 input data
	0x302432			

Peripheral	Address	Register name		Function
SD/MMC interface (SD_MMC) (32-bit device)	0x302600	MMC_CLKCTL	SD_MMC Clock Control Register	Control MMC clock
	0x302604	MMC_FUNCTL	SD_MMC Function Control Register	Control data transfer
	0x302608	MMC_BLK	SD_MMC Block Size Setting Register	Set block size
	0x30260c	MMC_TIMEOUT	SD_MMC Data/Busy Timeout Period Setting Register	Set data/busy timeout
	0x302610	MMC_CMD_IDX	SD_MMC Command Index Register	Set command index
	0x302614	MMC_CMD_ARG	SD_MMC Command Argument Register	Set command argument
	0x302618	MMC_RSP0	SD_MMC Response Register 0	Response code[31:0]
	0x30261c	MMC_RSP1	SD_MMC Response Register 1	Response code[63:32]
	0x302620	MMC_RSP2	SD_MMC Response Register 2	Response code[95:64]
	0x302624	MMC_RSP3	SD_MMC Response Register 3	Response code[127:96]
	0x302628	MMC_RSP4	SD_MMC Response Register 4	Response code[135:128]
	0x30262c	MMC_STATUS	SD_MMC Status Register	Indicate status
	0x302630	MMC_INT	SD_MMC Interrupt Control Register	Control interrupts and DMA transfer
	0x302650	MMC_DAT	SD_MMC Data Transfer FIFO Register	Transmit/receive data

Note: Do not access the unused peripheral circuit areas not listed in the table from the application program.

0x300010–0x300024 Misc Registers (MISC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Wait Control Register (MISC_RTCWT)	0x300010 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	RTCWT[2:0]	RTC register access wait control	0 to 7 cycles	0x7	R/W	Write-protected
USB Configuration Register (MISC_USB)	0x300012 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	USBINTEN	USB interrupt enable	1 Enable 0 Disable	0	R/W	Write-protected
		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	0 to 7 cycles	0x7	R/W	Write-protected
Internal RAM Wait Control Register (MISC_RAMWT)	0x300014 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	CALRWT	CALCRAM access wait control	1 1 wait cycle 0 0 wait cycles	1	R/W	Write-protected Should be set to 0.
		D2	–	reserved	–	–	–	Should be fixed at 0.
		D1	A3WT	Area 3 RAM access wait control	1 1 wait cycle 0 0 wait cycles	1	R/W	Write-protected
		D0	A0WT	Area 0 RAM access wait control	1 1 wait cycle 0 0 wait cycles	1	R/W	Should be set to 0.
Boot Register (MISC_BOOT)	0x300016 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5–4	BOOT[1:0]	Boot mode indicator	BOOT[1:0] Boot mode 0x3 SPI/RS232C 0x2 NOR/ROM 0x1 HIF 0x0 NAND	*	R	* Depends on the BOOT pin status at initial reset
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	BOOTEN	#CE10 area boot enable	1 Internal 0 External	1	R/W	Write-protected
		D0	–	reserved	–	–	–	0 when being read.
Debug Trace Enable Register (MISC_TRACE)	0x300018 (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	TRACE	PC trace debugging enable	1 Disable 0 Enable	0	R/W	Write-protected
Misc Protect Register 0 (MISC_ PROT0)	0x300020 (8 bits)	D7–0	PROT0[7:0]	Misc register write-protect flag	Writing 10010110 (0x96) removes the write protection of the Misc registers (0x300010–0x300018). Writing another value set the write protection.	0x0	R/W	
Misc Protect Register 1 (MISC_ PROT1)	0x300022 (8 bits)	D7–0	PROT1[7:0]	Misc register write-protect flag	Writing 10010110 (0x96) removes the write protection of the Misc register (0x300024). Writing another value set the write protection.	0x0	R/W	
RAM Configuration Register (MISC_ RAMCFG)	0x300024 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	RAMCFG [2:0]	DSTRAM/IVRAM configuration	RAMCFG[2:0] DSTRAM 0x7–0x4 DSTRAM 0x3 CALCRAM 0x2 DSTRAM 0x1 CALCRAM 0x0 DSTRAM RAMCFG[2:0] IVRAM 0x7–0x4 CALCRAM 0x3 IRAM 0x2 IRAM 0x1 VRAM 0x0 VRAM	0x0	R/W	Write-protected

0x300100–0x300118 Clock Management Unit (CMU)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Clock Source Select Register (CMU_OSCSEL)	0x300100 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1–0	CLKSEL [1:0]	System clock source select	CLKSEL[1:0] Clock source 0x3 Not allowed 0x2 PLL 0x1 OSC1 0x0 OSC3	0x0	R/W	Write-protected

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Oscillation Control Register (CMU_OSCCTL)	**0x300101** (8 bits)	D7–4	**OSC3WT[3:0]**	OSC3 wait cycle select	OSC3WT[3:0] / Wait cycle: 0xf 128 cycles 0xe 256 cycles 0xd 512 cycles 0xc 1,024 cycles 0xb 2,048 cycles 0xa 4,096 cycles 0x9 8,192 cycles 0x8 16,384 cycles 0x7 32,768 cycles 0x6 65,536 cycles 0x5 131,072 cycles 0x4 262,144 cycles 0x3 524,288 cycles 0x2 1,048,576 cycles 0x1 2,097,152 cycles 0x0 4,194,304 cycles	0xf	R/W	Write-protected
		D3–2	–	reserved	–	–	–	0 when being read.
		D1	**OSC1EN**	OSC1 enable	1 Enable 0 Disable	1	R/W	Write-protected
		D0	**OSC3EN**	OSC3 enable	1 Enable 0 Disable	1	R/W	
LCDC Clock Division Ratio Select Register (CMU_ LCLKDIV)	**0x300103** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4–0	**LCLKDIV[4:0]**	LCDC clock division ratio select	LCLKDIV[4:0] / Division ratio: 0x1f 1/32 0x1e 1/31 0x1d 1/30 0x1c 1/29 0x1b 1/28 0x1a 1/27 0x19 1/26 0x18 1/25 0x17 1/24 0x16 1/23 0x15 1/22 0x14 1/21 0x13 1/20 0x12 1/19 0x11 1/18 0x10 1/17 0xf 1/16 0xe 1/15 0xd 1/14 0xc 1/13 0xb 1/12 0xa 1/11 0x9 1/10 0x8 1/9 0x7 1/8 0x6 1/7 0x5 1/6 0x4 1/5 0x3 1/4 0x2 1/3 0x1 1/2 0x0 1/1	0x7	R/W	Clock source = OSC3 Write-protected
System Clock Division Ratio Select Register (CMU_ SYSCLKDIV)	**0x300105** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**SYSCLKDIV [2:0]**	System clock division ratio select	SYSCLKDIV[2:0] / Division ratio: 0x7–0x6 1/1 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Clock source = OSC (OSC3, PLL, or OSC1) Write-protected
CMU_CLK Select Register (CMU_CMUCLK)	**0x300106** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4–0	**CMU_ CLKSEL[4:0]**	CMU_CLK select	CMU_CLKSEL[4:0] / CMU_CLK: 0xf–0xb reserved 0xa OSC/32 0x9 OSC/16 0x8 OSC/8 0x7 OSC/4 0x6 OSC/2 0x5 OSC/1 0x4 LCDC_CLK 0x3 BCLK 0x2 PLL 0x1 OSC1 0x0 OSC3	0x0	R/W	OSC: system clock (OSC3, PLL, OSC1) Write-protected

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PLL Input Clock Division Ratio Select Register (CMU_ PLLINDIV)	**0x300107** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**PLLINDIV [3:0]**	PLL input clock division ratio select	PLLINDIV[3:0] / Division ratio 0xf–0xa / 1/8 0x9 / 1/10 0x8 / 1/9 0x7 / 1/8 0x6 / 1/7 0x5 / 1/6 0x4 / 1/5 0x3 / 1/4 0x2 / 1/3 0x1 / 1/2 0x0 / 1/1	0x7	R/W	Clock source = OSC3 Write-protected
PLL Control Register 0 (CMU_ PLLCTL0)	**0x300108** (8 bits)	D7–4	**PLLN[3:0]**	PLL multiplication rate setup	PLLN[3:0] / Multiplication rate 0xf / x16 0xe / x15 0xd / x14 0xc / x13 0xb / x12 0xa / x11 0x9 / x10 0x8 / x9 0x7 / x8 0x6 / x7 0x5 / x6 0x4 / x5 0x3 / x4 0x2 / x3 0x1 / x2 0x0 / x1	0x0	R/W	Write-protected
		D3–2	**PLLV[1:0]**	PLL V-divider setup	PLLV[1:0] / W 0x3 / 8 0x2 / 4 0x1 / 2 0x0 / Not allowed	0x1	R/W	
		D1	**–**	reserved	–	–	–	0 when being read.
		D0	**PLLPOWR**	PLL enable	1 Enable 0 Disable	0	R/W	Write-protected
PLL Control Register 1 (CMU_ PLLCTL1)	**0x300109** (8 bits)	D7–4	**PLLVC[3:0]**	PLL VCO Kv setup	PLLVC[3:0] / fVCO [MHz] 0x8 / 360 < fVCO ≤ 400 0x7 / 320 < fVCO ≤ 360 0x6 / 280 < fVCO ≤ 320 0x5 / 240 < fVCO ≤ 280 0x4 / 200 < fVCO ≤ 240 0x3 / 160 < fVCO ≤ 200 0x2 / 120 < fVCO ≤ 160 0x1 / 100 ≤ fVCO ≤ 120 Other / Not allowed	0x1	R/W	Write-protected
		D3–0	**PLLRS[3:0]**	PLL LPF resistance setup	PLLRS[3:0] / fREFCK [MHz] 0xa / 5 ≤ fREFCK < 20 0x8 / 20 ≤ fREFCK ≤ 150 Other / Not allowed	0x8	R/W	
PLL Control Register 2 (CMU_ PLLCTL2)	**0x30010a** (8 bits)	D7–6	**PLLCS[1:0]**	PLL LPF capacitance	0x0	0x0	R	
		D5	**PLLBYP**	PLL bypass mode	0	0	R	
		D4–0	**PLLCP[4:0]**	PLL charge pump current	0x10	0x10	R	
SSCG Macro Control Register 0 (CMU_SSCG0)	**0x30010c** (8 bits)	D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**SSMCON**	SSCG enable	1 Enable 0 Disable	0	R/W	Write-protected
SSCG Macro Control Register 1 (CMU_SSCG1)	**0x30010d** (8 bits)	D7–4	**SSMCITM [3:0]**	SSCG interval timer (ITM) setting	0x0 to 0xf	0xf	R/W	Write-protected
		D3–0	**SSMCIDT [3:0]**	SSCG maximum frequency change width setting	0x0 to 0xf	0x0	R/W	
CMU Write Protect Register (CMU_ PROTECT)	**0x300110** (8 bits)	D7–0	**CMUP[7:0]**	CMU register write-protect flag	Writing 10010110 (0x96) removes the write protection of the CMU registers (0x300100–0x30010d, 0x300114–0x300118). Writing another value set the write protection.	0x0	R/W	
Clock Control Register 0 (CMU_ CLKCTL0)	**0x300114** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**T16A_CKE**	T16A6 register clock enable	1 Enable 0 Disable	1	R/W	Write-protected
		D2	**TCLK_CKE**	T16A6 counter clock enable	1 Enable 0 Disable	1	R/W	
		D1–0	**TCLK_SEL [1:0]**	T16A6 counter clock source select	TCLK_SEL[1:0] / Clock source 0x3–0x2 / OSC1 0x1 / OSC3 0x0 / SYSCLK	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
Clock Control Register 1 (CMU_ CLKCTL1)	**0x300115** (8 bits)	D7	**NANDIF_ CKE**	CARD clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D6	**REMC_CKE**	REMC clock enable	1	Enable	0	Disable	1	R/W	
		D5	**ADC_CKE**	ADC10 clock enable	1	Enable	0	Disable	1	R/W	
		D4	**PORT_CKE**	GPIO/PMUX clock enable	1	Enable	0	Disable	1	R/W	
		D3	**SDMMC_ CKE**	SD_MMC clock enable	1	Enable	0	Disable	1	R/W	
		D2	**I2S_CKE**	I2S clock enable	1	Enable	0	Disable	1	R/W	
		D1	**ITC_CKE**	ITC clock enable	1	Enable	0	Disable	1	R/W	
		D0	**DMAC_CKE**	DMAC clock enable	1	Enable	0	Disable	1	R/W	
Clock Control Register 2 (CMU_ CLKCTL2)	**0x300116** (8 bits)	D7	**HIF_CKE**	HIF clock enable	1	Enable	0	Disable	0	R/W	Write-protected
		D6	**LCDC_ SAPB_CKE**	LCDC SAPB I/F clock enable	1	Enable	0	Disable	0	R/W	
		D5	**LCDC_ AHB_CKE**	LCDC AHB I/F clock enable	1	Enable	0	Disable	0	R/W	
		D4	**LCDC_CKE**	LCDC clock enable	1	Enable	0	Disable	0	R/W	
		D3	**USBREG_ CKE**	USB register clock enable	1	Enable	0	Disable	0	R/W	
		D2	**USB_CKE**	USB clock enable	1	Enable	0	Disable	0	R/W	
		D1	**BCU_CKE**	SRAMC/SDRAMC clock enable	1	Enable	0	Disable	1	R/W	
		D0	**BCLK_CKE**	BCLK clock enable (in HALT mode)	1	Enable	0	Disable	1	R/W	
Clock Control Register 3 (CMU_ CLKCTL3)	**0x300117** (8 bits)	D7	**PSC1/2_ CKE**	PSC1/2 clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D6	**T8F_CKE**	T8F clock enable	1	Enable	0	Disable	1	R/W	
		D5	**UART_CKE**	UART clock enable	1	Enable	0	Disable	1	R/W	
		D4	**T16P_CKE**	T16P clock enable	1	Enable	0	Disable	1	R/W	
		D3–2	**–**	reserved	–				–	–	0 when being read.
		D1	**USI_CKE**	USI clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D0	**USIL_CKE**	USIL clock enable	1	Enable	0	Disable	1	R/W	
Clock Control Register 4 (CMU_ CLKCTL4)	**0x300118** (8 bits)	D7–5	**–**	reserved	–				–	–	0 when being read.
		D4	**MISC_CKE**	MISC clock enable	1	Enable	0	Disable	1	R/W	Write-protected
		D3	**IVRAM_ ARB_CKE**	IVRAM arbiter clock enable	1	Enable	0	Disable	1	R/W	
		D2	**A3RAM_ CKE**	Area 3 RAM clock enable	1	Enable	0	Disable	1	R/W	
		D1	**RTC_ SAPB_CKE**	RTC SAPB I/F clock enable	1	Enable	0	Disable	1	R/W	
		D0	**WDT_CKE**	WDT clock enable	1	Enable	0	Disable	1	R/W	

0x300210–0x30022c Interrupt Controller (ITC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Port Input 0 Interrupt Level Register (ITC_PINT0_LV)	**0x300210** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	Port input 0 interrupt level	1 to 7	0x0	R/W	
Port Input 1 Interrupt Level Register (ITC_PINT1_LV)	**0x300211** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	Port input 1 interrupt level	1 to 7	0x0	R/W	
Key input 0 Interrupt Level Register (ITC_KINT0_LV)	**0x300212** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	Key input 0 interrupt level	1 to 7	0x0	R/W	
Key Input 1 Interrupt Level Register (ITC_KINT1_LV)	**0x300213** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	Key input 1 interrupt level	1 to 7	0x0	R/W	
DMAC Ch.0/2 Interrupt Level Register (ITC_DMA02_LV)	**0x300214** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	DMAC Ch.0 and 2 interrupt level	1 to 7	0x0	R/W	
DMAC Ch.1/3 Interrupt Level Register (ITC_DMA13_LV)	**0x300215** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	DMAC Ch.1 and 3 interrupt level	1 to 7	0x0	R/W	
DMAC Ch.4/6 Interrupt Level Register (ITC_DMA46_LV)	**0x300216** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	DMAC Ch.4 and 6 interrupt level	1 to 7	0x0	R/W	
DMAC Ch.5/7 Interrupt Level Register (ITC_DMA57_LV)	**0x300217** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	DMAC Ch.5 and 7 interrupt level	1 to 7	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.0 Interrupt Level Register (ITC_T16P0_LV)	**0x300218** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T16P Ch.0 interrupt level	1 to 7	0x0	R/W	
T16P Ch.1 Interrupt Level Register (ITC_T16P1_LV)	**0x300219** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T16P Ch.1 interrupt level	1 to 7	0x0	R/W	
T16A6 Ch.0 Interrupt Level Register (ITC_T16A0_LV)	**0x30021a** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T16A6 Ch.0 interrupt level	1 to 7	0x0	R/W	
T16A6 Ch.1 Interrupt Level Register (ITC_T16A1_LV)	**0x30021b** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T16A6 Ch.1 interrupt level	1 to 7	0x0	R/W	
T16A6 Ch.2 Interrupt Level Register (ITC_T16A2_LV)	**0x30021c** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T16A6 Ch.2 interrupt level	1 to 7	0x0	R/W	
T16A6 Ch.3 Interrupt Level Register (ITC_T16A3_LV)	**0x30021d** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T16A6 Ch.3 interrupt level	1 to 7	0x0	R/W	
T8F Ch.0/2/4 Interrupt Level Register (ITC_T8F024_LV)	**0x30021e** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T8F Ch.0, 2, and 4 interrupt level	1 to 7	0x0	R/W	
T8F Ch.1/3/5 Interrupt Level Register (ITC_T8F135_LV)	**0x30021f** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	T8F Ch.1, 3, and 5 interrupt level	1 to 7	0x0	R/W	
ADC10 Interrupt Level Register (ITC_ADC10_LV)	**0x300220** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	ADC10 interrupt level	1 to 7	0x0	R/W	
RTC Interrupt Level Register (ITC_RTC_LV)	**0x300221** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	RTC interrupt level	1 to 7	0x0	R/W	
USI Ch.0/UART Interrupt Level Register (ITC_USI0_LV)	**0x300222** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	USI Ch.0/UART interrupt level	1 to 7	0x0	R/W	
USI Ch.1 Interrupt Level Register (ITC_USI1_LV)	**0x300223** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	USI Ch.1 interrupt level	1 to 7	0x0	R/W	
USI Ch.2 Interrupt Level Register (ITC_USI2_LV)	**0x300224** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	USI Ch.2 interrupt level	1 to 7	0x0	R/W	
USIL Interrupt Level Register (ITC_USIL_LV)	**0x300225** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	USIL interrupt level	1 to 7	0x0	R/W	
LCDC Interrupt Level Register (ITC_LCDC_LV)	**0x300226** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	LCDC interrupt level	1 to 7	0x0	R/W	
SD_MMC Interrupt Level Register (ITC_SDMMC_LV)	**0x300227** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	SD_MMC interrupt level	1 to 7	0x0	R/W	
REMC Interrupt Level Register (ITC_REMC_LV)	**0x300228** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	REMC interrupt level	1 to 7	0x0	R/W	
I²S Output Interrupt Level Register (ITC_I2SOUT_LV)	**0x300229** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	I²S output channel interrupt level	1 to 7	0x0	R/W	
I²S Input Interrupt Level Register (ITC_I2SIN_LV)	**0x30022a** (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	**INT_LV[2:0]**	I²S input channel interrupt level	1 to 7	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Interrupt Level Register (ITC_HIF_LV)	0x30022b (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	INT_LV[2:0]	HIF interrupt level	1 to 7	0x0	R/W	
USB Interrupt Level Register (ITC_USB_LV)	0x30022c (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	INT_LV[2:0]	USB interrupt level	1 to 7	0x0	R/W	

0x300300–0x30083f GPIO & Port MUX

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
P0 Port Data Register (GPIO_P0_DAT)	0x300300 (8 bits)	D7–0	P0[7:0]D	P0[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P0 Port I/O Control Register (GPIO_P0_IOC)	0x300301 (8 bits)	D7–0	IOC0[7:0]	P0[7:0] I/O control	1 Output 0 Input	0x0	R/W	
P1 Port Data Register (GPIO_P1_DAT)	0x300302 (8 bits)	D7–0	P1[7:0]D	P1[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P1 Port I/O Control Register (GPIO_P1_IOC)	0x300303 (8 bits)	D7–0	IOC1[7:0]	P1[7:0] I/O control	1 Output 0 Input	0x0	R/W	
P2 Port Data Register (GPIO_P2_DAT)	0x300304 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1–0	P2[1:0]D	P2[1:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P2 Port I/O Control Register (GPIO_P2_IOC)	0x300305 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1–0	IOC2[1:0]	P2[1:0] I/O control	1 Output 0 Input	0x0	R/W	
P3 Port Data Register (GPIO_P3_DAT)	0x300306 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–0	P3[6:0]D	P3[6:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P3 Port I/O Control Register (GPIO_P3_IOC)	0x300307 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–0	IOC3[6:0]	P3[6:0] I/O control	1 Output 0 Input	0x0	R/W	
P4 Port Data Register (GPIO_P4_DAT)	0x300308 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	P4[2:0]D	P4[2:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P4 Port I/O Control Register (GPIO_P4_IOC)	0x300309 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	IOC4[2:0]	P4[2:0] I/O control	1 Output 0 Input	0x0	R/W	
P5 Port Data Register (GPIO_P5_DAT)	0x30030a (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–0	P5[6:0]D	P5[6:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P5 Port I/O Control Register (GPIO_P5_IOC)	0x30030b (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–0	IOC5[6:0]	P5[6:0] I/O control	1 Output 0 Input	0x0	R/W	
P6 Port Data Register (GPIO_P6_DAT)	0x30030c (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	P60D	P60 I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P6 Port I/O Control Register (GPIO_P6_IOC)	0x30030d (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	IOC60	P60 I/O control	1 Output 0 Input	0	R/W	
P7 Port Data Register (GPIO_P7_DAT)	0x30030e (8 bits)	D7–0	P7[7:0]D	P7[7:0] input port data	1 1 (High) 0 0 (Low)	Ext.	R	Ext.: Depends on the external pin status.
P8 Port Data Register (GPIO_P8_DAT)	0x300310 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3–0	P8[3:0]D	P8[3:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P8 Port I/O Control Register (GPIO_P8_IOC)	0x300311 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3–0	IOC8[3:0]	P8[3:0] I/O control	1 Output 0 Input	0x0	R/W	
P9 Port Data Register (GPIO_P9_DAT)	0x300312 (8 bits)	D7–0	P9[7:0]D	P9[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
P9 Port I/O Control Register (GPIO_P9_IOC)	0x300313 (8 bits)	D7–0	IOC9[7:0]	P9[7:0] I/O control	1 Output 0 Input	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PA Port Data Register (GPIO_PA_DAT)	**0x300314** (8 bits)	D7–0	**PA[7:0]D**	PA[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
PA Port I/O Control Register (GPIO_PA_IOC)	**0x300315** (8 bits)	D7–0	**IOCA[7:0]**	PA[7:0] I/O control	1 Output 0 Input	0x0	R/W	
PB Port Data Register (GPIO_PB_DAT)	**0x300316** (8 bits)	D7–0	**PB[7:0]D**	PB[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
PB Port I/O Control Register (GPIO_PB_IOC)	**0x300317** (8 bits)	D7–0	**IOCB[7:0]**	PB[7:0] I/O control	1 Output 0 Input	0x0	R/W	
PC Port Data Register (GPIO_PC_DAT)	**0x300318** (8 bits)	D7–0	**PC[7:0]D**	PC[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
PC Port I/O Control Register (GPIO_PC_IOC)	**0x300319** (8 bits)	D7–0	**IOCC[7:0]**	PC[7:0] I/O control	1 Output 0 Input	0x0	R/W	
PD Port Data Register (GPIO_PD_DAT)	**0x30031a** (8 bits)	D7–0	**PD[7:0]D**	PD[7:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
PD Port I/O Control Register (GPIO_PD_IOC)	**0x30031b** (8 bits)	D7–0	**IOCD[7:0]**	PD[7:0] I/O control	1 Output 0 Input	0x0	R/W	
PE Port Data Register (GPIO_PE_DAT)	**0x30031c** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**PE[5:0]D**	PE[5:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
PE Port I/O Control Register (GPIO_PE_IOC)	**0x30031d** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**IOCE[5:0]**	PE[5:0] I/O control	1 Output 0 Input	0x0	R/W	
PF Port Data Register (GPIO_PF_DAT)	**0x30031e** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4–0	**PF[4:0]D**	PF[4:0] I/O port data	1 1 (High) 0 0 (Low)	Ext.	R/W	Ext.: Depends on the external pin status.
PF Port I/O Control Register (GPIO_PF_IOC)	**0x30031f** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4–0	**IOCF[4:0]**	PF[4:0] I/O control	1 Output 0 Input	0x0	R/W	
FPT0–1 Interrupt Port Select Register (GPIO_FPT01_SEL)	**0x300340** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SPT1[2:0]**	FPT1 interrupt input port select	SPT1[2:0] Port 0x7 PE1 0x6 PD1 0x5 PB1 0x4 PA1 0x3 P91 0x2 P73 0x1 P13 0x0 P01	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SPT0[2:0]**	FPT0 interrupt input port select	SPT0[2:0] Port 0x7 PE0 0x6 PD0 0x5 PB0 0x4 PA0 0x3 P90 0x2 P71 0x1 P12 0x0 P00	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT2–3 Interrupt Port Select Register (GPIO_FPT23_SEL)	**0x300341** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SPT3[2:0]**	FPT3 interrupt input port select	SPT3[2:0] Port: 0x7 PE3; 0x6 PD3; 0x5 PB3; 0x4 PA3; 0x3 P93; 0x2 P77; 0x1 P30; 0x0 P03	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SPT2[2:0]**	FPT2 interrupt input port select	SPT2[2:0] Port: 0x7 PE2; 0x6 PD2; 0x5 PB2; 0x4 PA2; 0x3 P92; 0x2 P75; 0x1 P14; 0x0 P02	0x0	R/W	
FPT4–5 Interrupt Port Select Register (GPIO_FPT45_SEL)	**0x300342** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SPT5[2:0]**	FPT5 interrupt input port select	SPT5[2:0] Port: 0x7 P55; 0x6 PD5; 0x5 PB5; 0x4 PA5; 0x3 P95; 0x2 P81; 0x1 P32; 0x0 P05	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SPT4[2:0]**	FPT4 interrupt input port select	SPT4[2:0] Port: 0x7 PE4; 0x6 PD4; 0x5 PB4; 0x4 PA4; 0x3 P94; 0x2 P80; 0x1 P31; 0x0 P04	0x0	R/W	
FPT6–7 Interrupt Port Select Register (GPIO_FPT67_SEL)	**0x300343** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SPT7[2:0]**	FPT7 interrupt input port select	SPT7[2:0] Port: 0x7 PF1; 0x6 PD7; 0x5 PB7; 0x4 PA7; 0x3 P97; 0x2 P83; 0x1 P60; 0x0 P07	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SPT6[2:0]**	FPT6 interrupt input port select	SPT6[2:0] Port: 0x7 PF0; 0x6 PD6; 0x5 PB6; 0x4 PA6; 0x3 P96; 0x2 P82; 0x1 P33; 0x0 P06	0x0	R/W	
FPT0–3 Interrupt Polarity Select Register (GPIO_FPT03_POL)	**0x300344** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SPPT3**	FPT3 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D2	**SPPT2**	FPT2 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D1	**SPPT1**	FPT1 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D0	**SPPT0**	FPT0 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
FPT4–7 Interrupt Polarity Select Register (GPIO_FPT47_POL)	**0x300345** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SPPT7**	FPT7 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D2	**SPPT6**	FPT6 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D1	**SPPT5**	FPT5 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D0	**SPPT4**	FPT4 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
FPT0–3 Interrupt Mode Select Register (GPIO_FPT03_MOD)	**0x300346** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SEPT3**	FPT3 interrupt mode select	1 Edge 0 Level	1	R/W	
		D2	**SEPT2**	FPT2 interrupt mode select	1 Edge 0 Level	1	R/W	
		D1	**SEPT1**	FPT1 interrupt mode select	1 Edge 0 Level	1	R/W	
		D0	**SEPT0**	FPT0 interrupt mode select	1 Edge 0 Level	1	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT4–7 Interrupt Mode Select Register (GIO_FPT47_ MOD)	**0x300347** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SEPT7**	FPT7 interrupt mode select	1 Edge 0 Level	1	R/W	
		D2	**SEPT6**	FPT6 interrupt mode select	1 Edge 0 Level	1	R/W	
		D1	**SEPT5**	FPT5 interrupt mode select	1 Edge 0 Level	1	R/W	
		D0	**SEPT4**	FPT4 interrupt mode select	1 Edge 0 Level	1	R/W	
FPT0–3 Interrupt Mask Register (GPIO_FPT03_ MSK)	**0x300348** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SIET3**	FPT3 interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**SIET2**	FPT2 interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**SIET1**	FPT1 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**SIET0**	FPT0 interrupt enable	1 Enable 0 Disable	0	R/W	
FPT4–7 Interrupt Mask Register (GPIO_FPT47_ MSK)	**0x300349** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SIET7**	FPT7 interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**SIET6**	FPT6 interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**SIET5**	FPT5 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**SIET4**	FPT4 interrupt enable	1 Enable 0 Disable	0	R/W	
FPT0–3 Interrupt Flag Register (GPIO_FPT03_ FLG)	**0x30034a** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SFGP3**	FPT3 interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	X	R/W	Reset by writing 1.
		D2	**SFGP2**	FPT2 interrupt flag		X	R/W	
		D1	**SFGP1**	FPT1 interrupt flag		X	R/W	
		D0	**SFGP0**	FPT0 interrupt flag		X	R/W	
FPT4–7 Interrupt Flag Register (GPIO_FPT47_ FLG)	**0x30034b** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SFGP7**	FPT7 interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	X	R/W	Reset by writing 1.
		D2	**SFGP6**	FPT6 interrupt flag		X	R/W	
		D1	**SFGP5**	FPT5 interrupt flag		X	R/W	
		D0	**SFGP4**	FPT4 interrupt flag		X	R/W	
FPT0–1 Interrupt Chattering Filter Control Register (GPIO_FPT01_ CHAT)	**0x30034c** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SCTP1[2:0]**	FPT1 chattering filter time select	SCTP1[2:0] / Filter sampling time 0x7 64/fPCLK 0x6 32/fPCLK 0x5 16/fPCLK 0x4 8/fPCLK 0x3 4/fPCLK 0x2 2/fPCLK 0x1 1/fPCLK 0x0 None	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SCTP0[2:0]**	FPT0 chattering filter time select	SCTP0[2:0] / Filter sampling time 0x7 64/fPCLK 0x6 32/fPCLK 0x5 16/fPCLK 0x4 8/fPCLK 0x3 4/fPCLK 0x2 2/fPCLK 0x1 1/fPCLK 0x0 None	0x0	R/W	
FPT2–3 Interrupt Chattering Filter Control Register (GPIO_FPT23_ CHAT)	**0x30034d** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SCTP3[2:0]**	FPT3 chattering filter time select	SCTP3[2:0] / Filter sampling time 0x7 64/fPCLK 0x6 32/fPCLK 0x5 16/fPCLK 0x4 8/fPCLK 0x3 4/fPCLK 0x2 2/fPCLK 0x1 1/fPCLK 0x0 None	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SCTP2[2:0]**	FPT2 chattering filter time select	SCTP2[2:0] / Filter sampling time 0x7 64/fPCLK 0x6 32/fPCLK 0x5 16/fPCLK 0x4 8/fPCLK 0x3 4/fPCLK 0x2 2/fPCLK 0x1 1/fPCLK 0x0 None	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPT4–5 Interrupt Chattering Filter Control Register (GPIO_FPT45_CHAT)	**0x30034e** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SCTP5[2:0]**	FPT5 chattering filter time select	SCTP5[2:0] / Filter sampling time: 0x7: 64/fPCLK 0x6: 32/fPCLK 0x5: 16/fPCLK 0x4: 8/fPCLK 0x3: 4/fPCLK 0x2: 2/fPCLK 0x1: 1/fPCLK 0x0: None	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SCTP4[2:0]**	FPT4 chattering filter time select	SCTP4[2:0] / Filter sampling time: 0x7: 64/fPCLK 0x6: 32/fPCLK 0x5: 16/fPCLK 0x4: 8/fPCLK 0x3: 4/fPCLK 0x2: 2/fPCLK 0x1: 1/fPCLK 0x0: None	0x0	R/W	
FPT6–7 Interrupt Chattering Filter Control Register (GPIO_FPT67_CHAT)	**0x30034f** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SCTP7[2:0]**	FPT7 chattering filter time select	SCTP7[2:0] / Filter sampling time: 0x7: 64/fPCLK 0x6: 32/fPCLK 0x5: 16/fPCLK 0x4: 8/fPCLK 0x3: 4/fPCLK 0x2: 2/fPCLK 0x1: 1/fPCLK 0x0: None	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SCTP6[2:0]**	FPT6 chattering filter time select	SCTP6[2:0] / Filter sampling time: 0x7: 64/fPCLK 0x6: 32/fPCLK 0x5: 16/fPCLK 0x4: 8/fPCLK 0x3: 4/fPCLK 0x2: 2/fPCLK 0x1: 1/fPCLK 0x0: None	0x0	R/W	
Port DMA Trigger Source Select Register (GPIO_DMA)	**0x300350** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SDMAT[2:0]**	Port DMA trigger source select	SDMAT[2:0] / Trigger source: 0x7: FPT7 0x6: FPT6 0x5: FPT5 0x4: FPT4 0x3: FPT3 0x2: FPT2 0x1: FPT1 0x0: FPT0	0x0	R/W	
FPK000–007 Interrupt Select Register (GPIO_FPK0_07_SEL)	**0x300360** (8 bits)	D7	**SPK007**	FPK007 interrupt select	1: P07 input; 0: Disable	0	R/W	
		D6	**SPK006**	FPK006 interrupt select	1: P06 input; 0: Disable	0	R/W	
		D5	**SPK005**	FPK005 interrupt select	1: P05 input; 0: Disable	0	R/W	
		D4	**SPK004**	FPK004 interrupt select	1: P04 input; 0: Disable	0	R/W	
		D3	**SPK003**	FPK003 interrupt select	1: P03 input; 0: Disable	0	R/W	
		D2	**SPK002**	FPK002 interrupt select	1: P02 input; 0: Disable	0	R/W	
		D1	**SPK001**	FPK001 interrupt select	1: P01 input; 0: Disable	0	R/W	
		D0	**SPK000**	FPK000 interrupt select	1: P00 input; 0: Disable	0	R/W	
FPK008–015 Interrupt Select Register (GPIO_FPK0_8F_SEL)	**0x300361** (8 bits)	D7	**SPK015**	FPK015 interrupt select	1: PA7 input; 0: Disable	0	R/W	
		D6	**SPK014**	FPK014 interrupt select	1: PA6 input; 0: Disable	0	R/W	
		D5	**SPK013**	FPK013 interrupt select	1: PA5 input; 0: Disable	0	R/W	
		D4	**SPK012**	FPK012 interrupt select	1: PA4 input; 0: Disable	0	R/W	
		D3	**SPK011**	FPK011 interrupt select	1: PA3 input; 0: Disable	0	R/W	
		D2	**SPK010**	FPK010 interrupt select	1: PA2 input; 0: Disable	0	R/W	
		D1	**SPK009**	FPK009 interrupt select	1: PA1 input; 0: Disable	0	R/W	
		D0	**SPK008**	FPK008 interrupt select	1: PA0 input; 0: Disable	0	R/W	
FPK100–107 Interrupt Select Register (GPIO_FPK1_07_SEL)	**0x300362** (8 bits)	D7	**SPK107**	FPK107 interrupt select	1: P76 input; 0: Disable	0	R/W	
		D6	**SPK106**	FPK106 interrupt select	1: P74 input; 0: Disable	0	R/W	
		D5	**SPK105**	FPK105 interrupt select	1: P72 input; 0: Disable	0	R/W	
		D4	**SPK104**	FPK104 interrupt select	1: P70 input; 0: Disable	0	R/W	
		D3	**SPK103**	FPK103 interrupt select	1: P33 input; 0: Disable	0	R/W	
		D2	**SPK102**	FPK102 interrupt select	1: P32 input; 0: Disable	0	R/W	
		D1	**SPK101**	FPK101 interrupt select	1: P31 input; 0: Disable	0	R/W	
		D0	**SPK100**	FPK100 interrupt select	1: P30 input; 0: Disable	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPK108–115 Interrupt Select Register (GPIO_FPK1_ 8F_SEL)	**0x300363** (8 bits)	D7	**SPK115**	FPK115 interrupt select	1 PD7 input 0 Disable	0	R/W	
		D6	**SPK114**	FPK114 interrupt select	1 PD6 input 0 Disable	0	R/W	
		D5	**SPK113**	FPK113 interrupt select	1 PD5 input 0 Disable	0	R/W	
		D4	**SPK112**	FPK112 interrupt select	1 PD4 input 0 Disable	0	R/W	
		D3	**SPK111**	FPK111 interrupt select	1 PD3 input 0 Disable	0	R/W	
		D2	**SPK110**	FPK110 interrupt select	1 PD2 input 0 Disable	0	R/W	
		D1	**SPK109**	FPK109 interrupt select	1 PD1 input 0 Disable	0	R/W	
		D0	**SPK108**	FPK108 interrupt select	1 PD0 input 0 Disable	0	R/W	
FPK0–1 Interrupt Polarity Select Register (GPIO_FPK01_ POL)	**0x300370** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**SPKT1**	FPK1 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
		D0	**SPKT0**	FPK0 input polarity select	1 High / ↑ 0 Low / ↓	1	R/W	
FPK0–1 Interrupt Mode Select Register (GPIO_FPK01_ MOD)	**0x300371** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**SEKT1**	FPK1 interrupt mode select	1 Edge 0 Level	1	R/W	
		D0	**SEKT0**	FPK0 interrupt mode select	1 Edge 0 Level	1	R/W	
FPK0–1 Interrupt Mask Register (GPIO_FPK01_ MSK)	**0x300372** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**SIEKT1**	FPK1 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**SIEKT0**	FPK0 interrupt enable	1 Enable 0 Disable	0	R/W	
FPK0–1 Interrupt Flag Register (GPIO_FPK01_ FLG)	**0x300373** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**SFGK1**	FPK1 interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	X	R/W	
		D0	**SFGK0**	FPK0 interrupt flag		X	R/W	
FPK0–1 Interrupt Chattering Filter Control Register (GPIO_FPK01_ CHAT)	**0x300374** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–4	**SCTK1[2:0]**	FPK1 chattering filter time select	SCTK1[2:0] / Filter sampling time 0x7 64/fPCLK 0x6 32/fPCLK 0x5 16/fPCLK 0x4 8/fPCLK 0x3 4/fPCLK 0x2 2/fPCLK 0x1 1/fPCLK 0x0 None	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**SCTK0[2:0]**	FPK0 chattering filter time select	SCTK0[2:0] / Filter sampling time 0x7 64/fPCLK 0x6 32/fPCLK 0x5 16/fPCLK 0x4 8/fPCLK 0x3 4/fPCLK 0x2 2/fPCLK 0x1 1/fPCLK 0x0 None	0x0	R/W	
P0[3:0] Port Function Select Register (PMUX_P0_03)	**0x300800** (8 bits)	D7–6	**CFP03[1:0]**	P03 port function select	CFP03[1:0] / Function 0x3 HIF_D0 0x2 REMC_I 0x1 USI_CK0 0x0 P03	0x0	R/W	Write-protected
		D5–4	**CFP02[1:0]**	P02 port function select	CFP02[1:0] / Function 0x3 #HIF_CS 0x2 SCLK 0x1 USI_CS0 0x0 P02	0x0	R/W	
		D3–2	**CFP01[1:0]**	P01 port function select	CFP01[1:0] / Function 0x3 #HIF_RD 0x2 SOUT 0x1 USI_DO0 0x0 P01	0x0	R/W	
		D1–0	**CFP00[1:0]**	P00 port function select	CFP00[1:0] / Function 0x3 #HIF_WR 0x2 SIN 0x1 USI_DI0 0x0 P00	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P0[7:4] Port Function Select Register (PMUX_P0_47)	**0x300801** (8 bits)	D7–6	**CFP07[1:0]**	P07 port function select	CFP07[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D4			
					0x2	T16A_ATMB_1			
					0x1	USI_CK1			
					0x0	P07			
		D5–4	**CFP06[1:0]**	P06 port function select	CFP06[1:0]	Function	0x0	R/W	
					0x3	HIF_D3			
					0x2	T16A_ATMA_1			
					0x1	USI_CS1			
					0x0	P06			
		D3–2	**CFP05[1:0]**	P05 port function select	CFP05[1:0]	Function	0x0	R/W	
					0x3	HIF_D2			
					0x2	T16A_ATMB_0			
					0x1	USI_DO1			
					0x0	P05			
		D1–0	**CFP04[1:0]**	P04 port function select	CFP04[1:0]	Function	0x0	R/W	
					0x3	HIF_D1			
					0x2	T16A_ATMA_0			
					0x1	USI_DI1			
					0x0	P04			
P1[3:0] Port Function Select Register (PMUX_P1_03)	**0x300802** (8 bits)	D7–6	**CFP13[1:0]**	P13 port function select	CFP13[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D6			
					0x2	PWM_L0			
					0x1	I2S_WSO			
					0x0	P13			
		D5–4	**CFP12[1:0]**	P12 port function select	CFP12[1:0]	Function	0x0	R/W	
					0x3	HIF_D5			
					0x2	PWM_H0			
					0x1	I2S_SDO			
					0x0	P12			
		D3–2	**CFP11[1:0]**	P11 port function select	CFP11[1:0]	Function	0x0	R/W	
					0x3	SOUT			
					0x2	P11			
					0x1	TFT_CTL1			
					0x0	A25			
		D1–0	**CFP10[1:0]**	P10 port function select	CFP10[1:0]	Function	0x0	R/W	
					0x3	CMU_CLK			
					0x2	P10			
					0x1	TFT_CTL0			
					0x0	A24			
P1[7:4] Port Function Select Register (PMUX_P1_47)	**0x300803** (8 bits)	D7–6	**CFP17[1:0]**	P17 port function select	CFP17[1:0]	Function	0x0	R/W	Write-protected * When TRACE/ MISC_TRACE register = 0 (default)
					0x3	#NAND_RD			
					0x2	REMC_O			
					0x1	#CE5			
					0x0	P17			
					–	DPCO *			
		D5–4	**CFP16[1:0]**	P16 port function select	CFP16[1:0]	Function	0x0	R/W	
					0x3	#NAND_WR			
					0x2	CMU_CLK			
					0x1	#CE4			
					0x0	P16			
					–	DST1 *			
		D3–2	**CFP15[1:0]**	P15 port function select	CFP15[1:0]	Function	0x0	R/W	
					0x3	#HIF_WAIT			
					0x2	PWM_L1			
					0x1	I2S_MCLKO			
					0x0	P15			
					–	DST0 *			
		D1–0	**CFP14[1:0]**	P14 port function select	CFP14[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_D7			
					0x2	PWM_H1			
					0x1	I2S_SCLKO			
					0x0	P14			
P2[1:0] Port Function Select Register (PMUX_P2_01)	**0x300804** (8 bits)	D7–4	**–**	reserved	–		–	–	0 when being read.
		D3–2	**CFP21[1:0]**	P21 port function select	CFP21[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P21			
					0x0	SDCLK			
		D1–0	**CFP20[1:0]**	P20 port function select	CFP20[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P20			
					0x0	SDCKE			

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P3[3:0] Port Function Select Register (PMUX_P3_03)	**0x300806** (8 bits)	D7–6	**CFP33[1:0]**	P33 port function select	CFP33[1:0]	Function	0x0	R/W	Write-protected
					0x3	HIF_A3			
					0x2	T16A_ATMB_3			
					0x1	I2S_MCLKI			
					0x0	P33			
		D5–4	**CFP32[1:0]**	P32 port function select	CFP32[1:0]	Function	0x0	R/W	
					0x3	HIF_A2			
					0x2	T16A_ATMA_3			
					0x1	I2S_SCLKI			
					0x0	P32			
		D3–2	**CFP31[1:0]**	P31 port function select	CFP31[1:0]	Function	0x0	R/W	
					0x3	HIF_A1			
					0x2	T16A_ATMB_2			
					0x1	I2S_WSI			
					0x0	P31			
		D1–0	**CFP30[1:0]**	P30 port function select	CFP30[1:0]	Function	0x0	R/W	
					0x3	HIF_A0			
					0x2	T16A_ATMA_2			
					0x1	I2S_SDI			
					0x0	P30			
P3[6:4] Port Function Select Register (PMUX_P3_46)	**0x300807** (8 bits)	D7–6	**–**	reserved	–		–	–	0 when being read.
		D5–4	**CFP36[1:0]**	P36 port function select	CFP36[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P36			
					0x0	DST2			
		D3–2	**CFP35[1:0]**	P35 port function select	CFP35[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P35			
					0x0	DSIO			
		D1–0	**CFP34[1:0]**	P34 port function select	CFP34[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P34			
					0x0	DCLK			
P4[2:0] Port Function Select Register (PMUX_P4_02)	**0x300808** (8 bits)	D7–6	**–**	reserved	–		–	–	0 when being read.
		D5–4	**CFP42[1:0]**	P42 port function select	CFP42[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	TFT_CTL3			
					0x1	P42			
					0x0	A23			
		D3–2	**CFP41[1:0]**	P41 port function select	CFP41[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	TFT_CTL2			
					0x1	P41			
					0x0	A22			
		D1–0	**CFP40[1:0]**	P40 port function select	CFP40[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P40			
					0x0	A21			
P5[3:0] Port Function Select Register (PMUX_P5_03)	**0x30080a** (8 bits)	D7–6	**CFP53[1:0]**	P53 port function select	CFP53[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P53			
					0x0	#CE10			
		D5–4	**CFP52[1:0]**	P52 port function select	CFP52[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P52			
					0x0	#CE9			
		D3–2	**CFP51[1:0]**	P51 port function select	CFP51[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#HIF_INT			
					0x1	P51			
					0x0	#CE8			
		D1–0	**CFP50[1:0]**	P50 port function select	CFP50[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	#SDCS			
					0x1	P50			
					0x0	#CE7			

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P5[6:4] Port Function Select Register (PMUX_P5_46)	**0x30080b** (8 bits)	D7–6	**–**	reserved	–		–	–	0 when being read.
		D5–4	**CFP56[1:0]**	P56 port function select	CFP56[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	P56			
					0x0	#WRH/#BSH			
		D3–2	**CFP55[1:0]**	P55 port function select	CFP55[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P55			
					0x0	#WRL			
		D1–0	**CFP54[1:0]**	P54 port function select	CFP54[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	P54			
					0x0	#RD			
P60 Port Function Select Register (PMUX_P6_0)	**0x30080c** (8 bits)	D7–2	**–**	reserved	–		–	–	0 when being read.
		D1–0	**CFP60[1:0]**	P60 port function select	CFP60[1:0]	Function	0x0	R/W	Write-protected
					0x3	#WDT_NMI			
					0x2	WDT_CLK			
					0x1	#WAIT			
					0x0	P60			
P7[3:0] Port Function Select Register (PMUX_P7_03)	**0x30080e** (8 bits)	D7–6	**CFP73[1:0]**	P73 port function select	CFP73[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	I2S_SCLKI			
					0x1	AIN3			
					0x0	P73			
		D5–4	**CFP72[1:0]**	P72 port function select	CFP72[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL1			
					0x1	AIN2			
					0x0	P72			
		D3–2	**CFP71[1:0]**	P71 port function select	CFP71[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	I2S_WSI			
					0x1	AIN1			
					0x0	P71			
		D1–0	**CFP70[1:0]**	P70 port function select	CFP70[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	EXCL0			
					0x1	AIN0			
					0x0	P70			
P7[7:4] Port Function Select Register (PMUX_P7_47)	**0x30080f** (8 bits)	D7–6	**CFP77[1:0]**	P77 port function select	CFP77[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	AIN7			
					0x0	P77			
		D5–4	**CFP76[1:0]**	P76 port function select	CFP76[1:0]	Function	0x0	R/W	
					0x3	#ADTRIG			
					0x2	#WAIT			
					0x1	AIN6			
					0x0	P76			
		D3–2	**CFP75[1:0]**	P75 port function select	CFP75[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	I2S_MCLKI			
					0x1	AIN5			
					0x0	P75			
		D1–0	**CFP74[1:0]**	P74 port function select	CFP74[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	WDT_EXCL			
					0x1	AIN4			
					0x0	P74			

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
P8[3:0] Port Function Select Register (PMUX_P8_03)	**0x300810** (8 bits)	D7–6	**CFP83[1:0]**	P83 port function select	CFP83[1:0]	Function	0x0	R/W	Write-protected
					0x3	MMCD3			
					0x2	USIL_CK			
					0x1	FPDRDY			
					0x0	P83			
		D5–4	**CFP82[1:0]**	P82 port function select	CFP82[1:0]	Function	0x0	R/W	
					0x3	MMCD2			
					0x2	USIL_CS			
					0x1	FPSHIFT			
					0x0	P82			
		D3–2	**CFP81[1:0]**	P81 port function select	CFP81[1:0]	Function	0x0	R/W	
					0x3	MMCD1			
					0x2	USIL_DO			
					0x1	FPLINE			
					0x0	P81			
		D1–0	**CFP80[1:0]**	P80 port function select	CFP80[1:0]	Function	0x0	R/W	
					0x3	MMCD0			
					0x2	USIL_DI			
					0x1	FPFRAME			
					0x0	P80			
P9[3:0] Port Function Select Register (PMUX_P9_03)	**0x300812** (8 bits)	D7–6	**CFP93[1:0]**	P93 port function select	CFP93[1:0]	Function	0x0	R/W	Write-protected
					0x3	#NAND_RD			
					0x2	USIL_LCD_D3			
					0x1	FPDAT3			
					0x0	P93			
		D5–4	**CFP92[1:0]**	P92 port function select	CFP92[1:0]	Function	0x0	R/W	
					0x3	#NAND_WR			
					0x2	USIL_LCD_D2			
					0x1	FPDAT2			
					0x0	P92			
		D3–2	**CFP91[1:0]**	P91 port function select	CFP91[1:0]	Function	0x0	R/W	
					0x3	MMCCLK			
					0x2	USIL_LCD_D1			
					0x1	FPDAT1			
					0x0	P91			
		D1–0	**CFP90[1:0]**	P90 port function select	CFP90[1:0]	Function	0x0	R/W	
					0x3	MMCCMD			
					0x2	USIL_LCD_D0			
					0x1	FPDAT0			
					0x0	P90			
P9[7:4] Port Function Select Register (PMUX_P9_47)	**0x300813** (8 bits)	D7–6	**CFP97[1:0]**	P97 port function select	CFP97[1:0]	Function	0x0	R/W	Write-protected
					0x3	USI_CK2			
					0x2	USIL_LCD_D7			
					0x1	FPDAT7			
					0x0	P97			
		D5–4	**CFP96[1:0]**	P96 port function select	CFP96[1:0]	Function	0x0	R/W	
					0x3	USI_CS2			
					0x2	USIL_LCD_D6			
					0x1	FPDAT6			
					0x0	P96			
		D3–2	**CFP95[1:0]**	P95 port function select	CFP95[1:0]	Function	0x0	R/W	
					0x3	USI_DO2			
					0x2	USIL_LCD_D5			
					0x1	FPDAT5			
					0x0	P95			
		D1–0	**CFP94[1:0]**	P94 port function select	CFP94[1:0]	Function	0x0	R/W	
					0x3	USI_DI2			
					0x2	USIL_LCD_D4			
					0x1	FPDAT4			
					0x0	P94			

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PA[3:0] Port Function Select Register (PMUX_PA_03)	**0x300814** (8 bits)	D7–6	**CFPA3[1:0]**	PA3 port function select	CFPA3[1:0]	Function	0x0	R/W	Write-protected
					0x3	FPDAT19			
					0x2	MMCD3			
					0x1	HIF_D11			
					0x0	PA3			
		D5–4	**CFPA2[1:0]**	PA2 port function select	CFPA2[1:0]	Function	0x0	R/W	
					0x3	FPDAT18			
					0x2	MMCD2			
					0x1	HIF_D10			
					0x0	PA2			
		D3–2	**CFPA1[1:0]**	PA1 port function select	CFPA1[1:0]	Function	0x0	R/W	
					0x3	FPDAT17			
					0x2	MMCD1			
					0x1	HIF_D9			
					0x0	PA1			
		D1–0	**CFPA0[1:0]**	PA0 port function select	CFPA0[1:0]	Function	0x0	R/W	
					0x3	FPDAT16			
					0x2	MMCD0			
					0x1	HIF_D8			
					0x0	PA0			
PA[7:4] Port Function Select Register (PMUX_PA_47)	**0x300815** (8 bits)	D7–6	**CFPA7[1:0]**	PA7 port function select	CFPA7[1:0]	Function	0x0	R/W	Write-protected
					0x3	FPDAT23			
					0x2	T16A_ATMB_0			
					0x1	HIF_D15			
					0x0	PA7			
		D5–4	**CFPA6[1:0]**	PA6 port function select	CFPA6[1:0]	Function	0x0	R/W	
					0x3	FPDAT22			
					0x2	T16A_ATMA_0			
					0x1	HIF_D14			
					0x0	PA6			
		D3–2	**CFPA5[1:0]**	PA5 port function select	CFPA5[1:0]	Function	0x0	R/W	
					0x3	FPDAT21			
					0x2	MMCCLK			
					0x1	HIF_D13			
					0x0	PA5			
		D1–0	**CFPA4[1:0]**	PA4 port function select	CFPA4[1:0]	Function	0x0	R/W	
					0x3	FPDAT20			
					0x2	MMCCMD			
					0x1	HIF_D12			
					0x0	PA4			
PB[3:0] Port Function Select Register (PMUX_PB_03)	**0x300816** (8 bits)	D7–6	**CFPB3[1:0]**	PB3 port function select	CFPB3[1:0]	Function	0x0	R/W	Write-protected
					0x3	MMCD3			
					0x2	REMC_I			
					0x1	FPDAT11			
					0x0	PB3			
		D5–4	**CFPB2[1:0]**	PB2 port function select	CFPB2[1:0]	Function	0x0	R/W	
					0x3	MMCD2			
					0x2	REMC_O			
					0x1	FPDAT10			
					0x0	PB2			
		D3–2	**CFPB1[1:0]**	PB1 port function select	CFPB1[1:0]	Function	0x0	R/W	
					0x3	MMCD1			
					0x2	#NAND_RD			
					0x1	FPDAT9			
					0x0	PB1			
		D1–0	**CFPB0[1:0]**	PB0 port function select	CFPB0[1:0]	Function	0x0	R/W	
					0x3	MMCD0			
					0x2	#NAND_WR			
					0x1	FPDAT8			
					0x0	PB0			

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PB[7:4] Port Function Select Register (PMUX_PB_47)	**0x300817** (8 bits)	D7–6	**CFPB7[1:0]**	PB7 port function select	CFPB7[1:0]	Function	0x0	R/W	Write-protected
					0x3	T16A_ATMB_1			
					0x2	USI_CK2			
					0x1	FPDAT15			
					0x0	PB7			
		D5–4	**CFPB6[1:0]**	PB6 port function select	CFPB6[1:0]	Function	0x0	R/W	
					0x3	T16A_ATMA_1			
					0x2	USI_CS2			
					0x1	FPDAT14			
					0x0	PB6			
		D3–2	**CFPB5[1:0]**	PB5 port function select	CFPB5[1:0]	Function	0x0	R/W	
					0x3	MMCCLK			
					0x2	USI_DO2			
					0x1	FPDAT13			
					0x0	PB5			
		D1–0	**CFPB4[1:0]**	PB4 port function select	CFPB4[1:0]	Function	0x0	R/W	
					0x3	MMCCMD			
					0x2	USI_DI2			
					0x1	FPDAT12			
					0x0	PB4			
PC[3:0] Port Function Select Register (PMUX_PC_03)	**0x300818** (8 bits)	D7–6	**CFPC3[1:0]**	PC3 port function select	CFPC3[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	PC3			
					0x0	D11			
		D5–4	**CFPC2[1:0]**	PC2 port function select	CFPC2[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC2			
					0x0	D10			
		D3–2	**CFPC1[1:0]**	PC1 port function select	CFPC1[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC1			
					0x0	D9			
		D1–0	**CFPC0[1:0]**	PC0 port function select	CFPC0[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC0			
					0x0	D8			
PC[7:4] Port Function Select Register (PMUX_PC_47)	**0x300819** (8 bits)	D7–6	**CFPC7[1:0]**	PC7 port function select	CFPC7[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	reserved			
					0x1	PC7			
					0x0	D15			
		D5–4	**CFPC6[1:0]**	PC6 port function select	CFPC6[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC6			
					0x0	D14			
		D3–2	**CFPC5[1:0]**	PC5 port function select	CFPC5[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC5			
					0x0	D13			
		D1–0	**CFPC4[1:0]**	PC4 port function select	CFPC4[1:0]	Function	0x0	R/W	
					0x3	reserved			
					0x2	reserved			
					0x1	PC4			
					0x0	D12			

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
PD[3:0] Port Function Select Register (PMUX_PD_03)	**0x30081a** (8 bits)	D7–6	**CFPD3[1:0]**	PD3 port function select	CFPD3[1:0]	Function	0x0	R/W	Write-protected
					0x3	reserved			
					0x2	USI_CK0			
					0x1	FPDAT19			
					0x0	PD3			
		D5–4	**CFPD2[1:0]**	PD2 port function select	CFPD2[1:0]	Function	0x0	R/W	
					0x3	SCLK			
					0x2	USI_CS0			
					0x1	FPDAT18			
					0x0	PD2			
		D3–2	**CFPD1[1:0]**	PD1 port function select	CFPD1[1:0]	Function	0x0	R/W	
					0x3	SOUT			
					0x2	USI_DO0			
					0x1	FPDAT17			
					0x0	PD1			
		D1–0	**CFPD0[1:0]**	PD0 port function select	CFPD0[1:0]	Function	0x0	R/W	
					0x3	SIN			
					0x2	USI_DI0			
					0x1	FPDAT16			
					0x0	PD0			
PD[7:4] Port Function Select Register (PMUX_PD_47)	**0x30081b** (8 bits)	D7–6	**CFPD7[1:0]**	PD7 port function select	CFPD7[1:0]	Function	0x0	R/W	Write-protected
					0x3	T16A_ATMB_0			
					0x2	USI_CK1			
					0x1	FPDAT23			
					0x0	PD7			
		D5–4	**CFPD6[1:0]**	PD6 port function select	CFPD6[1:0]	Function	0x0	R/W	
					0x3	T16A_ATMA_0			
					0x2	USI_CS1			
					0x1	FPDAT22			
					0x0	PD6			
		D3–2	**CFPD5[1:0]**	PD5 port function select	CFPD5[1:0]	Function	0x0	R/W	
					0x3	#NAND_RD			
					0x2	USI_DO1			
					0x1	FPDAT21			
					0x0	PD5			
		D1–0	**CFPD4[1:0]**	PD4 port function select	CFPD4[1:0]	Function	0x0	R/W	
					0x3	#NAND_WR			
					0x2	USI_DI1			
					0x1	FPDAT20			
					0x0	PD4			
PE[3:0] Port Function Select Register (PMUX_PE_03)	**0x30081c** (8 bits)	D7–6	**CFPE3[1:0]**	PE3 port function select	CFPE3[1:0]	Function	0x0	R/W	Write-protected
					0x3	USI_CK2			
					0x2	T16A_ATMB_3			
					0x1	MMCD3			
					0x0	PE3			
		D5–4	**CFPE2[1:0]**	PE2 port function select	CFPE2[1:0]	Function	0x0	R/W	
					0x3	USI_CS2			
					0x2	T16A_ATMA_3			
					0x1	MMCD2			
					0x0	PE2			
		D3–2	**CFPE1[1:0]**	PE1 port function select	CFPE1[1:0]	Function	0x0	R/W	
					0x3	USI_DO2			
					0x2	T16A_ATMB_2			
					0x1	MMCD1			
					0x0	PE1			
		D1–0	**CFPE0[1:0]**	PE0 port function select	CFPE0[1:0]	Function	0x0	R/W	
					0x3	USI_DI2			
					0x2	T16A_ATMA_2			
					0x1	MMCD0			
					0x0	PE0			
PE[5:4] Port Function Select Register (PMUX_PE_45)	**0x30081d** (8 bits)	D7–4	**–**	reserved	–		–	–	0 when being read.
		D3–2	**CFPE5[1:0]**	PE5 port function select	CFPE5[1:0]	Function	0x0	R/W	Write-protected
					0x3	REMC_I			
					0x2	T16A_ATMB_1			
					0x1	MMCCLK			
					0x0	PE5			
		D1–0	**CFPE4[1:0]**	PE4 port function select	CFPE4[1:0]	Function	0x0	R/W	
					0x3	REMC_O			
					0x2	T16A_ATMA_1			
					0x1	MMCCMD			
					0x0	PE4			

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PF[3:0] Port Function Select Register (PMUX_PF_03)	**0x30081e** (8 bits)	D7–6	**CFPF3[1:0]**	PF3 port function select	CFPF3[1:0] Function 0x3 TFT_CTL3 0x2 PWM_L1 0x1 I2S_MCLKO 0x0 PF3	0x0	R/W	Write-protected
		D5–4	**CFPF2[1:0]**	PF2 port function select	CFPF2[1:0] Function 0x3 TFT_CTL2 0x2 PWM_H1 0x1 I2S_SCLKO 0x0 PF2	0x0	R/W	
		D3–2	**CFPF1[1:0]**	PF1 port function select	CFPF1[1:0] Function 0x3 TFT_CTL1 0x2 PWM_L0 0x1 I2S_WSO 0x0 PF1	0x0	R/W	
		D1–0	**CFPF0[1:0]**	PF0 port function select	CFPF0[1:0] Function 0x3 TFT_CTL0 0x2 PWM_H0 0x1 I2S_SDO 0x0 PF0	0x0	R/W	
PF4 Port Function Select Register (PMUX_PF_4)	**0x30081f** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1–0	**CFPF4[1:0]**	PF4 port function select	CFPF4[1:0] Function 0x3 reserved 0x2 #ADTRIG 0x1 I2S_SDI 0x0 PF4	0x0	R/W	Write-protected
P0 Port Pull-up Control Register (GPIO_P0_PUP)	**0x300820** (8 bits)	D7–0	**PUP0[7:0]**	P0[7:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P0[7:0]: not pulled up Write-protected
P1 Port Pull-up Control Register (GPIO_P1_PUP)	**0x300821** (8 bits)	D7–0	**PUP1[7:0]**	P1[7:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P1[7:0]: not pulled up Write-protected
P2 Port Pull-up Control Register (GPIO_P2_PUP)	**0x300822** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1–0	**PUP2[1:0]**	P2[1:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P2[1:0]: not pulled up Write-protected
P3 Port Pull-up Control Register (GPIO_P3_PUP)	**0x300823** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–0	**PUP3[6:0]**	P3[6:0] port pull-up enable	1 Enable 0 Disable	0x20 *	R/W	* P35: pulled up, others: not pulled up Write-protected
P4 Port Pull-up Control Register (GPIO_P4_PUP)	**0x300824** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**PUP4[2:0]**	P4[2:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P4[2:0]: not pulled up Write-protected
P5 Port Pull-up Control Register (GPIO_P5_PUP)	**0x300825** (8 bits)	D7	**–**	reserved	–	–	–	0 when being read.
		D6–0	**PUP5[6:0]**	P5[6:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P5[6:0]: not pulled up Write-protected
P6 Port Pull-up Control Register (GPIO_P6_PUP)	**0x300826** (8 bits)	D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**PUP60**	P60 port pull-up enable	1 Enable 0 Disable	1 *	R/W	* P60: pulled up Write-protected
P7 Port Pull-up Control Register (GPIO_P7_PUP)	**0x300827** (8 bits)	D7–0	**PUP7[7:0]**	P7[7:0] port pull-up enable	1 Enable 0 Disable	0x40 *	R/W	* P76: pulled up, others: not pulled up Write-protected
P8 Port Pull-up Control Register (GPIO_P8_PUP)	**0x300828** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**PUP8[3:0]**	P8[3:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P8[3:0]: not pulled up Write-protected
P9 Port Pull-up Control Register (GPIO_P9_PUP)	**0x300829** (8 bits)	D7–0	**PUP9[7:0]**	P9[7:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* P9[7:0]: not pulled up Write-protected
PA Port Pull-up Control Register (GPIO_PA_PUP)	**0x30082a** (8 bits)	D7–0	**PUPA[7:0]**	PA[7:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* PA[7:0]: not pulled up Write-protected
PB Port Pull-up Control Register (GPIO_PB_PUP)	**0x30082b** (8 bits)	D7–0	**PUPB[7:0]**	PB[7:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* PB[7:0]: not pulled up Write-protected
PD Port Pull-up Control Register (GPIO_PD_PUP)	**0x30082d** (8 bits)	D7–0	**PUPD[7:0]**	PD[7:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* PD[7:0]: not pulled up Write-protected

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PE Port Pull-up Control Register (GPIO_PE_PUP)	0x30082e (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5–0	PUPE[5:0]	PE[5:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* PE[5:0]: not pulled up Write-protected
PF Port Pull-up Control Register (GPIO_PF_PUP)	0x30082f (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4–0	PUPF[4:0]	PF[4:0] port pull-up enable	1 Enable 0 Disable	0x0 *	R/W	* PF[4:0]: not pulled up Write-protected
Bus Drive Control Register (GPIO_BUS_ DRV)	0x300830 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	LDRVDB	D[15:0] low drive	1 Low drive 0 Normal output	0	R/W	Write-protected
		D0	LDRVAD	A[25:0] low drive		0	R/W	
Port Noise Filter Control Register (GPIO_FILTER)	0x30083e (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	ANFEN	Input port noise filter enable	1 Enable 0 Disable	0	R/W	Write-protected
GPIO/PMUX Write Protect Register (GPIO_ PROTECT)	0x30083f (8 bits)	D7–0	PPROT[7:0]	GPIO/PMUX register protect flag	Writing 10010110 (0x96) removes the write protection of the GPIO and PMUX registers (0x300800–0x300830 and 0x30083e). Writing another value set the write protection.	0x0	R/W	

0x300400–0x300472 USI Ch.0

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.0 Global Configuration Register (USI_GCFG0)	0x300400 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	LSBFST	MSB/LSB first mode select	1 MSB first 0 LSB first	0	R/W	
		D2–0	USIMOD [2:0]	Interface mode configuration	USIMOD[2:0] I/F mode 0x7–0x6 reserved 0x5 I²C slave 0x4 I²C master 0x3 SPI slave 0x2 SPI master 0x1 UART 0x0 Software reset	0x0	R/W	
USI Ch.0 Transmit Data Buffer Register (USI_TD0)	0x300401 (8 bits)	D7–0	TD[7:0]	Transmit data buffer TD7 = MSB TD0 = LSB	0x0 to 0xff	0x0	R/W	
USI Ch.0 Receive Data Buffer Register (USI_RD0)	0x300402 (8 bits)	D7–0	RD[7:0]	Receive data buffer RD7 = MSB RD0 = LSB	0x0 to 0xff	0x0	R	
USI Ch.0 UART Mode Configuration Register (USI_UCFG0)	0x300440 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	UCHLN	Character length select	1 8 bits 0 7 bits	0	R/W	
		D2	USTPB	Stop bit select	1 2 bits 0 1 bit	0	R/W	
		D1	UPMD	Parity mode select	1 Even 0 Odd	0	R/W	
		D0	UPREN	Parity enable	1 With parity 0 No parity	0	R/W	
USI Ch.0 UART Mode Interrupt Enable Register (USI_UIE0)	0x300441 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2	UEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	URDIE	Receive buffer full interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	UTDIE	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.0 UART Mode Interrupt Flag Register (USI_UIF0)	0x300442 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	URBSY	Receive busy flag	1 Busy 0 Idle	0	R	
		D5	UTBSY	Transmit busy flag	1 Busy 0 Idle	0	R	
		D4	UPEIF	Parity error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D3	USEIF	Framing error flag	1 Error 0 Normal	0	R/W	
		D2	UOEIF	Overrun error flag	1 Error 0 Normal	0	R/W	
		D1	URDIF	Receive buffer full flag	1 Full 0 Not full	0	R/W	
		D0	UTDIF	Transmit buffer empty flag	1 Empty 0 Not empty	0	R/W	
USI Ch.0 SPI Master/Slave Mode Configu-ration Register (USI_SCFG0)	0x300450 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	SCMD	Command bit (for 9-bit data)	1 High 0 Low	0	R/W	
		D4	SCHLN	Character length select	1 9 bits 0 8 bits	0	R/W	
		D3	SCPHA	Clock phase select	1 Phase 1 0 Phase 0	0	R/W	
		D2	SCPOL	Clock polarity select	1 Active L 0 Active H	0	R/W	
		D1	–	reserved	–	–	–	Do not set to 1.
		D0	SFSTMOD	Fast mode select	1 Fast 0 Normal	0	R/W	
USI Ch.0 SPI Master/Slave Mode Interrupt Enable Register (USI_SIE0)	0x300451 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2	SEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	SRDIE	Receive buffer full interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	STDIE	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.0 SPI Master/Slave Mode Interrupt Flag Register (USI_SIF0)	**0x300452** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SSIF**	Transfer busy flag (master)	1 Busy 0 Idle	0	R	
				ss signal low flag (slave)	1 ss = H 0 ss = L			
		D2	**SEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D1	**SRDIF**	Receive buffer full flag	1 Full 0 Not full	0	R/W	
		D0	**STDIF**	Transmit buffer empty flag	1 Empty 0 Not empty	0	R/W	
USI Ch.0 I²C Master Mode Trigger Register (USI_IMTG0)	**0x300460** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**IMTG**	I²C master operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**IMTGMOD [2:0]**	I²C master trigger mode select	IMTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data 0x2 Transmit data 0x1 Stop condition 0x0 Start condition	0x0	R/W	
USI Ch.0 I²C Master Mode Interrupt Enable Register (USI_IMIE0)	**0x300461** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**IMEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**IMIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.0 I²C Master Mode Interrupt Flag Register (USI_IMIF0)	**0x300462** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**IMBSY**	I²C master busy flag	1 Busy 0 Standby	0	R	
		D4–2	**IMSTA[2:0]**	I²C master status	IMSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop generated 0x0 Start generated	0x0	R	
		D1	**IMEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**IMIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	
USI Ch.0 I²C Slave Mode Trigger Register (USI_ISTG0)	**0x300470** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**ISTG**	I²C slave operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**ISTGMOD [2:0]**	I²C slave trigger mode select	ISTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data/ Detect stop 0x2 Transmit data 0x1 reserved 0x0 Wait for start	0x0	R/W	
USI Ch.0 I²C Slave Mode Interrupt Enable Register (USI_ISIE0)	**0x300471** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**ISEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**ISIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.0 I²C Slave Mode Interrupt Flag Register (USI_ISIF0)	**0x300472** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**ISBSY**	I²C slave busy flag	1 Busy 0 Standby	0	R	
		D4–2	**ISSTA[2:0]**	I²C slave status	ISSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop detected 0x0 Start detected	0x0	R	
		D1	**ISEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**ISIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	

0x300500–0x300572 USI Ch.1

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.1 Global Configuration Register (USI_GCFG1)	0x300500 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	LSBFST	MSB/LSB first mode select	1 MSB first 0 LSB first	0	R/W	
		D2–0	USIMOD [2:0]	Interface mode configuration	USIMOD[2:0] I/F mode 0x7–0x6 reserved 0x5 I²C slave 0x4 I²C master 0x3 SPI slave 0x2 SPI master 0x1 UART 0x0 Software reset	0x0	R/W	
USI Ch.1 Transmit Data Buffer Register (USI_TD1)	0x300501 (8 bits)	D7–0	TD[7:0]	Transmit data buffer TD7 = MSB TD0 = LSB	0x0 to 0xff	0x0	R/W	
USI Ch.1 Receive Data Buffer Register (USI_RD1)	0x300502 (8 bits)	D7–0	RD[7:0]	Receive data buffer RD7 = MSB RD0 = LSB	0x0 to 0xff	0x0	R	
USI Ch.1 UART Mode Configuration Register (USI_UCFG1)	0x300540 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	UCHLN	Character length select	1 8 bits 0 7 bits	0	R/W	
		D2	USTPB	Stop bit select	1 2 bits 0 1 bit	0	R/W	
		D1	UPMD	Parity mode select	1 Even 0 Odd	0	R/W	
		D0	UPREN	Parity enable	1 With parity 0 No parity	0	R/W	
USI Ch.1 UART Mode Interrupt Enable Register (USI_UIE1)	0x300541 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2	UEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	URDIE	Receive buffer full interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	UTDIE	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.1 UART Mode Interrupt Flag Register (USI_UIF1)	0x300542 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6	URBSY	Receive busy flag	1 Busy 0 Idle	0	R	
		D5	UTBSY	Transmit busy flag	1 Busy 0 Idle	0	R	
		D4	UPEIF	Parity error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D3	USEIF	Framing error flag	1 Error 0 Normal	0	R/W	
		D2	UOEIF	Overrun error flag	1 Error 0 Normal	0	R/W	
		D1	URDIF	Receive buffer full flag	1 Full 0 Not full	0	R/W	
		D0	UTDIF	Transmit buffer empty flag	1 Empty 0 Not empty	0	R/W	
USI Ch.1 SPI Master/Slave Mode Configuration Register (USI_SCFG1)	0x300550 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	SCMD	Command bit (for 9-bit data)	1 High 0 Low	0	R/W	
		D4	SCHLN	Character length select	1 9 bits 0 8 bits	0	R/W	
		D3	SCPHA	Clock phase select	1 Phase 1 0 Phase 0	0	R/W	
		D2	SCPOL	Clock polarity select	1 Active L 0 Active H	0	R/W	
		D1	–	reserved	–	–	–	Do not set to 1.
		D0	SFSTMOD	Fast mode select	1 Fast 0 Normal	0	R/W	
USI Ch.1 SPI Master/Slave Mode Interrupt Enable Register (USI_SIE1)	0x300551 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2	SEIE	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	SRDIE	Receive buffer full interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	STDIE	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.1 SPI Master/Slave Mode Interrupt Flag Register (USI_SIF1)	0x300552 (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	SSIF	Transfer busy flag (master)	1 Busy 0 Idle	0	R	
				ss signal low flag (slave)	1 ss = H 0 ss = L			
		D2	SEIF	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D1	SRDIF	Receive buffer full flag	1 Full 0 Not full	0	R/W	
		D0	STDIF	Transmit buffer empty flag	1 Empty 0 Not empty	0	R/W	
USI Ch.1 I²C Master Mode Trigger Register (USI_IMTG1)	0x300560 (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	IMTG	I²C master operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	IMTGMOD [2:0]	I²C master trigger mode select	IMTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data 0x2 Transmit data 0x1 Stop condition 0x0 Start condition	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.1 I²C Master Mode Interrupt Enable Register (USI_IMIE1)	**0x300561** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**IMEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**IMIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.1 I²C Master Mode Interrupt Flag Register (USI_IMIF1)	**0x300562** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**IMBSY**	I²C master busy flag	1 Busy 0 Standby	0	R	
		D4–2	**IMSTA[2:0]**	I²C master status	IMSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop generated 0x0 Start generated	0x0	R	
		D1	**IMEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**IMIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	
USI Ch.1 I²C Slave Mode Trigger Register (USI_ISTG1)	**0x300570** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**ISTG**	I²C slave operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**ISTGMOD [2:0]**	I²C slave trigger mode select	ISTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data/ Detect stop 0x2 Transmit data 0x1 reserved 0x0 Wait for start	0x0	R/W	
USI Ch.1 I²C Slave Mode Interrupt Enable Register (USI_ISIE1)	**0x300571** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**ISEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**ISIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.1 I²C Slave Mode Interrupt Flag Register (USI_ISIF1)	**0x300572** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**ISBSY**	I²C slave busy flag	1 Busy 0 Standby	0	R	
		D4–2	**ISSTA[2:0]**	I²C slave status	ISSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop detected 0x0 Start detected	0x0	R	
		D1	**ISEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**ISIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	

0x300600–0x30069f USIL

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL Global Configuration Register (USIL_GCFG)	**0x300600** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**LSBFST**	MSB/LSB first mode select	1 MSB first 0 LSB first	0	R/W	
		D2–0	**USILMOD [2:0]**	Interface mode configuration	USILMOD[2:0] I/F mode 0x7 LCD Parallel 0x6 LCD SPI 0x5 I²C slave 0x4 I²C master 0x3 SPI slave 0x2 SPI master 0x1 UART 0x0 Software reset	0x0	R/W	
USIL Transmit Data Buffer Register (USIL_TD)	**0x300601** (8 bits)	D7–0	**TD[7:0]**	Transmit data buffer TD7 = MSB TD0 = LSB	0x0 to 0xff	0x0	R/W	
USIL Receive Data Buffer Register (USIL_RD)	**0x300602** (8 bits)	D7–0	**RD[7:0]**	Receive data buffer RD7 = MSB RD0 = LSB	0x0 to 0xff	0x0	R	

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USIL UART Mode Configuration Register (USIL_UCFG)	**0x300640** (8 bits)	D7–4	**–**	reserved	–				–	–	0 when being read.
		D3	**UCHLN**	Character length select	1	8 bits	0	7 bits	0	R/W	
		D2	**USTPB**	Stop bit select	1	2 bits	0	1 bit	0	R/W	
		D1	**UPMD**	Parity mode select	1	Even	0	Odd	0	R/W	
		D0	**UPREN**	Parity enable	1	With parity	0	No parity	0	R/W	
USIL UART Mode Interrupt Enable Register (USIL_UIE)	**0x300641** (8 bits)	D7–3	**–**	reserved	–				–	–	0 when being read.
		D2	**UEIE**	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	**URDIE**	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	**UTDIE**	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	
USIL UART Mode Interrupt Flag Register (USIL_UIF)	**0x300642** (8 bits)	D7	**–**	reserved	–				–	–	0 when being read.
		D6	**URBSY**	Receive busy flag	1	Busy	0	Idle	0	R	
		D5	**UTBSY**	Transmit busy flag	1	Busy	0	Idle	0	R	
		D4	**UPEIF**	Parity error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D3	**USEIF**	Framing error flag	1	Error	0	Normal	0	R/W	
		D2	**UOEIF**	Overrun error flag	1	Error	0	Normal	0	R/W	
		D1	**URDIF**	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	**UTDIF**	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	
USIL SPI Master/Slave Mode Configuration Register (USIL_SCFG)	**0x300650** (8 bits)	D7–4	**–**	reserved	–				–	–	0 when being read.
		D3	**SCPHA**	Clock phase select	1	Phase 1	0	Phase 0	0	R/W	
		D2	**SCPOL**	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	**–**	reserved	–				–	–	Do not set to 1.
		D0	**SFSTMOD**	Fast mode select	1	Fast	0	Normal	0	R/W	
USIL SPI Master/Slave Mode Interrupt Enable Register (USIL_SIE)	**0x300651** (8 bits)	D7–3	**–**	reserved	–				–	–	0 when being read.
		D2	**SEIE**	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	**SRDIE**	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	**STDIE**	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	
USIL SPI Master/Slave Mode Interrupt Flag Register (USIL_SIF)	**0x300652** (8 bits)	D7–4	**–**	reserved	–				–	–	0 when being read.
		D3	**SSIF**	Transfer busy flag (master)	1	Busy	0	Idle	0	R	
				ss signal low flag (slave)	1	ss = H	0	ss = L			
		D2	**SEIF**	Overrun error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D1	**SRDIF**	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	**STDIF**	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	
USIL I²C Master Mode Trigger Register (USIL_IMTG)	**0x300660** (8 bits)	D7–5	**–**	reserved	–				–	–	0 when being read.
		D4	**IMTG**	I²C master operation trigger	1	Trigger	0	Ignored	0	W	
					1	Waiting	0	Finished		R	
		D3	**–**	reserved	–				–	–	0 when being read.
		D2–0	**IMTGMOD [2:0]**	I²C master trigger mode select	IMTGMOD[2:0]	Trigger mode			0x0	R/W	
					0x7	reserved					
					0x6	Receive ACK/NAK					
					0x5	Transmit NAK					
					0x4	Transmit ACK					
					0x3	Receive data					
					0x2	Transmit data					
					0x1	Stop condition					
					0x0	Start condition					
USIL I²C Master Mode Interrupt Enable Register (USIL_IMIE)	**0x300661** (8 bits)	D7–2	**–**	reserved	–				–	–	0 when being read.
		D1	**IMEIE**	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	**IMIE**	Operation completion int. enable	1	Enable	0	Disable	0	R/W	
USIL I²C Master Mode Interrupt Flag Register (USIL_IMIF)	**0x300662** (8 bits)	D7–6	**–**	reserved	–				–	–	0 when being read.
		D5	**IMBSY**	I²C master busy flag	1	Busy	0	Standby	0	R	
		D4–2	**IMSTA[2:0]**	I²C master status	IMSTA[2:0]	Status			0x0	R	
					0x7	reserved					
					0x6	NAK received					
					0x5	ACK received					
					0x4	ACK/NAK sent					
					0x3	End of Rx data					
					0x2	End of Tx data					
					0x1	Stop generated					
					0x0	Start generated					
		D1	**IMEIF**	Overrun error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D0	**IMIF**	Operation completion flag	1	Completed	0	Not completed	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USIL I²C Slave Mode Trigger Register (USIL_ISTG)	**0x300670** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**ISTG**	I²C slave operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**ISTGMOD [2:0]**	I²C slave trigger mode select	ISTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data/ Detect stop 0x2 Transmit data 0x1 reserved 0x0 Wait for start	0x0	R/W	
USIL I²C Slave Mode Interrupt Enable Register (USIL_ISIE)	**0x300671** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**ISEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**ISIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USIL I²C Slave Mode Interrupt Flag Register (USIL_ISIF)	**0x300672** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**ISBSY**	I²C slave busy flag	1 Busy 0 Standby	0	R	
		D4–2	**ISSTA[2:0]**	I²C slave status	ISSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop detected 0x0 Start detected	0x0	R	
		D1	**ISEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**ISIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	
USIL LCD SPI Mode Configuration Register (USIL_LSCFG)	**0x300680** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**LSCPHA**	Clock phase select	1 Phase 1 0 Phase 0	0	R/W	
		D2	**LSCPOL**	Clock polarity select	1 Active L 0 Active H	0	R/W	
		D1	**LSCMD**	Command bit	1 High 0 Low	0	R/W	
		D0	**LSCMDEN**	Command bit enable	1 Enable 0 Disable	0	R/W	
USIL LCD SPI Mode Interrupt Enable Register (USIL_LSIE)	**0x300681** (8 bits)	D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**LSTDIE**	Transmit buffer empty int. enable	1 Enable 0 Disable	0	R/W	
USIL LCD SPI Mode Interrupt Flag Register (USIL_LSIF)	**0x300682** (8 bits)	D7–2	**–**	reserved	–	–	–	X when being read.
		D1	**LSBSY**	Transfer busy flag	1 Busy 0 Idle	0	R	
		D0	**LSTDIF**	Transmit buffer empty flag	1 Empty 0 Not empty	0	R/W	Reset by writing 1.
USIL LCD SPI Mode Data Configuration Register (USIL_ LSDCFG)	**0x30068f** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3–2	**LS18DFM [1:0]**	LCD SPI 18-bit data format select	LS18DFM[1:0] Data format 0x3 Format 3 0x2 Format 2 0x1 Format 1 0x0 Format 0	0x0	R/W	
		D1–0	**LSDMOD [1:0]**	LCD SPI data mode select	LSDMOD[1:0] Data mode 0x3 24-bit mode 0x2 18-bit mode 0x1 16-bit mode 0x0 8-bit mode	0x0	R/W	
USIL LCD Parallel I/F Mode Configuration Register (USIL_LPCFG)	**0x300690** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2	**LPSRDEN**	Successive read enable	1 Enable 0 Disable	0	R/W	
		D1	**LPCMD**	Command bit	1 High 0 Low	0	R/W	
		D0	**LPRD**	Read trigger	1 Trigger 0 Ignored	0	W	
					1 Read cycle 0 Read finished		R	
USIL LCD Parallel I/F Mode Interrupt Enable Register (USIL_LPIE)	**0x300691** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**LPRDIE**	Read buffer full interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**LPWRIE**	Write buffer empty interrupt enable	1 Enable 0 Disable	0	R/W	
USIL LCD Parallel I/F Mode Interrupt Flag Register (USIL_LPIF)	**0x300692** (8 bits)	D7–3	**–**	reserved	–	–	–	X when being read.
		D2	**LPBSY**	Transfer busy flag	1 Busy 0 Idle	0	R	
		D1	**LPRDIF**	Read buffer full flag	1 Full 0 Not full	0	R/W	Reset by writing 1.
		D0	**LPWRIF**	Write buffer empty flag	1 Empty 0 Not empty	0	R/W	

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
USIL LCD Parallel I/F Mode Access Timing Register (USIL_LPAC)	**0x30069f** (8 bits)	D7–6	**LPHD[1:0]**	Hold cycle	LPHD[1:0]	Hold cycle	0x0	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D5–4	**LPST[1:0]**	Setup cycle	LPST[1:0]	Setup cycle	0x0	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D3–0	**LPWT[3:0]**	Wait cycle	LPWT[3:0]	Wait cycle	0x0	R/W	
					0xf	15 cycles			
					0xe	14 cycles			
					:	:			
					0x1	1 cycle			
					0x0	0 cycles			

0x300700–0x300772 USI Ch.2

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
USI Ch.2 Global Configuration Register (USI_GCFG2)	**0x300700** (8 bits)	D7–4	**–**	reserved	–				–	–	0 when being read.
		D3	**LSBFST**	MSB/LSB first mode select	1	MSB first	0	LSB first	0	R/W	
		D2–0	**USIMOD [2:0]**	Interface mode configuration	USIMOD[2:0]	I/F mode			0x0	R/W	
					0x7–0x6	reserved					
					0x5	I²C slave					
					0x4	I²C master					
					0x3	SPI slave					
					0x2	SPI master					
					0x1	UART					
					0x0	Software reset					
USI Ch.2 Transmit Data Buffer Register (USI_TD2)	**0x300701** (8 bits)	D7–0	**TD[7:0]**	Transmit data buffer TD7 = MSB TD0 = LSB	0x0 to 0xff				0x0	R/W	
USI Ch.2 Receive Data Buffer Register (USI_RD2)	**0x300702** (8 bits)	D7–0	**RD[7:0]**	Receive data buffer RD7 = MSB RD0 = LSB	0x0 to 0xff				0x0	R	
USI Ch.2 UART Mode Configuration Register (USI_UCFG2)	**0x300740** (8 bits)	D7–4	**–**	reserved	–				–	–	0 when being read.
		D3	**UCHLN**	Character length select	1	8 bits	0	7 bits	0	R/W	
		D2	**USTPB**	Stop bit select	1	2 bits	0	1 bit	0	R/W	
		D1	**UPMD**	Parity mode select	1	Even	0	Odd	0	R/W	
		D0	**UPREN**	Parity enable	1	With parity	0	No parity	0	R/W	
USI Ch.2 UART Mode Interrupt Enable Register (USI_UIE2)	**0x300741** (8 bits)	D7–3	**–**	reserved	–				–	–	0 when being read.
		D2	**UEIE**	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	**URDIE**	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	**UTDIE**	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	
USI Ch.2 UART Mode Interrupt Flag Register (USI_UIF2)	**0x300742** (8 bits)	D7	**–**	reserved	–				–	–	0 when being read.
		D6	**URBSY**	Receive busy flag	1	Busy	0	Idle	0	R	
		D5	**UTBSY**	Transmit busy flag	1	Busy	0	Idle	0	R	
		D4	**UPEIF**	Parity error flag	1	Error	0	Normal	0	R/W	Reset by writing 1.
		D3	**USEIF**	Framing error flag	1	Error	0	Normal	0	R/W	
		D2	**UOEIF**	Overrun error flag	1	Error	0	Normal	0	R/W	
		D1	**URDIF**	Receive buffer full flag	1	Full	0	Not full	0	R/W	
		D0	**UTDIF**	Transmit buffer empty flag	1	Empty	0	Not empty	0	R/W	
USI Ch.2 SPI Master/Slave Mode Configuration Register (USI_SCFG2)	**0x300750** (8 bits)	D7–6	**–**	reserved	–				–	–	0 when being read.
		D5	**SCMD**	Command bit (for 9-bit data)	1	High	0	Low	0	R/W	
		D4	**SCHLN**	Character length select	1	9 bits	0	8 bits	0	R/W	
		D3	**SCPHA**	Clock phase select	1	Phase 1	0	Phase 0	0	R/W	
		D2	**SCPOL**	Clock polarity select	1	Active L	0	Active H	0	R/W	
		D1	**–**	reserved	–				–	–	Do not set to 1.
		D0	**SFSTMOD**	Fast mode select	1	Fast	0	Normal	0	R/W	
USI Ch.2 SPI Master/Slave Mode Interrupt Enable Register (USI_SIE2)	**0x300751** (8 bits)	D7–3	**–**	reserved	–				–	–	0 when being read.
		D2	**SEIE**	Receive error interrupt enable	1	Enable	0	Disable	0	R/W	
		D1	**SRDIE**	Receive buffer full interrupt enable	1	Enable	0	Disable	0	R/W	
		D0	**STDIE**	Transmit buffer empty int. enable	1	Enable	0	Disable	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
USI Ch.2 SPI Master/Slave Mode Interrupt Flag Register (USI_SIF2)	**0x300752** (8 bits)	D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**SSIF**	Transfer busy flag (master)	1 Busy 0 Idle	0	R	
				ss signal low flag (slave)	1 ss = H 0 ss = L			
		D2	**SEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D1	**SRDIF**	Receive buffer full flag	1 Full 0 Not full	0	R/W	
		D0	**STDIF**	Transmit buffer empty flag	1 Empty 0 Not empty	0	R/W	
USI Ch.2 I²C Master Mode Trigger Register (USI_IMTG2)	**0x300760** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**IMTG**	I²C master operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**IMTGMOD [2:0]**	I²C master trigger mode select	IMTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data 0x2 Transmit data 0x1 Stop condition 0x0 Start condition	0x0	R/W	
USI Ch.2 I²C Master Mode Interrupt Enable Register (USI_IMIE2)	**0x300761** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**IMEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**IMIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.2 I²C Master Mode Interrupt Flag Register (USI_IMIF2)	**0x300762** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**IMBSY**	I²C master busy flag	1 Busy 0 Standby	0	R	
		D4–2	**IMSTA[2:0]**	I²C master status	IMSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop generated 0x0 Start generated	0x0	R	
		D1	**IMEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**IMIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	
USI Ch.2 I²C Slave Mode Trigger Register (USI_ISTG2)	**0x300770** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**ISTG**	I²C slave operation trigger	1 Trigger 0 Ignored	0	W	
					1 Waiting 0 Finished		R	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**ISTGMOD [2:0]**	I²C slave trigger mode select	ISTGMOD[2:0] Trigger mode 0x7 reserved 0x6 Receive ACK/NAK 0x5 Transmit NAK 0x4 Transmit ACK 0x3 Receive data/ Detect stop 0x2 Transmit data 0x1 reserved 0x0 Wait for start	0x0	R/W	
USI Ch.2 I²C Slave Mode Interrupt Enable Register (USI_ISIE2)	**0x300771** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**ISEIE**	Receive error interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**ISIE**	Operation completion int. enable	1 Enable 0 Disable	0	R/W	
USI Ch.2 I²C Slave Mode Interrupt Flag Register (USI_ISIF2)	**0x300772** (8 bits)	D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**ISBSY**	I²C slave busy flag	1 Busy 0 Standby	0	R	
		D4–2	**ISSTA[2:0]**	I²C slave status	ISSTA[2:0] Status 0x7 reserved 0x6 NAK received 0x5 ACK received 0x4 ACK/NAK sent 0x3 End of Rx data 0x2 End of Tx data 0x1 Stop detected 0x0 Start detected	0x0	R	
		D1	**ISEIF**	Overrun error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D0	**ISIF**	Operation completion flag	1 Completed 0 Not completed	0	R/W	

Host 0x0–0xf — Host Interface (HIF) for 8-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Byte Data Register (HIF_DRIF_B)	**Host 0x0** (8 bits)	D7–0	**B_DAT[7:0]**	Byte access data	0x0–0xff	X	R/W	
HIF Direct Interface Half Word Data Register (HIF_DRIF_HW)	**Host 0x1** (8 bits)	D7–0	**HW_DAT [7:0]**	Half word access data	0x0–0xff	X	R/W	
HIF Direct Interface Word Data Register (HIF_DRIF_W)	**Host 0x2** (8 bits)	D7–0	**W_DAT[7:0]**	Word access data	0x0–0xff	X	R/W	
HIF Indirect Interface Data Register (HIF_IDIF_DAT)	**Host 0x3** (8 bits)	D7–0	**IDIF_DAT [7:0]**	Indirect interface data	0x0–0xff	X	R/W	
HIF Indirect Interface Control Register (HIF_IDIF_ CTRL)	**Host 0x4** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	**WR_BF**	Write buffer full flag	1 Full / 0 Empty	0	R	
		D3	**BUSY**	Bus controller busy flag	1 Busy / 0 Ready	0	R	
		D2–1	**ACCESS _MD[1:0]**	Access mode select	ACCESS_MD[1:0]: 0x3–0x2 Word; 0x1 Half word; 0x0 Byte	0x0	R/W	
		D0	**RW_SEL**	Read/write cycle select	1 Write / 0 Read	0	R/W	
HIF Control Register (HIF_CTRL)	**Host 0x6** (8 bits)	D7	**HIFSEL**	HIF data width select	1 16 bits / 0 8 bits	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	**INTSEL**	Interrupt source select	1 ITC_INT / 0 SINT	0	R/W	
		D3–2	**ADDR_MD [1:0]**	Address control mode select	ADDR_MD[1:0]: 0x3–0x2 reserved; 0x1 Post increment; 0x0 Fixed	0x0	R/W	
		D1	**ENDIAN**	Endian mode select	1 Big endian / 0 Little endian	0	R/W	
		D0	**AHB_EN**	AHB interface enable	1 Enable / 0 Disable	0	R/W	
HIF Address Register 0 (HIF_ADDR0)	**Host 0x8** (8 bits)	D7–0	**ADDR [7:0]**	Address A[7:0]	0x0–0xff	0x0	R/W	
HIF Address Register 1 (HIF_ADDR1)	**Host 0x9** (8 bits)	D7–0	**ADDR [15:8]**	Address A[15:8]	0x0–0xff	0x0	R/W	
HIF Address Register 2 (HIF_ADDR2)	**Host 0xa** (8 bits)	D7–0	**ADDR [23:16]**	Address A[23:16]	0x0–0xff	0x0	R/W	
HIF Address Register 3 (HIF_ADDR3)	**Host 0xb** (8 bits)	D7–0	**ADDR [31:24]**	Address A[31:24]	0x0–0xff	0x0	R/W	
HIF Special Register Index Register (HIF_SP_IDX)	**Host 0xc** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4–0	**INDEX[4:0]**	Special register index	0x1–0x13	0x0	R/W	
HIF Special Register Data Register (HIF_SP_DAT)	**Host 0xd** (8 bits)	D7–0	**SP_DAT[7:0]**	Special register data	0x0–0xff	0x0	R/W	
HIF Flag Register 0 (HIF_FLAG0)	**Host 0xe** (8 bits)	D7	**PROTECT**	HIF_FLAG write protect	1 Set/remove / 0 Disable	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	**SFLAG3**	Slave flag 3	1 Set / 0 Cleared	0	R	
					1 Reset / 0 Ignored	–	W	
		D3	**SFLAG2**	Slave flag 2	1 Set / 0 Cleared	0	R	
					1 Reset / 0 Ignored	–	W	
		D2	**SFLAG1**	Slave flag 1	1 Set / 0 Cleared	0	R	
					1 Reset / 0 Ignored	–	W	
		D1	**SFLAG0**	Slave flag 0	1 Set / 0 Cleared	0	R	
					1 Reset / 0 Ignored	–	W	
		D0	**SINT**	Slave interrupt flag	1 Set / 0 Cleared	0	R	
					1 Reset / 0 Ignored	–	W	
HIF Flag Register 1 (HIF_FLAG1)	**Host 0xf** (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	**HFLAG2**	Host flag 2	1 Set / 0 Clear	0	R/W	
		D2	**HFLAG1**	Host flag 1	1 Set / 0 Clear	0	R/W	
		D1	**HFLAG0**	Host flag 0	1 Set / 0 Clear	0	R/W	
		D0	**HINT**	Host interrupt flag	1 Set / 0 Clear	0	R/W	

Host 0x0–0xf — Host Interface (HIF) for 16-bit interface mode

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Direct Interface Byte Data Register (HIF_DRIF_B)	**Host 0x0** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	**B_DAT[7:0]**	Byte access data	0x0–0xff	X	R/W	
HIF Direct Interface Half Word Data Register (HIF_DRIF_HW)	**Host 0x1** (16 bits)	D15–0	**HW_DAT [15:0]**	Half word access data	0x0–0xffff	X	R/W	
HIF Direct Interface Word Data Register (HIF_DRIF_W)	**Host 0x2** (16 bits)	D15–0	**W_DAT [15:0]**	Word access data	0x0–0xffff	X	R/W	
HIF Indirect Interface Data Register (HIF_IDIF_DAT)	**Host 0x3** (16 bits)	D15–0	**IDIF_DAT [15:0]**	Indirect interface data	0x0–0xffff	X	R/W	
HIF Indirect Interface Control Register (HIF_IDIF_CTRL)	**Host 0x4** (16 bits)	D15–5	–	reserved	–	–	–	0 when being read.
		D4	**WR_BF**	Write buffer full flag	1 Full 0 Empty	0	R	
		D3	**BUSY**	Bus controller busy flag	1 Busy 0 Ready	0	R	
		D2–1	**ACCESS_MD[1:0]**	Access mode select	ACCESS_MD[1:0] Mode 0x3–0x2 Word 0x1 Half word 0x0 Byte	0x0	R/W	
		D0	**RW_SEL**	Read/write cycle select	1 Write 0 Read	0	R/W	
HIF Control Register (HIF_CTRL)	**Host 0x6** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7	**HIFSEL**	HIF data width select	1 16 bits 0 8 bits	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	**INTSEL**	Interrupt source select	1 ITC_INT 0 SINT	0	R/W	
		D3–2	**ADDR_MD [1:0]**	Address control mode select	ADDR_MD[1:0] Mode 0x3–0x2 reserved 0x1 Post increment 0x0 Fixed	0x0	R/W	
		D1	**ENDIAN**	Endian mode select	1 Big endian 0 Little endian	0	R/W	
		D0	**AHB_EN**	AHB interface enable	1 Enable 0 Disable	0	R/W	
HIF Address Register 0 (HIF_ADDR0)	**Host 0x8** (16 bits)	D15–0	**ADDR [15:0]**	Address A[15:0]	0x0–0xffff	0x0	R/W	
HIF Address Register 1 (HIF_ADDR1)	**Host 0x9** (16 bits)	D15–0	**ADDR [31:16]**	Address A[31:16]	0x0–0xffff	0x0	R/W	
HIF Special Register Index Register (HIF_SP_IDX)	**Host 0xc** (16 bits)	D15–5	–	reserved	–	–	–	0 when being read.
		D4–0	**INDEX[4:0]**	Special register index	0x1–0x13	0x0	R/W	
HIF Special Register Data Register (HIF_SP_DAT)	**Host 0xd** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	**SP_DAT[7:0]**	Special register data	0x0–0xff	0x0	R/W	
HIF Flag Register 0 (HIF_FLAG0)	**Host 0xe** (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7	**PROTECT**	HIF_FLAG write protect	1 Set/remove 0 Disable	0	R/W	
		D6–5	–	reserved	–	–	–	0 when being read.
		D4	**SFLAG3**	Slave flag 3	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D3	**SFLAG2**	Slave flag 2	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D2	**SFLAG1**	Slave flag 1	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D1	**SFLAG0**	Slave flag 0	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D0	**SINT**	Slave interrupt flag	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
HIF Flag Register 1 (HIF_FLAG1)	**Host 0xf** (16 bits)	D15–4	–	reserved	–	–	–	0 when being read.
		D3	**HFLAG2**	Host flag 2	1 Set 0 Clear	0	R/W	
		D2	**HFLAG1**	Host flag 1	1 Set 0 Clear	0	R/W	
		D1	**HFLAG0**	Host flag 0	1 Set 0 Clear	0	R/W	
		D0	**HINT**	Host interrupt flag	1 Set 0 Clear	0	R/W	

Host 0xd [Index 0x1–0x13] — Host Interface (HIF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Extended Control Register 1 (HIF_EXTCTRL1)	Host 0xd [0x1] (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	HINT_MD	Host interrupt mode select	1 INT 0 NMI	0	R/W	
		D0	–	reserved	–	–	–	0 when being read.
HIF Semaphore Register 0 (HIF_SMPH0)	Host 0xd [0x8] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH0	Semaphore 0	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 1 (HIF_SMPH1)	Host 0xd [0x9] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH1	Semaphore 1	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 2 (HIF_SMPH2)	Host 0xd [0xa] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH2	Semaphore 2	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 3 (HIF_SMPH3)	Host 0xd [0xb] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH3	Semaphore 3	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 4 (HIF_SMPH4)	Host 0xd [0xc] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH4	Semaphore 4	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 5 (HIF_SMPH5)	Host 0xd [0xd] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH5	Semaphore 5	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 6 (HIF_SMPH6)	Host 0xd [0xe] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH6	Semaphore 6	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 7 (HIF_SMPH7)	Host 0xd [0xf] (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH7	Semaphore 7	1 Busy 0 Not busy	0	R/W	
HIF Scratchpad Register 0 (HIF_SCRATCH0)	Host 0xd [0x10] (8 bits)	D7–0	SCR0[7:0]	Scratch data	0x0–0xff	0x0	R/W	
HIF Scratchpad Register 1 (HIF_SCRATCH1)	Host 0xd [0x11] (8 bits)	D7–0	SCR1[7:0]	Scratch data	0x0–0xff	0x0	R/W	
HIF Scratchpad Register 2 (HIF_SCRATCH2)	Host 0xd [0x12] (8 bits)	D7–0	SCR2[7:0]	Scratch data	0x0–0xff	0x0	R/W	
HIF Scratchpad Register 3 (HIF_SCRATCH3)	Host 0xd [0x13] (8 bits)	D7–0	SCR3[7:0]	Scratch data	0x0–0xff	0x0	R/W	

0x300901–0x300919 — Host Interface (HIF)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Extended Control Register 1 (HIF_EXTCTRL1)	0x300901 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	HINT_MD	Host interrupt mode select	1 INT 0 NMI	0	R/W	
		D0	–	reserved	–	–	–	0 when being read.
HIF Semaphore Register 0 (HIF_SMPH0)	0x300908 (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH0	Semaphore 0	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 1 (HIF_SMPH1)	0x300909 (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH1	Semaphore 1	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 2 (HIF_SMPH2)	0x30090a (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH2	Semaphore 2	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 3 (HIF_SMPH3)	0x30090b (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH3	Semaphore 3	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 4 (HIF_SMPH4)	0x30090c (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH4	Semaphore 4	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 5 (HIF_SMPH5)	0x30090d (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH5	Semaphore 5	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 6 (HIF_SMPH6)	0x30090e (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH6	Semaphore 6	1 Busy 0 Not busy	0	R/W	
HIF Semaphore Register 7 (HIF_SMPH7)	0x30090f (8 bits)	D7–1	–	reserved	–	–	–	0 when being read.
		D0	SMPH7	Semaphore 7	1 Busy 0 Not busy	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
HIF Scratchpad Register 0 (HIF_SCRATCH0)	**0x300910** (8 bits)	D7–0	**SCR0[7:0]**	Scratch data	0x0–0xff	0x0	R/W	
HIF Scratchpad Register 1 (HIF_SCRATCH1)	**0x300911** (8 bits)	D7–0	**SCR1[7:0]**	Scratch data	0x0–0xff	0x0	R/W	
HIF Scratchpad Register 2 (HIF_SCRATCH2)	**0x300912** (8 bits)	D7–0	**SCR2[7:0]**	Scratch data	0x0–0xff	0x0	R/W	
HIF Scratchpad Register 3 (HIF_SCRATCH3)	**0x300913** (8 bits)	D7–0	**SCR3[7:0]**	Scratch data	0x0–0xff	0x0	R/W	
HIF Flag Register 0 (HIF_FLAG0)	**0x300918** (8 bits)	D7–5	–	reserved	–	–	–	0 when being read.
		D4	**SFLAG3**	Slave flag 3	1 Set 0 Clear	0	R/W	
		D3	**SFLAG2**	Slave flag 2	1 Set 0 Clear	0	R/W	
		D2	**SFLAG1**	Slave flag 1	1 Set 0 Clear	0	R/W	
		D1	**SFLAG0**	Slave flag 0	1 Set 0 Clear	0	R/W	
		D0	**SINT**	Slave interrupt flag	1 Set 0 Clear	0	R/W	
HIF Flag Register 1 (HIF_FLAG1)	**0x300919** (8 bits)	D7–4	–	reserved	–	–	–	0 when being read.
		D3	**HFLAG2**	Host flag 2	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D2	**HFLAG1**	Host flag 1	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D1	**HFLAG0**	Host flag 0	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	
		D0	**HINT**	Host interrupt flag	1 Set 0 Cleared	0	R	
					1 Reset 0 Ignored	–	W	

0x300a00–0x300a0f Real-time Clock (RTC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Interrupt Status Register (RTC_INTSTAT)	**0x300a00** (8 bits)	D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**RTCIRQ**	Interrupt status	1 Occurred 0 Not occurred	X (0)	R/W	Reset by writing 1.
RTC Interrupt Mode Register (RTC_INTMODE)	**0x300a01** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4–2	**RTCT[2:0]**	RTC interrupt cycle setup	RTCT[2:0] Cycle 0x7 reserved 0x6 1/128 second 0x5 1/256 second 0x4 1/512 second 0x3 1 hour 0x2 1 minute 0x1 1 second 0x0 1/64 second	X (0x1)	R/W	
		D1	**RTCIMD**	reserved	1	X (1)	R/W	Always set to 1.
		D0	**RTCIEN**	RTC interrupt enable	1 Enable 0 Disable	X (0)	R/W	
RTC Control 0 Register (RTC_CNTL0)	**0x300a02** (8 bits)	D7–5	**–**	reserved	–	–	–	0 when being read.
		D4	**RTC24H**	24H/12H mode select	1 24H 0 12H	X (0)	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2	**RTCADJ**	30-second adjustment	1 Adjust 0 –	X (0)	R/W	
		D1	**RTCSTP**	Divider run/stop control	1 Stop 0 Run	X (0)	R/W	
		D0	**RTCRST**	Software reset	1 Reset 0 –	X (0)	R/W	
RTC Control 1 Register (RTC_CNTL1)	**0x300a03** (8 bits)	D7–3	**–**	reserved	–	–	–	0 when being read.
		D2	**RTCRDHLD**	Read buffer enable	1 Enable 0 Disable	X (0)	R/W	
		D1	**RTCBSY**	Counter busy flag	1 Busy 0 R/W possible	X (0)	R	
		D0	**RTCHLD**	Counter hold control	1 Hold 0 Running	X (0)	R/W	
RTC Second Register (RTC_SEC)	**0x300a04** (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)	**0x300a05** (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)	**0x300a06** (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 to 9	X (*)	R/W	
RTC Day Register (RTC_DAY)	**0x300a07** (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	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
RTC Month Register (RTC_MONTH)	**0x300a08** (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)	**0x300a09** (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)	**0x300a0a** (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	
RTC Wakeup Configuration Register (RTC_WAKEUP)	**0x300a0f** (8 bits)	D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**WUP_CTL**	WAKEUP control	1 Active 0 Inactive	X (0)	R/W	
		D0	**WUP_POL**	WAKEUP polarity select	1 Active low 0 Active high	X (0)	R/W	

Init.: ( ) indicates the value set after a software reset (RTCRST → 1 → 0) is performed.

* Software reset (RTCRST → 1 → 0) does not affect the counter values. This register retains the value set before a software reset is performed.

0x300b00–0x300b0f BBRAM

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
BBRAM byte 0 –byte 15 (BBRAM_0 –BBRAM_15)	**0x300b00** | **0x300b0f** (8 bits)	D7–0	**–**	BBRAM byte data	0x0 to 0xff	X	R/W	

0x300c00–0x300c9f USB Function Controller (USB)

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
MainIntStat (Main interrupt status)	**0x300c00** (8 bits)	D7	**SIE_IntStat**	1 SIE interrupts 0 None	0	R	
		D6	**EPrIntStat**	1 EPr interrupts 0 None	0	R	
		D5	**DMA_IntStat**	1 DMA interrupts 0 None	0	R	
		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 (SIE interrupt status)	**0x300c01** (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)	
EPrIntStat (EPr interrupt status)	**0x300c02** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**EPdIntStat**	1 EPd interrupt 0 None	0	R	
		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	
DMA_IntStat (DMA interrupt status)	**0x300c03** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1	**DMA_CountUp**	1 DMA counter overflow 0 None	0	R(W)	
		D0	**DMA_Cmp**	1 DMA complete 0 None	0	R(W)	
FIFO_IntStat (FIFO interrupt status)	**0x300c04** (8 bits)	D7	**DescriptorCmp**	1 Descriptor complete 0 None	0	R(W)	
		D6–2	**–**	–	–	–	0 when being read.
		D1	**FIFO_IN_Cmp**	1 IN FIFO Complete 0 None	0	R(W)	
		D0	**FIFO_OUT_Cmp**	1 OUT FIFO complete 0 None	0	R(W)	
EP0IntStat (EP0 interrupt status)	**0x300c07** (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)	

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPaIntStat	**0x300c08**	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	**0x300c09**	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	**0x300c0a**	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	**0x300c0b**	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)	
MainIntEnb	**0x300c10**	D7	**EnSIE_IntStat**	1	Enable	0	Disable	0	R/W	
(Main interrupt	(8 bits)	D6	**EnEPrIntStat**					0	R/W	
enable)		D5	**EnDMA_IntStat**					0	R/W	
		D4	**EnFIFO_IntStat**					0	R/W	
		D3–2	**–**	–				–	–	0 when being read.
		D1	**EnEP0IntStat**	1	Enable	0	Disable	0	R/W	
		D0	**EnRcvEP0SETUP**					0	R/W	
SIE_IntEnb	**0x300c11**	D7	**EnVBUS_Changed**	1	Enable	0	Disable	0	R/W	
(SIE interrupt	(8 bits)	D6	**EnNonJ**					0	R/W	
enable)		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	Enable	0	Disable	0	R/W	
EPrIntEnb	**0x300c12**	D7–4	**–**	–				–	–	0 when being read.
(EPr interrupt	(8 bits)	D3	**EnEPdIntStat**	1	Enable	0	Disable	0	R/W	
enable)		D2	**EnEPcIntStat**					0	R/W	
		D1	**EnEPbIntStat**					0	R/W	
		D0	**EnEPaIntStat**					0	R/W	
DMA_IntEnb	**0x300c13**	D7–2	**–**	–				–	–	0 when being read.
(DMA interrupt	(8 bits)	D1	**EnDMA_CountUp**	1	Enable	0	Disable	0	R/W	
enable)		D0	**EnDMA_Cmp**					0	R/W	
FIFO_IntEnb	**0x300c14**	D7	**EnDescriptorCmp**	1	Enable	0	Disable	0	R/W	
(FIFO interrupt	(8 bits)	D6–2	**–**	–				–	–	0 when being read.
enable)		D1	**EnFIFO_IN_Cmp**	1	Enable	0	Disable	0	R/W	
		D0	**EnFIFO_OUT_Cmp**					0	R/W	
EP0IntEnb	**0x300c17**	D7–6	**–**	–				–	–	0 when being read.
(EP0 interrupt	(8 bits)	D5	**EnIN_TranACK**	1	Enable	0	Disable	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	

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPaIntEnb (EPa interrupt enable)	**0x300c18** (8 bits)	D7	**–**	–	–	–	0 when being read.
		D6	**EnOUT_ShortACK**	1 Enable 0 Disable	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)	**0x300c19** (8 bits)	D7	**–**	–	–	–	0 when being read.
		D6	**EnOUT_ShortACK**	1 Enable 0 Disable	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)	**0x300c1a** (8 bits)	D7	**–**	–	–	–	0 when being read.
		D6	**EnOUT_ShortACK**	1 Enable 0 Disable	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)	**0x300c1b** (8 bits)	D7	**–**	–	–	–	0 when being read.
		D6	**EnOUT_ShortACK**	1 Enable 0 Disable	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	
RevisionNum (Revision number)	**0x300c20** (8 bits)	D7–0	**RevisionNum[7:0]**	Revision number (0x12)	0x12	R	
USB_Control (USB control)	**0x300c21** (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 Deactivate USB	0	R/W	
USB_Status (USB status)	**0x300c22** (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.
		D1–0	**LineState[1:0]**	LineState[1:0] / DP/DM: 0x3 SE1; 0x2 K; 0x1 J; 0x0 SE0	X	R	
XcvrControl (Xcvr control)	**0x300c23** (8 bits)	D7	**RpuEnb**	1 Enable pull-up 0 Disable pull-up	0	R/W	
		D6–2	**–**	–	–	–	0 when being read.
		D1–0	**OpMode[1:0]**	OpMode[1:0] / Operation mode: 0x3 reserved; 0x2 Disable bitstuffing and NRZI encoding; 0x1 Non-driving; 0x0 Normal operation	0x1	R/W	
USB_Test (USB test)	**0x300c24** (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)	**0x300c25** (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)	**0x300c26** (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	

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
FrameNumber_H (Frame number high)	0x300c2e (8 bits)	D7	FnInvalid	1 Invalid frame number 0 Valid frame number	1	R	
		D6–3	–	–	–	–	0 when being read.
		D2–0	FrameNumber[10:8]	Frame number high	0x0	R	
FrameNumber_L (Frame number low)	0x300c2f (8 bits)	D7–0	FrameNumber[7:0]	Frame number low	0x0	R	
EP0Setup_0 (EP0 setup 0)	0x300c30 (8 bits)	D7–0	EP0Setup_0[7:0]	Endpoint 0 setup data 0	0x0	R	
EP0Setup_1 (EP0 setup 1)	0x300c31 (8 bits)	D7–0	EP0Setup_1[7:0]	Endpoint 0 setup data 1	0x0	R	
EP0Setup_2 (EP0 setup 2)	0x300c32 (8 bits)	D7–0	EP0Setup_2[7:0]	Endpoint 0 setup data 2	0x0	R	
EP0Setup_3 (EP0 setup 3)	0x300c33 (8 bits)	D7–0	EP0Setup_3[7:0]	Endpoint 0 setup data 3	0x0	R	
EP0Setup_4 (EP0 setup 4)	0x300c34 (8 bits)	D7–0	EP0Setup_4[7:0]	Endpoint 0 setup data 4	0x0	R	
EP0Setup_5 (EP0 setup 5)	0x300c35 (8 bits)	D7–0	EP0Setup_5[7:0]	Endpoint 0 setup data 5	0x0	R	
EP0Setup_6 (EP0 setup 6)	0x300c36 (8 bits)	D7–0	EP0Setup_6[7:0]	Endpoint 0 setup data 6	0x0	R	
EP0Setup_7 (EP0 setup 7)	0x300c37 (8 bits)	D7–0	EP0Setup_7[7:0]	Endpoint 0 setup data 7	0x0	R	
USB_Address (USB address)	0x300c38 (8 bits)	D7	AutoSetAddress	1 Auto set address 0 Do nothing	0	R/W	
		D6–0	USB_Address[6:0]	USB address	0x0	R/W	
EP0Control (EP0 control)	0x300c39 (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	
EP0ControlIN (EP0 control IN)	0x300c3a (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	
EP0ControlOUT (EP0 control OUT)	0x300c3b (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	
EP0MaxSize (EP0 max packet size)	0x300c3f (8 bits)	D7	–	–	–	–	0 when being read.
		D6–3	EP0MaxSize[6:3]	Endpoint EP0 max packet size	0x1	R/W	
		D2–0	–	–	–	–	0 when being read.
EPaControl (EPa control)	0x300c40 (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	
EPbControl (EPb control)	0x300c41 (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	
EPcControl (EPc control)	0x300c42 (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	

Register name	Address	Bit	Name	Setting				Init.	R/W	Remarks
EPdControl (EPd control)	**0x300c43** (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	
EPaMaxSize_H (EPa max packet size high)	**0x300c50** (8 bits)	D7–2	**–**	–				–	–	0 when being read.
		D1–0	**EPaMaxSize[9:8]**	Endpoint EPa max packet size				0x0	R/W	
EPaMaxSize_L (EPa max packet size low)	**0x300c51** (8 bits)	D7–0	**EPaMaxSize[7:0]**	Endpoint EPa max packet size				0x0	R/W	
EPaConfig_0 (EPa configuration 0)	**0x300c52** (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.
		D3–0	**EndPointNumber [3:0]**	Endpoint number (0x1 to 0xf)				0x0	R/W	
EPaConfig_1 (EPa configuration 1)	**0x300c53** (8 bits)	D7	**ISO**	1	ISO	0	Non-ISO	0	R/W	
		D6	**ISO_CRCmode**	1	CRC mode	0	Normal ISO	0	R/W	
		D5–0	**–**	–				–	–	0 when being read.
EPbMaxSize_H (EPb max packet size high)	**0x300c54** (8 bits)	D7–2	**–**	–				–	–	0 when being read.
		D1–0	**EPbMaxSize[9:8]**	Endpoint EPb max packet size				0x0	R/W	
EPbMaxSize_L (EPb max packet size low)	**0x300c55** (8 bits)	D7–0	**EPbMaxSize[7:0]**	Endpoint EPb max packet size				0x0	R/W	
EPbConfig_0 (EPb configuration 0)	**0x300c56** (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.
		D3–0	**EndPointNumber [3:0]**	Endpoint number (0x1 to 0xf)				0x0	R/W	
EPbConfig_1 (EPb configuration 1)	**0x300c57** (8 bits)	D7	**ISO**	1	ISO	0	Non-ISO	0	R/W	
		D6	**ISO_CRCmode**	1	CRC mode	0	Normal ISO	0	R/W	
		D5–0	**–**	–				–	–	0 when being read.
EPcMaxSize_H (EPc max packet size high)	**0x300c58** (8 bits)	D7–2	**–**	–				–	–	0 when being read.
		D1–0	**EPcMaxSize[9:8]**	Endpoint EPc max packet size				0x0	R/W	
EPcMaxSize_L (EPc max packet size low)	**0x300c59** (8 bits)	D7–0	**EPcMaxSize[7:0]**	Endpoint EPc max packet size				0x0	R/W	
EPcConfig_0 (EPc configuration 0)	**0x300c5a** (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.
		D3–0	**EndPointNumber [3:0]**	Endpoint number (0x1 to 0xf)				0x0	R/W	
EPcConfig_1 (EPc configuration 1)	**0x300c5b** (8 bits)	D7	**ISO**	1	ISO	0	Non-ISO	0	R/W	
		D6	**ISO_CRCmode**	1	CRC mode	0	Normal ISO	0	R/W	
		D5–0	**–**	–				–	–	0 when being read.
EPdMaxSize_H (EPd max packet size high)	**0x300c5c** (8 bits)	D7–2	**–**	–				–	–	0 when being read.
		D1–0	**EPdMaxSize[9:8]**	Endpoint EPd max packet size				0x0	R/W	
EPdMaxSize_L (EPd max packet size low)	**0x300c5d** (8 bits)	D7–0	**EPdMaxSize[7:0]**	Endpoint EPd max packet size				0x0	R/W	
EPdConfig_0 (EPd configuration 0)	**0x300c5e** (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.
		D3–0	**EndPointNumber [3:0]**	Endpoint number (0x1 to 0xf)				0x0	R/W	

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
EPdConfig_1 (EPd configuration 1)	**0x300c5f** (8 bits)	D7	**ISO**	1 ISO 0 Non-ISO	0	R/W	
		D6	**ISO_CRCmode**	1 CRC mode 0 Normal ISO	0	R/W	
		D5–0	**–**	–	–	–	0 when being read.
EPaStartAdrs_H (EPa FIFO start address high)	**0x300c70** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPaStartAdrs[11:8]**	Endpoint EPa start address	0x0	R/W	
EPaStartAdrs_L (EPa FIFO start address low)	**0x300c71** (8 bits)	D7–2	**EPaStartAdrs[7:2]**	Endpoint EPa start address	0x0	R/W	
		D1–0	**–**	–	–	–	0 when being read.
EPbStartAdrs_H (EPb FIFO start address high)	**0x300c72** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPbStartAdrs[11:8]**	Endpoint EPb start address	0x0	R/W	
EPbStartAdrs_L (EPb FIFO start address low)	**0x300c73** (8 bits)	D7–2	**EPbStartAdrs[7:2]**	Endpoint EPb start address	0x0	R/W	
		D1–0	**–**	–	–	–	0 when being read.
EPcStartAdrs_H (EPc FIFO start address high)	**0x300c74** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPcStartAdrs[11:8]**	Endpoint EPc start address	0x0	R/W	
EPcStartAdrs_L (EPc FIFO start address low)	**0x300c75** (8 bits)	D7–2	**EPcStartAdrs[7:2]**	Endpoint EPc start address	0x0	R/W	
		D1–0	**–**	–	–	–	0 when being read.
EPdStartAdrs_H (EPd FIFO start address high)	**0x300c76** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPdStartAdrs[11:8]**	Endpoint EPd start address	0x0	R/W	
EPdStartAdrs_L (EPd FIFO start address low)	**0x300c77** (8 bits)	D7–2	**EPdStartAdrs[7:2]**	Endpoint EPd start address	0x0	R/W	
		D1–0	**–**	–	–	–	0 when being read.
CPU_JoinRd (CPU join FIFO read)	**0x300c80** (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)	**0x300c81** (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 (EPn FIFO access enable)	**0x300c82** (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)	**0x300c83** (8 bits)	D7–0	**EPnFIFOData[7:0]**	Endpoint n FIFO access from CPU	X	R/W	
EPnRdRemain_H (EPn FIFO read remain high)	**0x300c84** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPnRdRemain[11:8]**	Endpoint n FIFO read remain	0x0	R	
EPnRdRemain_L (EPn FIFO read remain low)	**0x300c85** (8 bits)	D7–0	**EPnRdRemain[7:0]**	Endpoint n FIFO read remain	0x0	R	
EPnWrRemain_H (EPn FIFO write remain high)	**0x300c86** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**EPnWrRemain[11:8]**	Endpoint n FIFO write remain	0x0	R	
EPnWrRemain_L (EPn FIFO write remain low)	**0x300c87** (8 bits)	D7–0	**EPnWrRemain[7:0]**	Endpoint n FIFO write remain	0x0	R	
DescAdrs_H (Descriptor address high)	**0x300c88** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**DescAdrs[11:8]**	Descriptor address	0x0	R/W	
DescAdrs_L (Descriptor address low)	**0x300c89** (8 bits)	D7–0	**DescAdrs[7:0]**	Descriptor address	0x0	R/W	
DescSize_H (Descriptor size high)	**0x300c8a** (8 bits)	D7–2	**–**	–	–	–	0 when being read.
		D1–0	**DescSize[9:8]**	Descriptor size	0x0	R/W	
DescSize_L (Descriptor size low)	**0x300c8b** (8 bits)	D7–0	**DescSize[7:0]**	Descriptor size	0x0	R/W	

Register name	Address	Bit	Name	Setting	Init.	R/W	Remarks
DescDoor (Descriptor door)	**0x300c8f** (8 bits)	D7–0	**DescMode[7:0]**	Descriptor door	0x0	R/W	
DMA_FIFO_ Control (DMA FIFO control)	**0x300c90** (8 bits)	D7	**FIFO_Running**	1 FIFO is running 0 FIFO is not running	0	R	
		D6	**AutoEnShort**	1 Auto enable short packet 0 Do nothing	0	R/W	
		D5–0	**–**	–	–	–	0 when being read.
DMA_Join (DMA join FIFO)	**0x300c91** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3	**JoinEPdDMA**	1 Join EPd to DMA 0 Do nothing	0	R/W	
		D2	**JoinEPcDMA**	1 Join EPc to DMA 0 Do nothing	0	R/W	
		D1	**JoinEPbDMA**	1 Join EPb to DMA 0 Do nothing	0	R/W	
		D0	**JoinEPaDMA**	1 Join EPa to DMA 0 Do nothing	0	R/W	
DMA_Control (DMA control)	**0x300c92** (8 bits)	D7	**DMA_Running**	1 DMA is running 0 DMA is not running	0	R	
		D6	**PDREQ**	PDREQ signal logic	0	R	
		D5	**PDACK**	PDACK signal logic	0	R	
		D4	**–**	–	–	–	0 when being read.
		D3	**CounterClr**	1 Clear DMA counter 0 Do nothing	0	W	
		D2	**–**	–	–	–	
		D1	**DMA_Stop**	1 Finish DMA 0 Do nothing	0	W	
		D0	**DMA_Go**	1 Start DMA 0 Do nothing	0	W	
DMA_Config_0 (DMA configuration 0)	**0x300c94** (8 bits)	D7	**ActivePort**	1 Activate DMA port 0 Deactivate DMA port	0	R/W	
		D6–4	**–**	–	–	–	0 when being read.
		D3	**PDREQ_Level**	1 Active-low 0 Active-high	0	R/W	
		D2	**PDACK_Level**	1 Active-low 0 Active-high	0	R/W	
		D1	**PDRDWR_Level**	1 Active-low 0 Active-high	0	R/W	
		D0	**–**	–	–	–	0 when being read.
DMA_Config_1 (DMA configuration 1)	**0x300c95** (8 bits)	D7	**RcvLimitMode**	1 Receive limit mode 0 Normal	0	R/W	
		D6–4	**–**	–	–	–	0 when being read.
		D3	**SingleWord**	1 Single word 0 Multi word	0	R/W	
		D2–1	**–**	–	–	–	0 when being read.
		D0	**CountMode**	1 Count-down mode 0 Free-run mode	0	R/W	
DMA_Latency (DMA latency)	**0x300c97** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**DMA_Latency[3:0]**	Latency	0x0	R/W	
DMA_Remain_H (DMA FIFO remain high)	**0x300c98** (8 bits)	D7–4	**–**	–	–	–	0 when being read.
		D3–0	**DMA_Remain[11:8]**	DMA FIFO remain	0x0	R	
DMA_Remain_L (DMA FIFO remain low)	**0x300c99** (8 bits)	D7–0	**DMA_Remain[7:0]**	DMA FIFO remain	0x0	R	
DMA_Count_HH (DMA transfer byte counter high/high)	**0x300c9c** (8 bits)	D7–0	**DMA_Count[31:24]**	DMA transfer byte counter	0x0	R/W	
DMA_Count_HL (DMA transfer byte counter high/low)	**0x300c9d** (8 bits)	D7–0	**DMA_Count[23:16]**	DMA transfer byte counter	0x0	R/W	
DMA_Count_LH (DMA transfer byte counter low/high)	**0x300c9e** (8 bits)	D7–0	**DMA_Count[15:8]**	DMA transfer byte counter	0x0	R/W	
DMA_Count_LL (DMA transfer byte counter low/low)	**0x300c9f** (8 bits)	D7–0	**DMA_Count[7:0]**	DMA transfer byte counter	0x0	R/W	

0x300e00–0x300e01 Prescaler (PSC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
PSC0 Control Register (PSC_CTL0)	**0x300e00** (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	
PSC1/2 Control Register (PSC_CTL12)	**0x300e01** (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	

0x300e10–0x300e15 UART

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
UART Status Register (UART_ST)	0x300e10 (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)	0x300e11 (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)	0x300e12 (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)	0x300e13 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5	–	reserved	–	0	–	Do not set to 1.
		D4	CHLN	Character length select	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)	0x300e14 (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 setup	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)	0x300e15 (8 bits)	D7	–	reserved	–	–	–	0 when being read.
		D6–4	IRCLK[2:0]	IrDA receive detection clock select	IRCLK[2:0] / Division ratio: 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
		D3–1	–	reserved	–	–	–	0 when being read.
		D0	IRMD	IrDA mode select	1 On 0 Off	0	R/W	

0x300f00–0x300f43 Card Interface (CARD)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CARD EDC Trigger Register (CARD_EDC_TRG)	0x300f00 (8 bits)	D7–2	–	reserved	–	–	–	0 when being read.
		D1	END	10-symbol EDC end trigger	1 Trigger 0 Ignored	0	W	
		D0	RST	EDC reset	1 Reset 0 Ignored	0	W	
CARD EDC Control Register (CARD_EDC_CTRL)	0x300f01 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2	SLT	EDC correction mode select	1 2-symbol EDC 0 10-symbol EDC	0	R/W	
		D1	MOD	EDC operation mode select	1 Decode 0 Encode	0	R/W	
		D0	EN	EDC enable	1 Enable 0 Disable	0	R/W	
CARD EDC Region Register (CARD_EDC_RGN)	0x300f02 (8 bits)	D7–3	–	reserved	–	–	–	0 when being read.
		D2–0	RGN[2:0]	Card area select	RGN[2:0] / Card area: 0x7 Disabled 0x6 reserved 0x5 #CE10 0x4 #CE9 0x3 #CE8 0x2 #CE7 0x1 #CE5 0x0 #CE4	0x4	R/W	
CARD EDC Flag Register (CARD_EDC_FLG)	0x300f10 (8 bits)	D7–6	–	reserved	–	–	–	0 when being read.
		D5–4	BSY[1:0]	10-symbol EDC busy flag	BSY[1:0] / Status: 0x3 Busy 0x2, 0x1 reserved 0x0 Idle	0x0	R	
		D3–1	–	reserved	–	–	–	0 when being read.
		D0	ERR	EDC error flag	1 Error 0 No error	0	R	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CARD 10-Symbol EDC Code Register 0 (CARD_EDC10_ COD0)	**0x300f20** (8 bits)	D7–0	**RS10_ COD0[7:0]**	RS code 0[7:0]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 1 (CARD_EDC10_ COD1)	**0x300f21** (8 bits)	D7–2	**RS10_ COD1[5:0]**	RS code 1[5:0]	0x0 to 0x3f	0x0	R	
		D1–0	**RS10_ COD0[9:8]**	RS code 0[9:8]	0x0 to 0x3	0x0	R	
CARD 10-Symbol EDC Code Register 2 (CARD_EDC10_ COD2)	**0x300f22** (8 bits)	D7–4	**RS10_ COD2[3:0]**	RS code 2[3:0]	0x0 to 0xf	0x0	R	
		D3–0	**RS10_ COD1[9:6]**	RS code 1[9:6]	0x0 to 0xf	0x0	R	
CARD 10-Symbol EDC Code Register 3 (CARD_EDC10_ COD3)	**0x300f23** (8 bits)	D7–6	**RS10_ COD3[1:0]**	RS code 3[1:0]	0x0 to 0x3	0x0	R	
		D5–0	**RS10_ COD2[9:4]**	RS code 2[9:4]	0x0 to 0x3f	0x0	R	
CARD 10-Symbol EDC Code Register 4 (CARD_EDC10_ COD4)	**0x300f24** (8 bits)	D7–0	**RS10_ COD3[9:2]**	RS code 3[9:2]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 5 (CARD_EDC10_ COD5)	**0x300f25** (8 bits)	D7–0	**RS10_ COD4[7:0]**	RS code 4[7:0]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 6 (CARD_EDC10_ COD6)	**0x300f26** (8 bits)	D7–2	**RS10_ COD5[5:0]**	RS code 5[5:0]	0x0 to 0x3f	0x0	R	
		D1–0	**RS10_ COD4[9:8]**	RS code 4[9:8]	0x0 to 0x3	0x0	R	
CARD 10-Symbol EDC Code Register 7 (CARD_EDC10_ COD7)	**0x300f27** (8 bits)	D7–4	**RS10_ COD6[3:0]**	RS code 6[3:0]	0x0 to 0xf	0x0	R	
		D3–0	**RS10_ COD5[9:6]**	RS code 5[9:6]	0x0 to 0xf	0x0	R	
CARD 10-Symbol EDC Code Register 8 (CARD_EDC10_ COD8)	**0x300f28** (8 bits)	D7–6	**RS10_ COD7[1:0]**	RS code 7[1:0]	0x0 to 0x3	0x0	R	
		D5–0	**RS10_ COD6[9:4]**	RS code 6[9:4]	0x0 to 0x3f	0x0	R	
CARD 10-Symbol EDC Code Register 9 (CARD_EDC10_ COD9)	**0x300f29** (8 bits)	D7–0	**RS10_ COD7[9:2]**	RS code 7[9:2]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 10 (CARD_EDC10_ COD10)	**0x300f2a** (8 bits)	D7–0	**RS10_ COD8[7:0]**	RS code 8[7:0]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 11 (CARD_EDC10_ COD11)	**0x300f2b** (8 bits)	D7–2	**RS10_ COD9[5:0]**	RS code 9[5:0]	0x0 to 0x3f	0x0	R	
		D1–0	**RS10_ COD8[9:8]**	RS code 8[9:8]	0x0 to 0x3	0x0	R	
CARD 10-Symbol EDC Code Register 12 (CARD_EDC10_ COD12)	**0x300f2c** (8 bits)	D7–4	**RS10_ COD10[3:0]**	RS code 10[3:0]	0x0 to 0xf	0x0	R	
		D3–0	**RS10_ COD9[9:6]**	RS code 9[9:6]	0x0 to 0xf	0x0	R	
CARD 10-Symbol EDC Code Register 13 (CARD_EDC10_ COD13)	**0x300f2d** (8 bits)	D7–6	**RS10_ COD11[1:0]**	RS code 11[1:0]	0x0 to 0x3	0x0	R	
		D5–0	**RS10_ COD10[9:4]**	RS code 10[9:4]	0x0 to 0x3f	0x0	R	
CARD 10-Symbol EDC Code Register 14 (CARD_EDC10_ COD14)	**0x300f2e** (8 bits)	D7–0	**RS10_ COD11[9:2]**	RS code 11[9:2]	0x0 to 0xff	0x0	R	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
CARD 10-Symbol EDC Code Register 15 (CARD_EDC10_ COD15)	0x300f2f (8 bits)	D7–0	RS10_ COD12[7:0]	RS code 12[7:0]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 16 (CARD_EDC10_ COD16)	0x300f30 (8 bits)	D7–2	RS10_ COD13[5:0]	RS code 13[5:0]	0x0 to 0x3f	0x0	R	
		D1–0	RS10_ COD12[9:8]	RS code 12[9:8]	0x0 to 0x3	0x0	R	
CARD 10-Symbol EDC Code Register 17 (CARD_EDC10_ COD17)	0x300f31 (8 bits)	D7–4	RS10_ COD14[3:0]	RS code 14[3:0]	0x0 to 0xf	0x0	R	
		D3–0	RS10_ COD13[9:6]	RS code 13[9:6]	0x0 to 0xf	0x0	R	
CARD 10-Symbol EDC Code Register 18 (CARD_EDC10_ COD18)	0x300f32 (8 bits)	D7–6	RS10_ COD15[1:0]	RS code 15[1:0]	0x0 to 0x3	0x0	R	
		D5–0	RS10_ COD14[9:4]	RS code 14[9:4]	0x0 to 0x3f	0x0	R	
CARD 10-Symbol EDC Code Register 19 (CARD_EDC10_ COD19)	0x300f33 (8 bits)	D7–0	RS10_ COD15[9:2]	RS code 15[9:2]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 20 (CARD_EDC10_ COD20)	0x300f34 (8 bits)	D7–0	RS10_ COD16[7:0]	RS code 16[7:0]	0x0 to 0xff	0x0	R	
CARD 10-Symbol EDC Code Register 21 (CARD_EDC10_ COD21)	0x300f35 (8 bits)	D7–2	RS10_ COD17[5:0]	RS code 17[5:0]	0x0 to 0x3f	0x0	R	
		D1–0	RS10_ COD16[9:8]	RS code 16[9:8]	0x0 to 0x3	0x0	R	
CARD 10-Symbol EDC Code Register 22 (CARD_EDC10_ COD22)	0x300f36 (8 bits)	D7–4	RS10_ COD18[3:0]	RS code 18[3:0]	0x0 to 0xf	0x0	R	
		D3–0	RS10_ COD17[9:6]	RS code 17[9:6]	0x0 to 0xf	0x0	R	
CARD 10-Symbol EDC Code Register 23 (CARD_EDC10_ COD23)	0x300f37 (8 bits)	D7–6	RS10_ COD19[1:0]	RS code 19[1:0]	0x0 to 0x3	0x0	R	
		D5–0	RS10_ COD18[9:4]	RS code 18[9:4]	0x0 to 0x3f	0x0	R	
CARD 10-Symbol EDC Code Register 24 (CARD_EDC10_ COD24)	0x300f38 (8 bits)	D7–0	RS10_ COD19[9:2]	RS code 19[9:2]	0x0 to 0xff	0x0	R	
CARD 2-Symbol EDC Code Register 0 (CARD_EDC2_ COD0)	0x300f40 (8 bits)	D7–0	RS2_ COD0[7:0]	RS code 0[7:0]	0x0 to 0xff	0x0	R	
CARD 2-Symbol EDC Code Register 1 (CARD_EDC2_ COD1)	0x300f41 (8 bits)	D7–0	RS2_ COD1[7:0]	RS code 1[7:0]	0x0 to 0xff	0x0	R	
CARD 2-Symbol EDC Code Register 2 (CARD_EDC2_ COD2)	0x300f42 (8 bits)	D7–0	RS2_ COD2[7:0]	RS code 2[7:0]	0x0 to 0xff	0x0	R	
CARD 2-Symbol EDC Code Register 3 (CARD_EDC2_ COD3)	0x300f43 (8 bits)	D7–0	RS2_ COD3[7:0]	RS code 3[7:0]	0x0 to 0xff	0x0	R	

0x301000–0x30100c Watchdog Timer (WDT)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
WDT Write Protect Register (WD_ PROTECT)	0x301000 (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 (0x301002–0x301006). Writing another value set the write protection.	X	W	0 when being read.
WDT Enable and Setup Register (WD_EN)	0x301002 (16 bits)	D15–7	–	reserved	–	–	–	0 when being read.
		D6	CLKSEL	WDT input clock select	1 External clk 0 Internal clk	0	R/W	Write-protected
		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	Write-protected
		D0	RESEN	WDT RESET enable	1 Enable 0 Disable	0	R/W	
WDT Comparison Data L Register (WD_CMP_L)	0x301004 (16 bits)	D15–0	CMPDT [15:0]	WDT comparison data CMPDT0 = LSB	0x0 to 0x3fffffff (low-order 16 bits)	0x0	R/W	Write-protected
WDT Comparison Data H Register (WD_CMP_H)	0x301006 (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	Write-protected
WDT Count Data L Register (WD_CNT_L)	0x301008 (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)	0x30100a (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)	0x30100c (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	WDRESEN	WDT reset	1 Reset 0 ignored	0	W	

0x301100–0x301108 Fine Mode 8-bit Timer (T8F) Ch.0

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.0 Input Clock Select Register (T8F_CLK0)	0x301100 (16 bits)	D15–4	–	reserved	–	–	–	0 when being read.
		D3–0	DF[3:0]	T8F clock division ratio select (Prescaler output clock)	DF[3:0] / Division ratio: 0xf reserved; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T8F Ch.0 Reload Data Register (T8F_TR0)	0x301102 (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	TR[7:0]	T8F reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
T8F Ch.0 Counter Data Register (T8F_TC0)	0x301104 (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7–0	TC[7:0]	T8F counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0x0	R	
T8F Ch.0 Control Register (T8F_CTL0)	0x301106 (16 bits)	D15–12	–	reserved	–	–	–	0 when being read.
		D11–8	TFMD[3:0]	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
		D7–5	–	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	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.0 Interrupt Control Register (T8F_INT0)	**0x301108** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**T8FIE**	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**T8FIF**	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

0x301110–0x301118 Fine Mode 8-bit Timer (T8F) Ch.1

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.1 Input Clock Select Register (T8F_CLK1)	**0x301110** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	T8F clock division ratio select (Prescaler output clock)	DF[3:0] / Division ratio: 0xf reserved; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T8F Ch.1 Reload Data Register (T8F_TR1)	**0x301112** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	T8F reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
T8F Ch.1 Counter Data Register (T8F_TC1)	**0x301114** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	T8F counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0x0	R	
T8F Ch.1 Control Register (T8F_CTL1)	**0x301116** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**TFMD[3:0]**	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
		D7–5	**–**	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	
T8F Ch.1 Interrupt Control Register (T8F_INT1)	**0x301118** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**T8FIE**	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**T8FIF**	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

0x301120–0x301128 Fine Mode 8-bit Timer (T8F) Ch.2

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.2 Input Clock Select Register (T8F_CLK2)	**0x301120** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	T8F clock division ratio select (Prescaler output clock)	DF[3:0] / Division ratio: 0xf reserved; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T8F Ch.2 Reload Data Register (T8F_TR2)	**0x301122** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	T8F reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.2 Counter Data Register (T8F_TC2)	**0x301124** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	T8F counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0x0	R	
T8F Ch.2 Control Register (T8F_CTL2)	**0x301126** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**TFMD[3:0]**	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
		D7–5	**–**	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	
T8F Ch.2 Interrupt Control Register (T8F_INT2)	**0x301128** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**T8FIE**	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**T8FIF**	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

0x301130–0x301138 Fine Mode 8-bit Timer (T8F) Ch.3

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.3 Input Clock Select Register (T8F_CLK3)	**0x301130** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	T8F clock division ratio select (Prescaler output clock)	DF[3:0] / Division ratio: 0xf reserved 0xe 1/16384 0xd 1/8192 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T8F Ch.3 Reload Data Register (T8F_TR3)	**0x301132** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	T8F reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
T8F Ch.3 Counter Data Register (T8F_TC3)	**0x301134** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	T8F counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0x0	R	
T8F Ch.3 Control Register (T8F_CTL3)	**0x301136** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**TFMD[3:0]**	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
		D7–5	**–**	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	
T8F Ch.3 Interrupt Control Register (T8F_INT3)	**0x301138** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**T8FIE**	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**T8FIF**	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

0x301140–0x301148 Fine Mode 8-bit Timer (T8F) Ch.4

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.4 Input Clock Select Register (T8F_CLK4)	**0x301140** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	T8F clock division ratio select (Prescaler output clock)	DF[3:0] / Division ratio 0xf reserved 0xe 1/16384 0xd 1/8192 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T8F Ch.4 Reload Data Register (T8F_TR4)	**0x301142** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	T8F reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
T8F Ch.4 Counter Data Register (T8F_TC4)	**0x301144** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	T8F counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0x0	R	
T8F Ch.4 Control Register (T8F_CTL4)	**0x301146** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**TFMD[3:0]**	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
		D7–5	**–**	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	
T8F Ch.4 Interrupt Control Register (T8F_INT4)	**0x301148** (16 bits)	D15–9	**–**	reserved	–	–	–	0 when being read.
		D8	**T8FIE**	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**T8FIF**	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

0x301150–0x301158 Fine Mode 8-bit Timer (T8F) Ch.5

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.5 Input Clock Select Register (T8F_CLK5)	**0x301150** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**DF[3:0]**	T8F clock division ratio select (Prescaler output clock)	DF[3:0] / Division ratio 0xf reserved 0xe 1/16384 0xd 1/8192 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T8F Ch.5 Reload Data Register (T8F_TR5)	**0x301152** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TR[7:0]**	T8F reload data TR7 = MSB TR0 = LSB	0x0 to 0xff	0x0	R/W	
T8F Ch.5 Counter Data Register (T8F_TC5)	**0x301154** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**TC[7:0]**	T8F counter data TC7 = MSB TC0 = LSB	0x0 to 0xff	0x0	R	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T8F Ch.5 Control Register (T8F_CTL5)	0x301156 (16 bits)	D15–12	–	reserved	–	–	–	0 when being read.
		D11–8	TFMD[3:0]	Fine mode setup	0x0 to 0xf	0x0	R/W	Set a number of times to insert delay into a 16-underflow period.
		D7–5	–	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	
T8F Ch.5 Interrupt Control Register (T8F_INT5)	0x301158 (16 bits)	D15–9	–	reserved	–	–	–	0 when being read.
		D8	T8FIE	T8F interrupt enable	1 Enable 0 Disable	0	R/W	
		D7–1	–	reserved	–	–	–	0 when being read.
		D0	T8FIF	T8F interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.

0x301160–0x30116c 16-bit PWM Timer (T16A6) Ch.0

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.0 Counter Control Register (T16A_CTL0)	0x301160 (16 bits)	D15–14	–	reserved	–	–	–	0 when being read.
		D13–12	DMASEL [1:0]	DMAC Ch.2/3 invoking channel select	DMASEL[1:0] T16A6 channel 0x3 Ch.3 0x2 Ch.2 0x1 Ch.1 0x0 Ch.0	0x0	R/W	
		D11–8	CLKS[3:0]	Counter clock (division ratio) select	CLKS[3:0] Division ratio 0xf External clock 0xe 1/16384 0xd 1/8192 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	BUSY	Register writing status	1 Busy 0 Idle	0	R	
		D6	–	reserved	–	–	–	0 when being read.
		D5–4	T16SEL [1:0]	Counter select	T16SEL[1:0] Counter channel 0x3 Ch.3 0x2 Ch.2 0x1 Ch.1 0x0 Ch.0	0x0	R/W	
		D3	CBUFEN	Compare buffer enable	1 Enable 0 Disable	0	R/W	
		D2	TMMD	Count mode select	1 One-shot 0 Repeat	0	R/W	
		D1	PRESET	Counter reset	1 Reset 0 Ignored	0	W	0 when being read.
		D0	PRUN	Counter run/stop control	1 Run 0 Stop	0	R/W	
T16A6 Ch.0 Counter Data Register (T16A_TC0)	0x301162 (16 bits)	D15–0	T16ATC [15:0]	Counter data T16ATC15 = MSB T16ATC0 = LSB	0x0 to 0xffff	0x0	R	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.0 Comparator/ Capture Control Register (T16A_CCCTL0)	**0x301164** (16 bits)	D15–14	**CAPBTRG [1:0]**	Capture B trigger select	CAPBTRG[1:0] / Trigger edge 0x3 / ↑ and ↓ 0x2 / ↓ 0x1 / ↑ 0x0 / None	0x0	R/W	
		D13–12	**TOUTBMD [1:0]**	TOUT B mode select	TOUTBMD[1:0] / Mode 0x3 / cmp B: ↑ or ↓ 0x2 / cmp A: ↑ or ↓ 0x1 / cmp A: ↑, B: ↓ 0x0 / Off	0x0	R/W	
		D11–10	**–**	reserved	–	–	–	0 when being read.
		D9	**TOUTBINV**	TOUT B invert	1 Invert 0 Normal	0	R/W	
		D8	**CCBMD**	T16A_CCB register mode select	1 Capture 0 Comparator	0	R/W	
		D7–6	**CAPATRG [1:0]**	Capture A trigger select	CAPATRG[1:0] / Trigger edge 0x3 / ↑ and ↓ 0x2 / ↓ 0x1 / ↑ 0x0 / None	0x0	R/W	
		D5–4	**TOUTAMD [1:0]**	TOUT A mode select	TOUTAMD[1:0] / Mode 0x3 / cmp B: ↑ or ↓ 0x2 / cmp A: ↑ or ↓ 0x1 / cmp A: ↑, B: ↓ 0x0 / Off	0x0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**TOUTAINV**	TOUT A invert	1 Invert 0 Normal	0	R/W	
		D0	**CCAMD**	T16A_CCA register mode select	1 Capture 0 Comparator	0	R/W	
T16A6 Ch.0 Comparator/ Capture A Data Register (T16A_CCA0)	**0x301166** (16 bits)	D15–0	**CCA[15:0]**	Compare/capture A data CCA15 = MSB CCA0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.0 Comparator/ Capture B Data Register (T16A_CCB0)	**0x301168** (16 bits)	D15–0	**CCB[15:0]**	Compare/capture B data CCB15 = MSB CCB0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.0 Comparator/ Capture Interrupt Enable Register (T16A_IEN0)	**0x30116a** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIE**	Capture B overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**CAPAOWIE**	Capture A overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D3	**CAPBIE**	Capture B interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**CAPAIE**	Capture A interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**CBIE**	Compare B interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**CAIE**	Compare A interrupt enable	1 Enable 0 Disable	0	R/W	
T16A6 Ch.0 Comparator/ Capture Interrupt Flag Register (T16A_IFLG0)	**0x30116c** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIF**	Capture B overwrite interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D4	**CAPAOWIF**	Capture A overwrite interrupt flag		0	R/W	
		D3	**CAPBIF**	Capture B interrupt flag		0	R/W	
		D2	**CAPAIF**	Capture A interrupt flag		0	R/W	
		D1	**CBIF**	Compare B interrupt flag		0	R/W	
		D0	**CAIF**	Compare A interrupt flag		0	R/W	

0x301170–0x30117c 16-bit PWM Timer (T16A6) Ch.1

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.1 Counter Control Register (T16A_CTL1)	**0x301170** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13–12	**DMASEL [1:0]**	DMAC Ch.4/5 invoking channel select	DMASEL[1:0] / T16A6 channel 0x3 Ch.3 0x2 Ch.2 0x1 Ch.1 0x0 Ch.0	0x0	R/W	
		D11–8	**CLKS[3:0]**	Counter clock (division ratio) select	CLKS[3:0] / Division ratio 0xf External clock 0xe 1/16384 0xd 1/8192 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	**BUSY**	Register writing status	1 Busy / 0 Idle	0	R	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**T16SEL [1:0]**	Counter select	T16SEL[1:0] / Counter channel 0x3 Ch.3 0x2 Ch.2 0x1 Ch.1 0x0 Ch.0	0x1	R/W	
		D3	**CBUFEN**	Compare buffer enable	1 Enable / 0 Disable	0	R/W	
		D2	**TMMD**	Count mode select	1 One-shot / 0 Repeat	0	R/W	
		D1	**PRESET**	Counter reset	1 Reset / 0 Ignored	0	W	0 when being read.
		D0	**PRUN**	Counter run/stop control	1 Run / 0 Stop	0	R/W	
T16A6 Ch.1 Counter Data Register (T16A_TC1)	**0x301172** (16 bits)	D15–0	**T16ATC [15:0]**	Counter data T16ATC15 = MSB T16ATC0 = LSB	0x0 to 0xffff	0x0	R	
T16A6 Ch.1 Comparator/ Capture Control Register (T16A_CCCTL1)	**0x301174** (16 bits)	D15–14	**CAPBTRG [1:0]**	Capture B trigger select	CAPBTRG[1:0] / Trigger edge 0x3 ↑ and ↓ 0x2 ↓ 0x1 ↑ 0x0 None	0x0	R/W	
		D13–12	**TOUTBMD [1:0]**	TOUT B mode select	TOUTBMD[1:0] / Mode 0x3 cmp B: ↑ or ↓ 0x2 cmp A: ↑ or ↓ 0x1 cmp A: ↑, B: ↓ 0x0 Off	0x0	R/W	
		D11–10	**–**	reserved	–	–	–	0 when being read.
		D9	**TOUTBINV**	TOUT B invert	1 Invert / 0 Normal	0	R/W	
		D8	**CCBMD**	T16A_CCB register mode select	1 Capture / 0 Comparator	0	R/W	
		D7–6	**CAPATRG [1:0]**	Capture A trigger select	CAPATRG[1:0] / Trigger edge 0x3 ↑ and ↓ 0x2 ↓ 0x1 ↑ 0x0 None	0x0	R/W	
		D5–4	**TOUTAMD [1:0]**	TOUT A mode select	TOUTAMD[1:0] / Mode 0x3 cmp B: ↑ or ↓ 0x2 cmp A: ↑ or ↓ 0x1 cmp A: ↑, B: ↓ 0x0 Off	0x0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**TOUTAINV**	TOUT A invert	1 Invert / 0 Normal	0	R/W	
		D0	**CCAMD**	T16A_CCA register mode select	1 Capture / 0 Comparator	0	R/W	
T16A6 Ch.1 Comparator/ Capture A Data Register (T16A_CCA1)	**0x301176** (16 bits)	D15–0	**CCA[15:0]**	Compare/capture A data CCA15 = MSB CCA0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.1 Comparator/ Capture B Data Register (T16A_CCB1)	**0x301178** (16 bits)	D15–0	**CCB[15:0]**	Compare/capture B data CCB15 = MSB CCB0 = LSB	0x0 to 0xffff	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.1 Comparator/ Capture Interrupt Enable Register (T16A_IEN1)	**0x30117a** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIE**	Capture B overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**CAPAOWIE**	Capture A overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D3	**CAPBIE**	Capture B interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**CAPAIE**	Capture A interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**CBIE**	Compare B interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**CAIE**	Compare A interrupt enable	1 Enable 0 Disable	0	R/W	
T16A6 Ch.1 Comparator/ Capture Interrupt Flag Register (T16A_IFLG1)	**0x30117c** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIF**	Capture B overwrite interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D4	**CAPAOWIF**	Capture A overwrite interrupt flag		0	R/W	
		D3	**CAPBIF**	Capture B interrupt flag		0	R/W	
		D2	**CAPAIF**	Capture A interrupt flag		0	R/W	
		D1	**CBIF**	Compare B interrupt flag		0	R/W	
		D0	**CAIF**	Compare A interrupt flag		0	R/W	

0x301180–0x30118c 16-bit PWM Timer (T16A6) Ch.2

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.2 Counter Control Register (T16A_CTL2)	**0x301180** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**CLKS[3:0]**	Counter clock (division ratio) select	CLKS[3:0] / Division ratio: 0xf External clock; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	**BUSY**	Register writing status	1 Busy 0 Idle	0	R	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**T16SEL [1:0]**	Counter select	T16SEL[1:0] / Counter channel: 0x3 Ch.3; 0x2 Ch.2; 0x1 Ch.1; 0x0 Ch.0	0x2	R/W	
		D3	**CBUFEN**	Compare buffer enable	1 Enable 0 Disable	0	R/W	
		D2	**TMMD**	Count mode select	1 One-shot 0 Repeat	0	R/W	
		D1	**PRESET**	Counter reset	1 Reset 0 Ignored	0	W	0 when being read.
		D0	**PRUN**	Counter run/stop control	1 Run 0 Stop	0	R/W	
T16A6 Ch.2 Counter Data Register (T16A_TC2)	**0x301182** (16 bits)	D15–0	**T16ATC [15:0]**	Counter data T16ATC15 = MSB T16ATC0 = LSB	0x0 to 0xffff	0x0	R	
T16A6 Ch.2 Comparator/ Capture Control Register (T16A_CCCTL2)	**0x301184** (16 bits)	D15–14	**CAPBTRG [1:0]**	Capture B trigger select	CAPBTRG[1:0] / Trigger edge: 0x3 ↑ and ↓; 0x2 ↓; 0x1 ↑; 0x0 None	0x0	R/W	
		D13–12	**TOUTBMD [1:0]**	TOUT B mode select	TOUTBMD[1:0] / Mode: 0x3 cmp B: ↑ or ↓; 0x2 cmp A: ↑ or ↓; 0x1 cmp A: ↑, B: ↓; 0x0 Off	0x0	R/W	
		D11–10	**–**	reserved	–	–	–	0 when being read.
		D9	**TOUTBINV**	TOUT B invert	1 Invert 0 Normal	0	R/W	
		D8	**CCBMD**	T16A_CCB register mode select	1 Capture 0 Comparator	0	R/W	
		D7–6	**CAPATRG [1:0]**	Capture A trigger select	CAPATRG[1:0] / Trigger edge: 0x3 ↑ and ↓; 0x2 ↓; 0x1 ↑; 0x0 None	0x0	R/W	
		D5–4	**TOUTAMD [1:0]**	TOUT A mode select	TOUTAMD[1:0] / Mode: 0x3 cmp B: ↑ or ↓; 0x2 cmp A: ↑ or ↓; 0x1 cmp A: ↑, B: ↓; 0x0 Off	0x0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**TOUTAINV**	TOUT A invert	1 Invert 0 Normal	0	R/W	
		D0	**CCAMD**	T16A_CCA register mode select	1 Capture 0 Comparator	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.2 Comparator/ Capture A Data Register (T16A_CCA2)	**0x301186** (16 bits)	D15–0	**CCA[15:0]**	Compare/capture A data CCA15 = MSB CCA0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.2 Comparator/ Capture B Data Register (T16A_CCB2)	**0x301188** (16 bits)	D15–0	**CCB[15:0]**	Compare/capture B data CCB15 = MSB CCB0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.2 Comparator/ Capture Interrupt Enable Register (T16A_IEN2)	**0x30118a** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIE**	Capture B overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**CAPAOWIE**	Capture A overwrite interrupt enable	1 Enable 0 Disable	0	R/W	
		D3	**CAPBIE**	Capture B interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**CAPAIE**	Capture A interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**CBIE**	Compare B interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**CAIE**	Compare A interrupt enable	1 Enable 0 Disable	0	R/W	
T16A6 Ch.2 Comparator/ Capture Interrupt Flag Register (T16A_IFLG2)	**0x30118c** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIF**	Capture B overwrite interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D4	**CAPAOWIF**	Capture A overwrite interrupt flag		0	R/W	
		D3	**CAPBIF**	Capture B interrupt flag		0	R/W	
		D2	**CAPAIF**	Capture A interrupt flag		0	R/W	
		D1	**CBIF**	Compare B interrupt flag		0	R/W	
		D0	**CAIF**	Compare A interrupt flag		0	R/W	

0x301190–0x30119c 16-bit PWM Timer (T16A6) Ch.3

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.3 Counter Control Register (T16A_CTL3)	**0x301190** (16 bits)	D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–8	**CLKS[3:0]**	Counter clock (division ratio) select	CLKS[3:0] / Division ratio: 0xf External clock 0xe 1/16384 0xd 1/8192 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = TCLK
		D7	**BUSY**	Register writing status	1 Busy 0 Idle	0	R	
		D6	**–**	reserved	–	–	–	0 when being read.
		D5–4	**T16SEL [1:0]**	Counter select	T16SEL[1:0] / Counter channel: 0x3 Ch.3 0x2 Ch.2 0x1 Ch.1 0x0 Ch.0	0x3	R/W	
		D3	**CBUFEN**	Compare buffer enable	1 Enable 0 Disable	0	R/W	
		D2	**TMMD**	Count mode select	1 One-shot 0 Repeat	0	R/W	
		D1	**PRESET**	Counter reset	1 Reset 0 Ignored	0	W	0 when being read.
		D0	**PRUN**	Counter run/stop control	1 Run 0 Stop	0	R/W	
T16A6 Ch.3 Counter Data Register (T16A_TC3)	**0x301192** (16 bits)	D15–0	**T16ATC [15:0]**	Counter data T16ATC15 = MSB T16ATC0 = LSB	0x0 to 0xffff	0x0	R	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16A6 Ch.3 Comparator/ Capture Control Register (T16A_CCCTL3)	**0x301194** (16 bits)	D15–14	**CAPBTRG [1:0]**	Capture B trigger select	CAPBTRG[1:0] / Trigger edge 0x3 / ↑ and ↓ 0x2 / ↓ 0x1 / ↑ 0x0 / None	0x0	R/W	
		D13–12	**TOUTBMD [1:0]**	TOUT B mode select	TOUTBMD[1:0] / Mode 0x3 / cmp B: ↑ or ↓ 0x2 / cmp A: ↑ or ↓ 0x1 / cmp A: ↑, B: ↓ 0x0 / Off	0x0	R/W	
		D11–10	**–**	reserved	–	–	–	0 when being read.
		D9	**TOUTBINV**	TOUT B invert	1 Invert / 0 Normal	0	R/W	
		D8	**CCBMD**	T16A_CCB register mode select	1 Capture / 0 Comparator	0	R/W	
		D7–6	**CAPATRG [1:0]**	Capture A trigger select	CAPATRG[1:0] / Trigger edge 0x3 / ↑ and ↓ 0x2 / ↓ 0x1 / ↑ 0x0 / None	0x0	R/W	
		D5–4	**TOUTAMD [1:0]**	TOUT A mode select	TOUTAMD[1:0] / Mode 0x3 / cmp B: ↑ or ↓ 0x2 / cmp A: ↑ or ↓ 0x1 / cmp A: ↑, B: ↓ 0x0 / Off	0x0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**TOUTAINV**	TOUT A invert	1 Invert / 0 Normal	0	R/W	
		D0	**CCAMD**	T16A_CCA register mode select	1 Capture / 0 Comparator	0	R/W	
T16A6 Ch.3 Comparator/ Capture A Data Register (T16A_CCA3)	**0x301196** (16 bits)	D15–0	**CCA[15:0]**	Compare/capture A data CCA15 = MSB CCA0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.3 Comparator/ Capture B Data Register (T16A_CCB3)	**0x301198** (16 bits)	D15–0	**CCB[15:0]**	Compare/capture B data CCB15 = MSB CCB0 = LSB	0x0 to 0xffff	0x0	R/W	
T16A6 Ch.3 Comparator/ Capture Interrupt Enable Register (T16A_IEN3)	**0x30119a** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIE**	Capture B overwrite interrupt enable	1 Enable / 0 Disable	0	R/W	
		D4	**CAPAOWIE**	Capture A overwrite interrupt enable	1 Enable / 0 Disable	0	R/W	
		D3	**CAPBIE**	Capture B interrupt enable	1 Enable / 0 Disable	0	R/W	
		D2	**CAPAIE**	Capture A interrupt enable	1 Enable / 0 Disable	0	R/W	
		D1	**CBIE**	Compare B interrupt enable	1 Enable / 0 Disable	0	R/W	
		D0	**CAIE**	Compare A interrupt enable	1 Enable / 0 Disable	0	R/W	
T16A6 Ch.3 Comparator/ Capture Interrupt Flag Register (T16A_IFLG3)	**0x30119c** (16 bits)	D15–6	**–**	reserved	–	–	–	0 when being read.
		D5	**CAPBOWIF**	Capture B overwrite interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D4	**CAPAOWIF**	Capture A overwrite interrupt flag		0	R/W	
		D3	**CAPBIF**	Capture B interrupt flag		0	R/W	
		D2	**CAPAIF**	Capture A interrupt flag		0	R/W	
		D1	**CBIF**	Compare B interrupt flag		0	R/W	
		D0	**CAIF**	Compare A interrupt flag		0	R/W	

0x301200–0x30120e 16-bit Audio PWM Timer (T16P) Ch.0

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.0 Compare A Buffer Register (T16P_A0)	**0x301200** (16 bits)	D15–0	**CMPA[15:0]**	Compare A data CMPA15 = MSB CMPA0 = LSB	0x0 to 0xffff	X	R/W	
T16P Ch.0 Compare B Buffer Register (T16P_B0)	**0x301202** (16 bits)	D15–0	**CMPB[15:0]**	Compare B data CMPB15 = MSB CMPB0 = LSB	0x0 to 0xffff	X	R/W	
T16P Ch.0 Counter Data Register (T16P_CNT_ DATA0)	**0x301204** (16 bits)	D15–0	**CNT_DATA [15:0]**	Counter data CNT_DATA15 = MSB CNT_DATA0 = LSB	0x0 to 0xffff	X	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.0 Volume Control Register (T16P_VOL_CTL0)	**0x301206** (16 bits)	D15–8	**–**	reserved	–	–	–	0 when being read.
		D7	**VOLBPS**	Volume control enable	1 Disable 0 Enable	1	R/W	Effective only for 16-bit data
		D6–0	**VOLSEL [6:0]**	Volume level select	VOLSEL[6:0] / Volume level 0x7f / × 127/64 0x7e / × 126/64 : / : 0x40 / × 64/64 : / : 0x2 / × 2/64 0x1 / × 1/64 0x0 / × 0 (mute)	0x40	R/W	
T16P Ch.0 Control Register (T16P_CTL0)	**0x301208** (16 bits)	D15–12	**BCNT[3:0]**	B match count	0x0 to 0xf	0x0	R/W	
		D11	**RESSEL**	PCM data resolution select	1 16 bits 0 8 bits	1	R/W	
		D10	**SGNSEL**	PCM data format select	1 Signed 0 Unsigned	1	R/W	
		D9–8	**SPLTMD [1:0]**	Split mode select	SPLTMD[1:0] / Split mode 0x3 / 10 bits + 6 bits 0x2 / 9 bits + 7 bits 0x1 / 8 bits + 8 bits 0x0 / Normal (16 bits)	0x0	R/W	Effective only for 16-bit data
		D7	**CHSYNC**	Channel sync enable	1 Enable 0 Disable	0	R/W	
		D6	**SELFM**	Fine mode select	1 Fine mode 0 Normal	0	R/W	
		D5	**–**	reserved	–	–	–	0 when being read.
		D4	**INITOL**	Initial output level select	1 High 0 Low	0	R/W	
		D3	**CLKSEL**	Input clock select	1 External 0 Internal	0	R/W	
		D2	**–**	reserved	–	–	–	0 when being read.
		D1	**PRESET**	T16P reset	1 Reset 0 Ignored	0	W	
		D0	**–**	reserved	–	–	–	
T16P Ch.0 Running Control Register (T16P_RUN0)	**0x30120a** (16 bits)	D15–1	**–**	reserved	–	–	–	0 when being read.
		D0	**PRUN**	Timer run/stop control	1 Run 0 Stop	0	R/W	
T16P Ch.0 Internal Clock Control Register (T16P_CLK0)	**0x30120c** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**CLKDIV [3:0]**	Counter clock division ratio select (Prescaler output clock)	CLKDIV[3:0] / Division ratio 0xf–0xd / reserved 0xc / 1/4096 0xb / 1/2048 0xa / 1/1024 0x9 / 1/512 0x8 / 1/256 0x7 / 1/128 0x6 / 1/64 0x5 / 1/32 0x4 / 1/16 0x3 / 1/8 0x2 / 1/4 0x1 / 1/2 0x0 / 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T16P Ch.0 Interrupt Control Register (T16P_INT0)	**0x30120e** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10	**BUFEF**	Buffer empty interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	1	R/W	Reset by writing 1.
		D9	**INTBF**	B match interrupt flag		0	R/W	
		D8	**INTAF**	A match interrupt flag		0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2	**INTBEEN**	Buffer empty interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**INTBEN**	B match interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**INTAEN**	A match interrupt enable	1 Enable 0 Disable	0	R/W	

0x301210–0x30121e 16-bit Audio PWM Timer (T16P) Ch.1

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.1 Compare A Buffer Register (T16P_A1)	**0x301210** (16 bits)	D15–0	**CMPA[15:0]**	Compare A data CMPA15 = MSB CMPA0 = LSB	0x0 to 0xffff	X	R/W	
T16P Ch.1 Compare B Buffer Register (T16P_B1)	**0x301212** (16 bits)	D15–0	**CMPB[15:0]**	Compare B data CMPB15 = MSB CMPB0 = LSB	0x0 to 0xffff	X	R/W	
T16P Ch.1 Counter Data Register (T16P_CNT_DATA1)	**0x301214** (16 bits)	D15–0	**CNT_DATA [15:0]**	Counter data CNT_DATA15 = MSB CNT_DATA0 = LSB	0x0 to 0xffff	X	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
T16P Ch.1 Volume Control Register (T16P_VOL_CTL1)	0x301216 (16 bits)	D15–8	–	reserved	–	–	–	0 when being read.
		D7	VOLBPS	Volume control enable	1 Disable 0 Enable	1	R/W	Effective only for 16-bit data
		D6–0	VOLSEL [6:0]	Volume level select	VOLSEL[6:0] Volume level 0x7f × 127/64 0x7e × 126/64 : 0x40 × 64/64 : 0x2 × 2/64 0x1 × 1/64 0x0 × 0 (mute)	0x40	R/W	
T16P Ch.1 Control Register (T16P_CTL1)	0x301218 (16 bits)	D15–12	BCNT[3:0]	B match count	0x0 to 0xf	0x0	R/W	
		D11	RESSEL	PCM data resolution select	1 16 bits 0 8 bits	1	R/W	
		D10	SGNSEL	PCM data format select	1 Signed 0 Unsigned	1	R/W	
		D9–8	SPLTMD [1:0]	Split mode select	SPLTMD[1:0] Split mode 0x3 10 bits + 6 bits 0x2 9 bits + 7 bits 0x1 8 bits + 8 bits 0x0 Normal (16 bits)	0x0	R/W	Effective only for 16-bit data
		D7	CHSYNC	Channel sync enable	1 Enable 0 Disable	0	R/W	
		D6	SELFM	Fine mode select	1 Fine mode 0 Normal	0	R/W	
		D5	–	reserved	–	–	–	0 when being read.
		D4	INITOL	Initial output level select	1 High 0 Low	0	R/W	
		D3	CLKSEL	Input clock select	1 External 0 Internal	0	R/W	
		D2	–	reserved	–	–	–	0 when being read.
		D1	PRESET	T16P reset	1 Reset 0 Ignored	0	W	
		D0	–	reserved	–	–	–	
T16P Ch.1 Running Control Register (T16P_RUN1)	0x30121a (16 bits)	D15–1	–	reserved	–	–	–	0 when being read.
		D0	PRUN	Timer run/stop control	1 Run 0 Stop	0	R/W	
T16P Ch.1 Internal Clock Control Register (T16P_CLK1)	0x30121c (16 bits)	D15–4	–	reserved	–	–	–	0 when being read.
		D3–0	CLKDIV [3:0]	Counter clock division ratio select (Prescaler output clock)	CLKDIV[3:0] Division ratio 0xf–0xd reserved 0xc 1/4096 0xb 1/2048 0xa 1/1024 0x9 1/512 0x8 1/256 0x7 1/128 0x6 1/64 0x5 1/32 0x4 1/16 0x3 1/8 0x2 1/4 0x1 1/2 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
T16P Ch.1 Interrupt Control Register (T16P_INT1)	0x30121e (16 bits)	D15–11	–	reserved	–	–	–	0 when being read.
		D10	BUFEF	Buffer empty interrupt flag	1 Cause of interrupt occurred 0 Cause of interrupt not occurred	1	R/W	Reset by writing 1.
		D9	INTBF	B match interrupt flag		0	R/W	
		D8	INTAF	A match interrupt flag		0	R/W	
		D7–3	–	reserved	–	–	–	0 when being read.
		D2	INTBEEN	Buffer empty interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	INTBEN	B match interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	INTAEN	A match interrupt enable	1 Enable 0 Disable	0	R/W	

0x301300–0x301306 A/D Converter (ADC10)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Conversion Result Register (ADC10_ADD)	0x301300 (16 bits)	D15–0	ADD[15:0]	A/D converted data ADD[9:0] are effective when STMD = 0 (ADD[15:10] = 0) ADD[15:6] are effective when STMD = 1 (ADD[5:0] = 0)	0x0 to 0x3ff	0x0	R	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
A/D Trigger/ Channel Select Register (ADC10_TRG)	**0x301302** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13–11	**ADCE[2:0]**	End channel select	0x0 to 0x7	0x0	R/W	
		D10–8	**ADCS[2:0]**	Start channel select	0x0 to 0x7	0x0	R/W	
		D7	**STMD**	Conversion result storing mode	1 ADD[15:6] 0 ADD[9:0]	0	R/W	
		D6	**ADMS**	Conversion mode select	1 Continuous 0 Single	0	R/W	
		D5–4	**ADTS[1:0]**	Conversion trigger select	ADTS[1:0] / Trigger: 0x3 #ADTRIG pin; 0x2 reserved; 0x1 T8F Ch.4; 0x0 Software	0x0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**ADST[2:0]**	Sampling time setting	ADST[2:0] / Sampling time: 0x7 9 clocks; 0x6 8 clocks; 0x5 7 clocks; 0x4 6 clocks; 0x3 5 clocks; 0x2 4 clocks; 0x1 3 clocks; 0x0 2 clocks	0x7	R/W	
A/D Control/ Status Register (ADC10_CTL)	**0x301304** (16 bits)	D15	**–**	reserved	–	–	–	0 when being read.
		D14–12	**ADICH[2:0]**	Conversion channel indicator	0x0 to 0x7	0x0	R	
		D11	**–**	reserved	–	–	–	0 when being read.
		D10	**ADIBS**	ADC10 status	1 Busy 0 Idle	0	R	
		D9	**ADOWE**	Overwrite error flag	1 Error 0 Normal	0	R/W	Reset by writing 1.
		D8	**ADCF**	Conversion completion flag	1 Completed 0 Run/Stand-by	0	R	Reset when ADC10_ADD is read.
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5	**ADOIE**	Overwrite error interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**ADCIE**	Conversion completion int. enable	1 Enable 0 Disable	0	R/W	
		D3–2	**–**	reserved	–	–	–	0 when being read.
		D1	**ADCTL**	A/D conversion control	1 Start 0 Stop	0	R/W	
		D0	**ADEN**	ADC10 enable	1 Enable 0 Disable	0	R/W	
A/D Clock Control Register (ADC10_CLK)	**0x301306** (16 bits)	D15–4	**–**	reserved	–	–	–	0 when being read.
		D3–0	**ADDF[3:0]**	A/D converter clock division ratio select	ADDF[3:0] / Division ratio: 0xf reserved; 0xe 1/32768; 0xd 1/16384; 0xc 1/8192; 0xb 1/4096; 0xa 1/2048; 0x9 1/1024; 0x8 1/512; 0x7 1/256; 0x6 1/128; 0x5 1/64; 0x4 1/32; 0x3 1/16; 0x2 1/8; 0x1 1/4; 0x0 1/2	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)

0x301500–0x301506 Remote Controller (REMC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Configuration Register (REMC_CFG)	**0x301500** (16 bits)	D15–12	**CGCLK[3:0]**	Carrier generator clock division ratio select (Prescaler output clock)	CGCLK[3:0] LCCLK[3:0] / Division ratio: 0xf reserved; 0xe 1/16384; 0xd 1/8192; 0xc 1/4096; 0xb 1/2048; 0xa 1/1024; 0x9 1/512; 0x8 1/256; 0x7 1/128; 0x6 1/64; 0x5 1/32; 0x4 1/16; 0x3 1/8; 0x2 1/4; 0x1 1/2; 0x0 1/1	0x0	R/W	Source clock = PSC1/2_CLK (PCLK)
		D11–8	**LCCLK[3:0]**	Length counter clock division ratio select (Prescaler output clock)	(same as CGCLK[3:0])	0x0	R/W	
		D7–2	**–**	reserved	–	–	–	0 when being read.
		D1	**REMMD**	REMC mode select	1 Receive 0 Transmit	0	R/W	
		D0	**REMEN**	REMC enable	1 Enable 0 Disable	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
REMC Carrier Length Setup Register (REMC_CAR)	**0x301502** (16 bits)	D15–14	**–**	reserved	–	–	–	0 when being read.
		D13–8	**REMCL[5:0]**	Carrier L length setup	0x0 to 0x3f	0x0	R/W	
		D7–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**REMCH[5:0]**	Carrier H length setup	0x0 to 0x3f	0x0	R/W	
REMC Length Counter Register (REMC_LCNT)	**0x301504** (16 bits)	D15–8	**REMLEN[7:0]**	Transmit/receive data length count (down counter)	0x0 to 0xff	0x0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**REMDT**	Transmit/receive data	1 1 (H) 0 0 (L)	0	R/W	
REMC Interrupt Control Register (REMC_INT)	**0x301506** (16 bits)	D15–11	**–**	reserved	–	–	–	0 when being read.
		D10	**REMFIF**	Falling edge interrupt flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D9	**REMRIF**	Rising edge interrupt flag		0	R/W	
		D8	**REMUIF**	Underflow interrupt flag		0	R/W	
		D7–3	**–**	reserved	–	–	–	0 when being read.
		D2	**REMFIE**	Falling edge interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**REMRIE**	Rising edge interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**REMUIE**	Underflow interrupt enable	1 Enable 0 Disable	0	R/W	

0x302000–0x302094 LCD Controller (LCDC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
LCDC Interrupt Enable Register (LCDC_INT)	**0x302000** (32 bits)	D31–1	**–**	reserved	–	–	–	0 when being read.
		D0	**FRINTEN**	Frame interrupt enable	1 Enable 0 Disable	0	R/W	
Status and Power Save Configuration Register (LCDC_PSAVE)	**0x302004** (32 bits)	D31	**FRINTF**	Frame interrupt flag	1 Occurred 0 Not occurred	0	R/W	Reset by writing 1.
		D30–8	**–**	reserved	–	–	–	0 when being read.
		D7	**VNDPF**	Vertical display status flag	1 VNDP 0 VDP	1	R	
		D6–2	**–**	reserved	–	–	–	0 when being read.
		D1–0	**PSAVE[1:0]**	Power save mode select	PSAVE[1:0] Mode: 0x3 Normal; 0x2 reserved; 0x1 reserved; 0x0 Power save	0x0	R/W	
Horizontal Display Register (LCDC_HDISP)	**0x302010** (32 bits)	D31–23	**–**	reserved	–	–	–	0 when being read.
		D22–16	**HTCNT[6:0]**	Horizontal total period (HT) setup HT = HDP + HNDP HT > HDPS + HDP (for TFT)	HT = (HTCNT + 1) × 8 [Ts] HNDP = (HTCNT - HDPCNT) × 8 [Ts]	0x0	R/W	
		D15–7	**–**	reserved	–	–	–	0 when being read.
		D6–0	**HDPCNT [6:0]**	Horizontal display period (HDP) setup	HDP = (HDPCNT + 1) × 8 [Ts]	0x0	R/W	
Vertical Display Register (LCDC_VDISP)	**0x302014** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**VTCNT[9:0]**	Vertical total period (VT) setup VT = VDP + VNDP VT > VDPS + VDP (for TFT)	VT = VTCNT + 1 [lines] VNDP = VTCNT - VDPCNT [lines]	0x0	R/W	
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**VDPCNT [9:0]**	Vertical display period (VDP) setup	VDP = VDPCNT + 1 [lines]	0x0	R/W	
MOD Rate Register (LCDC_MODR)	**0x302018** (32 bits)	D31–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**MOD[5:0]**	LCD MOD rate setup	0x0 to 0x3f	0x0	R/W	
Horizontal Display Start Position Register (LCDC_HDPS)	**0x302020** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**HDPSCNT [9:0]**	Horizontal display period start position for TFT HT > HDP + HDPS	HDPS = HDPSCNT [Ts]	0x0	R/W	0x0 must be set for STN panels.
Vertical Display Start Position Register (LCDC_VDPS)	**0x302024** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**VDPSCNT [9:0]**	Vertical display period start position for TFT VT > VDP + VDPS	VDPS = VDPSCNT [lines]	0x0	R/W	0x0 must be set for STN panels.
FPLINE Pulse Setting Register (LCDC_ FPLINE)	**0x302028** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**FPLINE_ ST[9:0]**	FPLINE pulse start position setup	Start position = FPLINE_ST + 1 [Ts]	0x0	R/W	∗1: For TFT 0x0 must be set for STN panels.
		D15–8	**–**	reserved	–	–	–	0 when being read.
		D7	**FPLINE_ POL**	FPLINE pulse polarity setup	1 Active high 0 Active low	0	R/W	(∗1)
		D6–0	**FPLINE_ WD[6:0]**	FPLINE pulse width setup	Pulse width = FPLINE_WD + 1 [Ts]	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
FPFRAME Pulse Setting Register (LCDC_FPFR)	**0x30202c** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**FPFRAME_ ST[9:0]**	FPFRAME pulse start position setup	Start position = FPFRAME_ST × HT [Ts]	0x0	R/W	∗1: For TFT 0x0 must be set for STN panels.
		D15–8	**–**	reserved	–	–	–	0 when being read.
		D7	**FPFRAME_ POL**	FPFRAME pulse polarity setup	1 Active high 0 Active low	0	R/W	(∗1)
		D6–0	**FPFRAME_ WD[6:0]**	FPFRAME pulse width setup	Pulse width = (FPFRAME_WD+1) × HT [Ts]	0x0	R/W	(∗1)
FPFRAME Pulse Offset Register (LCDC_ FPFROFS)	**0x302030** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**FPFRAME_ STPOFS [9:0]**	FPFRAME pulse stop offset	Stop offset = FPFRAME_ STPOFS [Ts]	0x0	R/W	∗1: For TFT 0x0 must be set for STN panels.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**FPFRAME_ STOFS[9:0]**	FPFRAME pulse start offset	Start offset = FPFRAME_ STOFS [Ts]	0x0	R/W	(∗1)
TFT Special Output Register (LCDC_TFTSO)	**0x302040** (32 bits)	D31–4	**–**	reserved	–	–	–	0 when being read.
		D3	**CTL1CTL**	TFT_CTL1 control	1 Program 0 Toggle/line	0	R/W	For TFT 0x0 must be set for STN panels.
		D2	**CTLCNT_ RUN**	TFT_CTL0–2 control counter run/ stop	1 Run 0 Stop	0	R/W	
		D1	**FPSHIFT_ POL**	FPSHIFT polarity	1 Falling 0 Rising	0	R/W	
		D0	**CTL01SWAP**	TFT_CTL0/TFT_CTL1 swap	1 Swap 0 Not swap	0	R/W	
TFT_CTL1 Pulse Register (LCDC_TFT_ CTL1)	**0x302044** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**CTL1STP [9:0]**	TFT_CTL1 pulse stop offset TFT_CTL1 pulse width = (CTL1STP - CTL1ST +1) Ts	Stop offset = CTL1STP + 1 [Ts]	0x0	R/W	∗2: For TFT This register is enabled when CTLCNT_RUN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**CTL1ST [9:0]**	TFT_CTL1 pulse start offset	Start offset = CTL1ST [Ts]	0x0	R/W	(∗2)
TFT_CTL0 Pulse Register (LCDC_TFT_ CTL0)	**0x302048** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**CTL0STP [9:0]**	TFT_CTL0 pulse stop offset TFT_CTL0 pulse width = (CTL0STP - CTL0ST +1) Ts	Stop offset = CTL0STP + 1 [Ts]	0x0	R/W	∗2: For TFT This register is enabled when CTLCNT_RUN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**CTL0ST [9:0]**	TFT_CTL0 pulse start offset	Start offset = CTL0ST [Ts]	0x0	R/W	(∗2)
TFT_CTL2 Register (LCDC_TFT_ CTL2)	**0x30204c** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**CTL2DLY [9:0]**	TFT_CTL2 delay setup	Delay = CTL2DLY [Ts]	0x0	R/W	∗2: For TFT This register is enabled when CTLCNT_RUN = 1.
LCDC Reload Control Register (LCDC_ RLDCTL)	**0x302050** (32 bits)	D31–2	**–**	reserved	–	–	–	0 when being read.
		D1	**LUTRLD**	LUT reload trigger	1 Trigger 0 Ignored	0	W	
					1 Reloading 0 Finished		R	
		D0	**CTABRLD**	Control table reload trigger	1 Trigger 0 Ignored	0	W	
					1 Reloading 0 Finished		R	
LCDC Reload Table Base Address Register (LCDC_ RLDADR)	**0x302054** (32 bits)	D31–10	**RTBL_ BADR[31:10]**	Reload table base address (1KB boundary address, A[9:0] = 0x0)	Area 3*	0x0	R/W	∗ DSTRAM cannot be used.
		D9–0	**–**	reserved	–	–	–	0 when being read.

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
LCDC Display Mode Register (LCDC_ DISPMOD)	**0x302060** (32 bits)	D31	**PANELSEL**	Panel type select	1 TFT 0 STN	0	R/W	
		D30	**COLOR**	Color/mono select	1 Color 0 Mono	0	R/W	
		D29	**FPSHIFT_ MSK**	FPSHIFT mask enable	1 Enable 0 Disable	0	R/W	
		D28	**–**	reserved	–	–	–	0 when being read.
		D27–26	**DWD[1:0]**	LCD panel data width select	DWD[1:0] Data width 0x3 8 bits (fmt2) 0x2 reserved 0x1 8 bits (fmt1) 0x0 4 bits	0x0	R/W	
		D25	**SWINV**	Software video invert	1 Invert 0 Normal	0	R/W	
		D24	**BLANK**	Display blank enable	1 Blank 0 Normal	0	R/W	
		D23–8	**–**	reserved	–	–	–	0 when being read.
		D7	**FRMRPT**	Frame repeat for EL panel	1 Repeat 0 Not repeat	0	R/W	
		D6–5	**–**	reserved	–	–	–	0 when being read.
		D4	**LUTPASS**	LUT bypass mode select	1 Bypass 0 Use	1	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2–0	**BPP[2:0]**	Bit-per-pixel select	BPP[2:0] bpp 0x7 reserved 0x6 24 bpp 0x5 16 bpp 0x4 12 bpp 0x3 8 bpp 0x2 4 bpp 0x1 2 bpp 0x0 1 bpp	0x0	R/W	
Main Window Display Start Address Register (LCDC_ MAINADR)	**0x302070** (32 bits)	D31–0	**MW_START [31:0]**	Main window start address MW_START31 = MSB MW_START0 = LSB	0x0 to 0xffffffff (Areas 3–5, 7–10, 13–16, and 19–22)	0x0	R/W	
Main Screen Address Offset Register (LCDC_ MAINOFS)	**0x302074** (32 bits)	D31–12	**–**	reserved	–	–	–	0 when being read.
		D11–0	**MW_OFS [11:0]**	Main screen address offset	Main screen width (pixels) × bpp/32	0x0	R/W	
Sub-window Display Start Address Register (LCDC_ SUBADR)	**0x302080** (32 bits)	D31–0	**SW_START [31:0]**	Sub-window start address SW_START31 = MSB SW_START0 = LSB	0x0 to 0xffffffff (Areas 3–5, 7–10, 13–16, and 19–22)	0x0	R/W	
Sub-screen Address Offset Register (LCDC_ SUBOFS)	**0x302084** (32 bits)	D31–12	**–**	reserved	–	–	–	0 when being read.
		D11–0	**SW_OFS [11:0]**	Sub-screen address offset	Sub-screen width (pixels) × bpp/32	0x0	R/W	
Sub-window Start Position Register (LCDC_SUBSP)	**0x302088** (32 bits)	D31	**PIP_EN**	PIP enable	1 Enable 0 Disable	0	R/W	
		D30–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**PIP_ YSTART [9:0]**	Sub-window vertical (Y) start position	Y start position = PIP_YSTART (lines) from the origin	0x0	R/W	∗3: This register is enabled when PIP_EN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**PIP_ XSTART [9:0]**	Sub-window horizontal (X) start position	X start position = PIP_XSTART (pixels) from the origin (word units)	0x0	R/W	(∗3)
Sub-window End Position Register (LCDC_SUBEP)	**0x30208c** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25–16	**PIP_YEND [9:0]**	Sub-window vertical (Y) end position	Y end position = PIP_YEND (lines) from the origin	0x0	R/W	∗3: This register is enabled when PIP_EN = 1.
		D15–10	**–**	reserved	–	–	–	0 when being read.
		D9–0	**PIP_XEND [9:0]**	Sub-window horizontal (X) end position	X end position = PIP_XEND (pixels) from the origin (word units)	0x0	R/W	(∗3)
Monochrome Look-up Table Register 0 (LCDC_MLUT0)	**0x302090** (32 bits)	D31–28	**MLUT7[3:0]**	Monochrome LUT entry 7 data	0x0 to 0xf	0x0	R/W	
		D27–24	**MLUT6[3:0]**	Monochrome LUT entry 6 data	0x0 to 0xf	0x0	R/W	
		D23–20	**MLUT5[3:0]**	Monochrome LUT entry 5 data	0x0 to 0xf	0x0	R/W	
		D19–16	**MLUT4[3:0]**	Monochrome LUT entry 4 data	0x0 to 0xf	0x0	R/W	
		D15–12	**MLUT3[3:0]**	Monochrome LUT entry 3 data	0x0 to 0xf	0x0	R/W	
		D11–8	**MLUT2[3:0]**	Monochrome LUT entry 2 data	0x0 to 0xf	0x0	R/W	
		D7–4	**MLUT1[3:0]**	Monochrome LUT entry 1 data	0x0 to 0xf	0x0	R/W	
		D3–0	**MLUT0[3:0]**	Monochrome LUT entry 0 data	0x0 to 0xf	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Monochrome Look-up Table Register 1 (LCDC_MLUT1)	**0x302094** (32 bits)	D31–28	**MLUT15[3:0]**	Monochrome LUT entry 15 data	0x0 to 0xf	0x0	R/W	
		D27–24	**MLUT14[3:0]**	Monochrome LUT entry 14 data	0x0 to 0xf	0x0	R/W	
		D23–20	**MLUT13[3:0]**	Monochrome LUT entry 13 data	0x0 to 0xf	0x0	R/W	
		D19–16	**MLUT12[3:0]**	Monochrome LUT entry 12 data	0x0 to 0xf	0x0	R/W	
		D15–12	**MLUT11[3:0]**	Monochrome LUT entry 11 data	0x0 to 0xf	0x0	R/W	
		D11–8	**MLUT10[3:0]**	Monochrome LUT entry 10 data	0x0 to 0xf	0x0	R/W	
		D7–4	**MLUT9[3:0]**	Monochrome LUT entry 9 data	0x0 to 0xf	0x0	R/W	
		D3–0	**MLUT8[3:0]**	Monochrome LUT entry 8 data	0x0 to 0xf	0x0	R/W	

0x302100–0x30211c DMA Controller (DMAC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
DMAC Channel Enable Register (DMAC_CH_EN)	**0x302100** (32 bits)	D31–8	**–**	reserved	–	–	–	0 when being read.
		D7	**DMAON7**	DMAC Ch.7 enable	1 Enable 0 Disable	0	R/W	
		D6	**DMAON6**	DMAC Ch.6 enable	1 Enable 0 Disable	0	R/W	
		D5	**DMAON5**	DMAC Ch.5 enable	1 Enable 0 Disable	0	R/W	
		D4	**DMAON4**	DMAC Ch.4 enable	1 Enable 0 Disable	0	R/W	
		D3	**DMAON3**	DMAC Ch.3 enable	1 Enable 0 Disable	0	R/W	
		D2	**DMAON2**	DMAC Ch.2 enable	1 Enable 0 Disable	0	R/W	
		D1	**DMAON1**	DMAC Ch.1 enable	1 Enable 0 Disable	0	R/W	
		D0	**DMAON0**	DMAC Ch.0 enable	1 Enable 0 Disable	0	R/W	
DMAC Control Table Base Address Register (DMAC_TBL_BASE)	**0x302104** (32 bits)	D31–10	**TBL_BASE [31:10]**	DMAC control table base address	0x0 to 0xfffffc00 (1,024-byte boundary address within a RAM)	0x84 000	R/W	
		D9–0	**TBL_BASE [9:0]**	Fixed at 0x0 (Cannot be altered.)			R	
DMAC Interrupt Enable Register (DMAC_IE)	**0x302108** (32 bits)	D31–8	**–**	reserved	–	–	–	0 when being read.
		D7	**DMAIE7**	DMAC Ch.7 interrupt enable	1 Enable 0 Disable	0	R/W	
		D6	**DMAIE6**	DMAC Ch.6 interrupt enable	1 Enable 0 Disable	0	R/W	
		D5	**DMAIE5**	DMAC Ch.5 interrupt enable	1 Enable 0 Disable	0	R/W	
		D4	**DMAIE4**	DMAC Ch.4 interrupt enable	1 Enable 0 Disable	0	R/W	
		D3	**DMAIE3**	DMAC Ch.3 interrupt enable	1 Enable 0 Disable	0	R/W	
		D2	**DMAIE2**	DMAC Ch.2 interrupt enable	1 Enable 0 Disable	0	R/W	
		D1	**DMAIE1**	DMAC Ch.1 interrupt enable	1 Enable 0 Disable	0	R/W	
		D0	**DMAIE0**	DMAC Ch.0 interrupt enable	1 Enable 0 Disable	0	R/W	
DMAC Trigger Select Register (DMAC_TRG_SEL)	**0x30210c** (32 bits)	D31–16	**–**	reserved	–	–	–	0 when being read.
		D15–14	**TRG_SEL7 [1:0]**	Ch.7 trigger select	TRG_SEL7[1:0] Trigger source 0x3 ADC complete 0x2 USI Ch.2 Tx 0x1 USIL Tx 0x0 No hard trigger	0x0	R/W	
		D13–12	**TRG_SEL6 [1:0]**	Ch.6 trigger select	TRG_SEL6[1:0] Trigger source 0x3 USB 0x2 USI Ch.2 Rx 0x1 USIL Rx 0x0 No hard trigger	0x0	R/W	
		D11–10	**TRG_SEL5 [1:0]**	Ch.5 trigger select	TRG_SEL5[1:0] Trigger source 0x3 T16A6 A 0x2 SD_MMC Tx 0x1 USI Ch.1 Tx 0x0 No hard trigger	0x0	R/W	
		D9–8	**TRG_SEL4 [1:0]**	Ch.4 trigger select	TRG_SEL4[1:0] Trigger source 0x3 T16A6 B 0x2 SD_MMC Rx 0x1 USI Ch.1 Rx 0x0 No hard trigger	0x0	R/W	
		D7–6	**TRG_SEL3 [1:0]**	Ch.3 trigger select	TRG_SEL3[1:0] Trigger source 0x3 T16A6 A 0x2 I^2S input R 0x1 USI Ch.0 Tx 0x0 No hard trigger	0x0	R/W	
		D5–4	**TRG_SEL2 [1:0]**	Ch.2 trigger select	TRG_SEL2[1:0] Trigger source 0x3 T16A6 B 0x2 I^2S input L 0x1 USI Ch.0 Rx 0x0 No hard trigger	0x0	R/W	
		D3–2	**TRG_SEL1 [1:0]**	Ch.1 trigger select	TRG_SEL1[1:0] Trigger source 0x3 Port 0x2 T16P Ch.1 0x1 I^2S output R 0x0 No hard trigger	0x0	R/W	
		D1–0	**TRG_SEL0 [1:0]**	Ch.0 trigger select	TRG_SEL0[1:0] Trigger source 0x3 ADC complete 0x2 T16P Ch.0 0x1 I^2S output L 0x0 No hard trigger	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
DMAC Trigger Flag Register (DMAC_TRG_FLG)	**0x302110** (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**TRG7**	Ch.7 software trigger/trigger status	1	(W) Soft trigger	0	(W) Ignored	0	R/W	
		D6	**TRG6**	Ch.6 software trigger/trigger status					0	R/W	
		D5	**TRG5**	Ch.5 software trigger/trigger status					0	R/W	
		D4	**TRG4**	Ch.4 software trigger/trigger status					0	R/W	
		D3	**TRG3**	Ch.3 software trigger/trigger status		(R) Triggered		(R) Not triggered	0	R/W	
		D2	**TRG2**	Ch.2 software trigger/trigger status					0	R/W	
		D1	**TRG1**	Ch.1 software trigger/trigger status					0	R/W	
		D0	**TRG0**	Ch.0 software trigger/trigger status					0	R/W	
DMAC End-of-Transfer Flag Register (DMAC_END_FLG)	**0x302114** (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**ENDF7**	Ch.7 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	Reset by writing 1.
		D6	**ENDF6**	Ch.6 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D5	**ENDF5**	Ch.5 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D4	**ENDF4**	Ch.4 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D3	**ENDF3**	Ch.3 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D2	**ENDF2**	Ch.2 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D1	**ENDF1**	Ch.1 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
		D0	**ENDF0**	Ch.0 end-of-transfer flag	1	Finished	0	Not finished	0	R/W	
DMAC Running Status Register (DMAC_RUN_STAT)	**0x302118** (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**RUN7**	Ch.7 running status	1	Running	0	Idle/paused	0	R	
		D6	**RUN6**	Ch.6 running status	1	Running	0	Idle/paused	0	R	
		D5	**RUN5**	Ch.5 running status	1	Running	0	Idle/paused	0	R	
		D4	**RUN4**	Ch.4 running status	1	Running	0	Idle/paused	0	R	
		D3	**RUN3**	Ch.3 running status	1	Running	0	Idle/paused	0	R	
		D2	**RUN2**	Ch.2 running status	1	Running	0	Idle/paused	0	R	
		D1	**RUN1**	Ch.1 running status	1	Running	0	Idle/paused	0	R	
		D0	**RUN0**	Ch.0 running status	1	Running	0	Idle/paused	0	R	
DMAC Pause Status Register (DMAC_PAUSE_STAT)	**0x30211c** (32 bits)	D31–8	–	reserved	–				–	–	0 when being read.
		D7	**PAUSE7**	Ch.7 pause status	1	Paused	0	Not paused	0	R	
		D6	**PAUSE6**	Ch.6 pause status	1	Paused	0	Not paused	0	R	
		D5	**PAUSE5**	Ch.5 pause status	1	Paused	0	Not paused	0	R	
		D4	**PAUSE4**	Ch.4 pause status	1	Paused	0	Not paused	0	R	
		D3	**PAUSE3**	Ch.3 pause status	1	Paused	0	Not paused	0	R	
		D2	**PAUSE2**	Ch.2 pause status	1	Paused	0	Not paused	0	R	
		D1	**PAUSE1**	Ch.1 pause status	1	Paused	0	Not paused	0	R	
		D0	**PAUSE0**	Ch.0 pause status	1	Paused	0	Not paused	0	R	

0x302200–0x302210 SDRAM Controller (SDRAMC)

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SDRAM Initialization Register (SDRAMC_INIT)	**0x302200** (32 bits)	D31–5	**–**	reserved	–				–	–	0 when being read.
		D4	**SDON**	SDRAM controller enable	1	Enable	0	Disable	0	R/W	
		D3	**INIDO**	SDRAM initialization status	1	Finished	0	Busy	0	R	
		D2	**INIMRS**	MRS command enable for init.	1	Enable	0	Disable	0	R/W	
		D1	**INIPRE**	PALL command enable for init.	1	Enable	0	Disable	0	R/W	
		D0	**INIREF**	REF command enable for init.	1	Enable	0	Disable	0	R/W	
SDRAM Configuration Register (SDRAMC_CFG)	**0x302204** (32 bits)	D31–14	**–**	reserved	–				–	–	0 when being read.
		D13–12	**T24NS[1:0]**	Number of SDRAM t_{RP} and t_{RCD} cycles	T24NS[1:0]	# of cycles			0x0	R/W	
					0x3	4 cycles					
					0x2	3 cycles					
					0x1	2 cycles					
					0x0	1 cycle					
		D11	**–**	reserved	–				–	–	0 when being read.
		D10–8	**T60NS[2:0]**	Number of SDRAM t_{RAS} cycles	T60NS[2:0]	# of cycles			0x0	R/W	
					0x7	8 cycles					
					0x6	7 cycles					
					:	:					
					0x1	2 cycles					
					0x0	1 cycle					
		D7–4	**T80NS[3:0]**	Number of SDRAM t_{RC}, t_{RFC} and t_{XSR} cycles	T80NS[3:0]	# of cycles			0xe	R/W	
					0xc	13 cycles					
					0x8	9 cycles					
					0x4	5 cycles					
					0x0	1 cycle					
					Other	reserved					
		D3	**–**	reserved	–				–	–	0 when being read.
		D2–0	**ADDRC[2:0]**	SDRAM address configuration	ADDRC[2:0]	Configuration			0x0	R/W	Do not set to 0x4.
					0x7	512M bits					
					0x6	128M bits x 2					
					0x5	64M bits x 2					
					0x4	reserved					
					0x3	256M bits					
					0x2	128M bits					
					0x1	64M bits					
					0x0	16M bits					

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SDRAM Refresh Control Register (SDRAMC_REF)	**0x302208** (32 bits)	D31–26	**–**	reserved	–	–	–	0 when being read.
		D25	**SREFDO**	SDRAM self-refresh status	1 Finished 0 Busy	0	R	
		D24	**SCKON**	SDRAM clock during self-refresh	1 Enable 0 Disable	0	R/W	
		D23	**SELEN**	SDRAM self-refresh enable	1 Enable 0 Disable	0	R/W	
		D22–16	**SELCO[6:0]**	SDRAM self-refresh counter	0x0 to 0x7f	0x7f	R/W	
		D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–0	**AURCO[11:0]**	SDRAM auto-refresh counter	0x0 to 0xfff	0x8c	R/W	
SDRAM Application Configuration Register (SDRAMC_APP)	**0x302210** (32 bits)	D31–6	**–**	reserved	–	–	–	0 when being read.
		D5	**–**	reserved	–	0	–	Do not set to 1.
		D4	**–**	reserved	–	–	–	0 when being read.
		D3–2	**CAS[1:0]**	CAS latency setup	CAS[1:0] / CAS latency 0x3 / 3 0x2 / 2 0x1 / 1 0x0 / 0	0x0	R/W	
		D1	**–**	reserved	–	–	–	0 when being read.
		D0	**–**	reserved	–	0	–	Do not set to 1.

0x302220–0x302228 SRAM Controller (SRAMC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
#CE[7:4] Access Timing Configuration Register (SRAMC_ TMG47)	**0x302220** (32 bits)	D31–30	**CE7SETUP [1:0]**	#CE7 setup cycle	CE7SETUP[1:0] / Setup cycle 0x3 / 4 cycles 0x2 / 3 cycles 0x1 / 2 cycles 0x0 / 1 cycle	0x3	R/W	
		D29–28	**CE7HOLD [1:0]**	#CE7 hold cycle	CE7HOLD[1:0] / Hold cycle 0x3 / 4 cycles 0x2 / 3 cycles 0x1 / 2 cycles 0x0 / 1 cycle	0x3	R/W	
		D27–24	**CE7WAIT [3:0]**	#CE7 static wait cycle	CE7WAIT[3:0] / Wait cycle 0xf / 15 cycles 0xe / 14 cycles : / : 0x1 / 1 cycle 0x0 / 0 cycles	0xf	R/W	
		D23–16	**–**	reserved	–	–	–	1 when being read.
		D15–14	**CE5SETUP [1:0]**	#CE5 setup cycle	CE5SETUP[1:0] / Setup cycle 0x3 / 4 cycles : / : 0x0 / 1 cycle	0x3	R/W	
		D13–12	**CE5HOLD [1:0]**	#CE5 hold cycle	CE5HOLD[1:0] / Hold cycle 0x3 / 4 cycles : / : 0x0 / 1 cycle	0x3	R/W	
		D11–8	**CE5WAIT [3:0]**	#CE5 static wait cycle	CE5WAIT[3:0] / Wait cycle 0xf / 15 cycles : / : 0x0 / 0 cycles	0xf	R/W	
		D7–6	**CE4SETUP [1:0]**	#CE4 setup cycle	CE4SETUP[1:0] / Setup cycle 0x3 / 4 cycles : / : 0x0 / 1 cycle	0x3	R/W	
		D5–4	**CE4HOLD [1:0]**	#CE4 hold cycle	CE4HOLD[1:0] / Hold cycle 0x3 / 4 cycles : / : 0x0 / 1 cycle	0x3	R/W	
		D3–0	**CE4WAIT [3:0]**	#CE4 static wait cycle	CE4WAIT[3:0] / Wait cycle 0xf / 15 cycles : / : 0x0 / 0 cycles	0xf	R/W	

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
#CE[10:8] Access Timing Configuration Register (SRAMC_ TMG810)	**0x302224** (32 bits)	D31–24	**–**	reserved	–		–	–	1 when being read.
		D23–22	**CE10SETUP [1:0]**	#CE10 setup cycle	CE10SETUP[1:0]	Setup cycle	0x3	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D21–20	**CE10HOLD [1:0]**	#CE10 hold cycle	CE10HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					0x2	3 cycles			
					0x1	2 cycles			
					0x0	1 cycle			
		D19–16	**CE10WAIT [3:0]**	#CE10 static wait cycle	CE10WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					0xe	14 cycles			
					:	:			
					0x1	1 cycle			
					0x0	0 cycles			
		D15–14	**CE9SETUP [1:0]**	#CE9 setup cycle	CE9SETUP[1:0]	Setup cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D13–12	**CE9HOLD [1:0]**	#CE9 hold cycle	CE9HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D11–8	**CE9WAIT [3:0]**	#CE9 static wait cycle	CE9WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					:	:			
					0x0	0 cycles			
		D7–6	**CE8SETUP [1:0]**	#CE8 setup cycle	CE8SETUP[1:0]	Setup cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D5–4	**CE8HOLD [1:0]**	#CE8 hold cycle	CE8HOLD[1:0]	Hold cycle	0x3	R/W	
					0x3	4 cycles			
					:	:			
					0x0	1 cycle			
		D3–0	**CE8WAIT [3:0]**	#CE8 static wait cycle	CE8WAIT[3:0]	Wait cycle	0xf	R/W	
					0xf	15 cycles			
					:	:			
					0x0	0 cycles			
#CE[10:4] Device Configuration Register (SRAMC_TYPE)	**0x302228** (32 bits)	D31–14	**–**	reserved	–		–	–	0 when being read.
		D13–12	**CE10TYPE [1:0]**	#CE10 device type	CExTYPE[1:0]	Device type	0x0	R/W	
					0x3–0x2	8-bit device			
		D11–10	**CE9TYPE [1:0]**	#CE9 device type	0x1	16-bit BSL type	0x0	R/W	
					0x0	16-bit A0 type			
		D9–8	**CE8TYPE [1:0]**	#CE8 device type			0x0	R/W	
		D7–6	**CE7TYPE [1:0]**	#CE7 device type			0x0	R/W	
		D5–4	**–**	reserved	–		–	–	0 when being read.
		D3–2	**CE5TYPE [1:0]**	#CE5 device type	CExTYPE[1:0]	Device type	0x0	R/W	
					0x3–0x2	8-bit device			
		D1–0	**CE4TYPE [1:0]**	#CE4 device type	0x1	16-bit BSL type	0x0	R/W	
					0x0	16-bit A0 type			

0x302300–0x302318 Cache Controller (CCU)

Register name	Address	Bit	Name	Function	Setting		Init.	R/W	Remarks
Cache Configuration Register (CCU_CFG)	**0x302300** (32 bits)	D31–9	–	reserved	–		–	–	0 when being read.
		D8	**WBEN**	Write buffer enable	1 Enable	0 Disable	1	R/W	
		D7–3	–	reserved	–		–	–	0 when being read.
		D2	**SBRK**	Software break enable	1 Enable	0 Disable	1	R/W	
		D1	**IC**	Instruction cache enable	1 Enable	0 Disable	0	R/W	
		D0	**DC**	Data cache enable	1 Enable	0 Disable	0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
Cacheable Area Select Register (CCU_AREA)	**0x302304** (32 bits)	D31–7	–	reserved	–	–	–	0 when being read.
		D6–4	**ARIC[2:0]**	Instruction cache area select	ARIC[2:0] Area 0x7 Area 22 0x6 Area 21 0x5 Area 20 0x4 Area 19 0x3 Area 18 0x2 Area 17 0x1 Areas 15 & 16 0x0 Area 14	0x0	R/W	
		D3	–	reserved	–	–	–	0 when being read.
		D2–0	**ARDC[2:0]**	Data cache area select	ARDC[2:0] Area 0x7 Area 22 0x6 Area 21 0x5 Area 20 0x4 Area 19 0x3 Area 18 0x2 Area 17 0x1 Areas 15 & 16 0x0 Area 14	0x0	R/W	
Cache Lock Register (CCU_LK)	**0x302308** (32 bits)	D31–8	–	reserved	–	–	–	0 when being read.
		D7	**LKPRI7**	Interrupt level 7 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D6	**LKPRI6**	Interrupt level 6 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D5	**LKPRI5**	Interrupt level 5 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D4	**LKPRI4**	Interrupt level 4 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D3	**LKPRI3**	Interrupt level 3 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D2	**LKPRI2**	Interrupt level 2 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D1	**LKPRI1**	Interrupt level 1 cache-lock enable	1 Lock 0 Unlock	0	R/W	
		D0	**LKPRI0**	Interrupt level 0 cache-lock enable	1 Lock 0 Unlock	0	R/W	
Cache Status Register (CCU_STAT)	**0x30230c** (32 bits)	D31–4	–	reserved	–	–	–	0 when being read.
		D3	**ICLKS**	Instruction cache lock status	1 Locked 0 Not locked	X	R	
		D2	**DCLKS**	Data cache lock status	1 Locked 0 Not locked	0	R	
		D1	**ICS**	Instruction cache operating status	1 Active 0 Inactive	0	R	
		D0	**DCS**	Data cache operating status	1 Active 0 Inactive	0	R	
Cache Write Buffer Status Register (CCU_WB_STAT)	**0x302318** (32 bits)	D31–10	–	reserved	–	–	–	X when being read.
		D9	**WEFINISH**	Write-finish flag	1 Finished 0 Writing	1	R	
		D8	**WBEMPTY**	Write buffer empty flag	1 Empty 0 Full	1	R	
		D7–0	–	reserved	–	–	–	X when being read.

0x302400–0x302432 I²S

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Ch.0 Control Register (I2S_CTL0)	**0x302400** (32 bits)	D31–11	**–**	reserved	–	–	–	0 when being read.
		D10	**DTSIGN0**	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	**DTFORM0**	I²S Ch.0 output data format select	1 LSB first 0 MSB first	0	R/W	
		D4	**I2SOUTEN0**	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 0x3 reserved 0x2 Right justified 0x1 Left justified 0x0 I²S	0x0	R/W	
		D1–0	**CHMD0[1:0]**	I²S Ch.0 output channel mode select	CHMD0[1:0] Channel mode 0x3 Mute 0x2 Mono left 0x1 Mono right 0x0 Stereo	0x0	R/W	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S Ch.1 Control Register (I2S_CTL1)	**0x302404** (32 bits)	D31–7	**–**	reserved	–	–	–	0 when being read.
		D6	**I2SBYPASS**	I²S bypass mode select	1 Bypass 0 Normal	0	R/W	
		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] Timing mode 0x3 reserved 0x2 Right justified 0x1 Left justified 0x0 I²S	0x0	R/W	
		D1	**DATRES1**	I²S Ch.1 input data resolution select	1 24 bits 0 16 bits	0	R/W	
		D0	**I2SEN1**	I²S Ch.1 enable	1 Enable 0 Disable	0	R/W	
I²S Master Clock Division Ratio Register (I2S_DV_MCLK)	**0x302408** (32 bits)	D31–16	**–**	reserved	–	–	–	0 when being read.
		D15	**MCLKSEL**	I2S_MCLK source clock select	1 External clk 0 System clk	0	R/W	
		D14–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**MCLKDIV [5:0]**	I2S_MCLK division ratio select	MCLKDIV[5:0] Division ratio 0x3f 1/64 0x3e 1/63 0x3d 1/62 : 0x2 1/3 0x1 1/2 0x0 reserved	0x0	R/W	Source clock = I2S_CLK (PCLK)
I²S Audio Clock Division Ratio Register (I2S_DV_BCLK)	**0x30240c** (32 bits)	D31–21	**–**	reserved	–	–	–	0 when being read.
		D20–16	**WSCLKCYC1 [4:0]**	I²S Ch.1 WS clock cycle setup	WSCLKCYC1[4:0] Clock period Other reserved 0x10 32 clocks 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	0x0	R/W	
		D15–13	**–**	reserved	–	–	–	0 when being read.
		D12–8	**WSCLKCYC0 [4:0]**	I²S Ch.0 WS clock cycle setup	WSCLKCYC0[4:0] Clock period Other reserved 0x10 32 clocks 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	0x0	R/W	
		D7–0	**BCLKDIV0 [7:0]**	I²S Ch.0 bit clock division ratio select	BCLKDIV0[7:0] Division ratio 0xff 1/512 0xfe 1/510 0xfd 1/508 : 0x2 1/6 0x1 1/4 0x0 1/2	0x0	R/W	Source clock = Clock selected by MCLKSEL
I²S Start/Stop Register (I2S_START)	**0x302410** (32 bits)	D31–9	**–**	reserved	–	–	–	0 when being read.
		D8	**I2SSTART1**	I²S Ch.1 start/stop control	1 Start (run) 0 Stop	0	R/W	
		D7	**I2SBUSY**	I²S 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	

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
I²S FIFO Status Register (I2S_STATUS)	**0x302414** (32 bits)	D31–10	**–**	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 Control Register (I2S_INT)	**0x302418** (32 bits)	D31–15	**–**	reserved	–	–	–	0 when being read.
		D14	**I2SWFIF1**	I²S Ch.1 FIFO whole full int. flag	1 Occurred 0 Not occurred	0	R/W	Reset by writing 1.
		D13	**–**	reserved	–	–	–	0 when being read.
		D12	**I2SOFIF1**	I²S Ch.1 FIFO one data int. flag	1 Occurred 0 Not occurred	0	R/W	Reset by writing 1.
		D11	**–**	reserved	–	–	–	0 when being read.
		D10	**I2SWEIF0**	I²S Ch.0 FIFO whole empty int. flag	1 Cause of interrupt occurred / 0 Cause of interrupt not occurred	0	R/W	Reset by writing 1.
		D9	**I2SHEIF0**	I²S Ch.0 FIFO half empty int. flag		0	R/W	
		D8	**I2SOEIF0**	I²S Ch.0 FIFO one empty int. flag		0	R/W	
		D7	**–**	reserved	–	–	–	0 when being read.
		D6	**I2SWFIE1**	I²S Ch.1 FIFO whole full int. enable	1 Enable 0 Disable	0	R/W	
		D5	**–**	reserved	–	–	–	0 when being read.
		D4	**I2SOFIE1**	I²S Ch.1 FIFO one data int. enable	1 Enable 0 Disable	0	R/W	
		D3	**–**	reserved	–	–	–	0 when being read.
		D2	**I2SWEIE0**	I²S Ch.0 FIFO whole empty int. enable	1 Enable 0 Disable	0	R/W	
		D1	**I2SHEIE0**	I²S Ch.0 FIFO half empty int. enable	1 Enable 0 Disable	0	R/W	
		D0	**I2SOEIE0**	I²S Ch.0 FIFO one empty int. enable	1 Enable 0 Disable	0	R/W	
I²S Mode Select Register (I2S_MODE)	**0x30241c** (32 bits)	D31–10	**–**	reserved	–	–	–	0 when being read.
		D9	**I2SDMA1**	I²S Ch.1 DMA mode select	1 Dual ch. 0 Single ch.	0	R/W	
		D8	**I2SDMA0**	I²S Ch.0 DMA mode select	1 Dual ch. 0 Single ch.	0	R/W	
		D7–1	**–**	reserved	–	–	–	0 when being read.
		D0	**I2SPIN**	I²S pin mode select	1 5 pins 0 8 pins	0	R/W	
I²S Ch.0 FIFO Register (I2S_FIFO0)	**0x302420** (32 bits)	D31–0	**I2SFIFO0 [31:0]**	I²S Ch.0 output FIFO	0x0 to 0xffffffff	0x0	W	for 16-bit data (W write) or 24-bit data (W write) 0 when being read.
	0x302420 (16 bits)	D15–0	**I2SFIFO0 [15:0]**	I²S Ch.0 output FIFO	0x0 to 0xffff	0x0	W	for 16-bit data (HW write)
	0x302422 (16 bits)	D15–0	**I2SFIFO0 [31:16]**	I²S Ch.0 output FIFO	0x0 to 0xffff	0x0	W	0 when being read.
I²S Ch.1 FIFO Register (I2S_FIFO1)	**0x302430** (32 bits)	D31–0	**I2SFIFO1 [31:0]**	I²S Ch.1 input FIFO	0x0 to 0xffffffff	0x0	R	for 16-bit data (W read) or 24-bit data (W read) 0 when being read.
	0x302430 (16 bits)	D15–0	**I2SFIFO1 [15:0]**	I²S Ch.1 input FIFO	0x0 to 0xffff	0x0	R	for 16-bit data (HW read)
	0x302432 (16 bits)	D15–0	**I2SFIFO1 [31:16]**	I²S Ch.1 input FIFO	0x0 to 0xffff	0x0	R	0 when being read.

0x302600–0x302650 SD/MMC Interface (SD_MMC)

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SD_MMC Clock Control Register (MMC_CLKCTL)	**0x302600** (32 bits)	D31–25	**–**	reserved	–	–	–	0 when being read.
		D24	**CLKSTF**	Clock change status flag	1 Changing 0 Completed	0	R	
		D23–17	**–**	reserved	–	–	–	0 when being read.
		D16	**CLKRUN**	MMC clock divider run/stop control	1 Run 0 Stop	0 *	R/W	
		D15–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**CLKDIV[7:0]**	MMC clock division ratio setting	$f_{PCLK} / 2 \times f_{SDMMCCLK}$ -1	0x0 *	R/W	

* Not initialized by SYNRST/MMC_FUNCTL register.

Register name	Address	Bit	Name	Function	Setting	Init.	R/W	Remarks
SD_MMC Function Control Register (MMC_FUNCTL)	**0x302604** (32 bits)	D31–17	**–**	reserved	–	–	–	0 when being read.
		D16	**SYNRST**	Synchronous reset enable	1 Enable 0 Disable	0	R/W	
		D15–14	**–**	reserved	–	–	–	0 when being read.
		D13	**8CLKEN**	8 clocks sending enable	1 Enable 0 Disable	0	R/W	
		D12	**BUSYEN**	Busy wait enable	1 Enable 0 Disable	0	R/W	
		D11	**DAREN**	Data reception enable	1 Enable 0 Disable	0	R/W	
		D10	**DATEN**	Data transmission enable	1 Enable 0 Disable	0	R/W	
		D9	**RSPEN**	Response reception enable	1 Enable 0 Disable	0	R/W	
		D8	**COMSED**	Command transmission enable	1 Enable 0 Disable	0	R/W	
		D7–4	**–**	reserved	–	–	–	0 when being read.
		D3	**DATBYT**	Transfer data byte order type select	1 D[31:24] first 0 D[7:0] first	0	R/W	
		D2	**DATMD**	Data transfer mode select	1 Multi block 0 Single block	0	R/W	
		D1	**RSPMD**	Response mode select	1 136-bit long 0 48-bit short	0	R/W	
		D0	**BUSSEL**	Wide bus mode enable	1 Wide 0 Standard	0	R/W	
SD_MMC Block Size Setting Register (MMC_BLK)	**0x302608** (32 bits)	D31–12	**–**	reserved	–	–	–	0 when being read.
		D11–0	BLKSIZ [11:0]	Block size setting	0x0–0xfff	0x0	R/W	
SD_MMC Data/ Busy Timeout Period Setting Register (MMC_TIMEOUT)	**0x30260c** (32 bits)	D31–24	**–**	reserved	–	–	–	0 when being read.
		D23–0	**TMOUT [23:0]**	Data/busy timeout period setting	0x0–0xffffff	0xff ffff	R/W	
SD_MMC Command Index Register (MMC_CMD_IDX)	**0x302610** (32 bits)	D31–6	**–**	reserved	–	–	–	0 when being read.
		D5–0	**CMDIDX [5:0]**	SD/MMC memory card command index	0x0–0x3f	0x0	R/W	
SD_MMC Command Argument Register (MMC_CMD _ARG)	**0x302614** (32 bits)	D31–0	**CMDARG [31:0]**	SD/MMC memory card command argument	0x0–0xffffffff	0x0	R/W	
SD_MMC Response Register 0 (MMC_RSP0)	**0x302618** (32 bits)	D31–0	**RSP[31:0]**	SD/MMC memory card response	0x0–0xffffffff	*	R	* 0xffffffff
SD_MMC Response Register 1 (MMC_RSP1)	**0x30261c** (32 bits)	D31–0	**RSP[63:32]**	SD/MMC memory card response	0x0–0xffffffff	*	R	* 0x0000ffff
SD_MMC Response Register 2 (MMC_RSP2)	**0x302620** (32 bits)	D31–0	**RSP[95:64]**	SD/MMC memory card response	0x0–0xffffffff	0x0	R	
SD_MMC Response Register 3 (MMC_RSP3)	**0x302624** (32 bits)	D31–0	**RSP[127:96]**	SD/MMC memory card response	0x0–0xffffffff	0x0	R	
SD_MMC Response Register 4 (MMC_RSP4)	**0x302628** (32 bits)	D31–8	**–**	reserved	–	–	–	0 when being read.
		D7–0	**RSP[135:128]**	SD/MMC memory card response	0x0–0xffffffff	0x0	R	
SD_MMC Status Register (MMC_STATUS)	**0x30262c** (32 bits)	D31–18	**–**	reserved	–	–	–	0 when being read.
		D17	**EMPTFLG**	Transmit FIFO empty flag	1 Empty 0 Not empty	1	R	
		D16	**FULLFLG**	Receive FIFO full flag	1 Full 0 Not full	0	R	
		D15–12	**–**	reserved	–	–	–	0 when being read.
		D11–9	**CRCST[2:0]**	Card CRC status	CRCST[2:0] Status 0x7 Flash prg. error 0x5 Transmit error 0x2 No error Other reserved	0x0	R	
		D8	**BUSY_ST**	Card busy status	1 Busy 0 Ready	0	R	
		D7	**RSPFLG**	Command with response flag	1 Completed 0 Not completed	0	R/W	Reset by writing 1.
		D6	**CMDFLG**	Command without response flag	1 Completed 0 Not completed	0	R/W	
		D5	**HEMPTFLG**	Transmit FIFO half-empty flag	1 Empty 0 Not empty	0	R/W	
		D4	**HFULLFLG**	Receive FIFO half-full flag	1 Full 0 Not full	0	R/W	
		D3	**RSPTMFLG**	Response timeout flag	1 Timeout 0 No error	0	R/W	
		D2	**DATTMFLG**	Data transfer timeout flag	1 Timeout 0 No error	0	R/W	
		D1	**RSPCRCFLG**	Response CRC error flag	1 CRC error 0 No error	0	R/W	
		D0	**DATCRCFLG**	Data CRC error/error token flag	1 CRC error 0 No error	0	R/W	

Register name	Address	Bit	Name	Function	Setting				Init.	R/W	Remarks
SD_MMC Interrupt Control Register (MMC_INT)	**0x302630** (32 bits)	D31–9	**–**	reserved	–				–	–	0 when being read.
		D8	**DMAEN**	FIFO half-empty/half-full DMA transfer enable	1	Enable	0	Disable	0 *	R/W	
		D7	**RSPINTEN**	Command with response interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D6	**CMDINTEN**	Command without response interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D5	**TFEINTEN**	Transmit FIFO half-mpty interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D4	**RFFINTEN**	Receive FIFO half-full interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D3	**RSPTMEN**	Response timeout interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D2	**DATTMEN**	Data transfer timeout interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D1	**RSPCRCEN**	Response CRC error interrupt enable	1	Enable	0	Disable	0 *	R/W	
		D0	**DATCRCEN**	Data CRC error/error token interrupt enable	1	Enable	0	Disable	0 *	R/W	
SD_MMC Data Transfer FIFO Register (MMC_DAT)	**0x302650** (32 bits)	D31–0	**FIFODAT [31:0]**	FIFO read/write data	0x0–0xffffffff				0x0	R/W	

* Not initialized by SYNRST/MMC_FUNCTL register.

Appendix B 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.

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. Therefore, all peripheral circuits (except the OSC1 oscillator circuit and RTC) stop operating.
  The CPU is reawaken from SLEEP mode by initial reset, RTC interrupt, #NMI signal, other interrupt from an external device (port/key input interrupt), or a forcible break from the debugger.

System clock

- Selecting the clock source (CMU module)
  Either OSC3, PLL, 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)
  Using OSC1 for the system clock and disabling the OSC3 oscillator circuit achieves more reduction of current consumed.
- Selecting a low system clock (CMU module)
  The CMU module provides a clock divider to set the system clock speed to 1/1 to 1/32 of the source clock (OSC3, PLL, OSC1). By running the S1C33L27 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 LCD controller and 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 and PSC modules)
  The peripheral clock supply can be disabled if the peripheral modules listed below can be placed in standby state.

Table B.1 Peripheral Modules and Operating Clocks

Clock	Clock enable bit	Peripheral modules
PCLK	T16A_CKE/CMU_CLKCTL0 register	• 16-bit PWM timer (T16A6 Ch.0–3) registers • Prescaler (PSC Ch.0) → T16A6 Ch.0–3
	NANDIF_CKE/CMU_CLKCTL1 register	Card interface (CARD)
	REMC_CKE/CMU_CLKCTL1 register	Remote controller (REMC)
	ADC_CKE/CMU_CLKCTL1 register	A/D converter (ADC10)
	PORT_CKE/CMU_CLKCTL1 register	I/O ports and port MUX (GPIO/PMUX)
	SDMMC_CKE/CMU_CLKCTL1 register	SD/MMC interface (SD_MMC)
	I2S_CKE/CMU_CLKCTL1 register	I^2S (I2S)
	ITC_CKE/CMU_CLKCTL1 register	Interrupt controller (ITC)
	DMAC_CKE/CMU_CLKCTL1 register	DMA controller (DMAC)
	HIF_CKE/CMU_CLKCTL2 register	Host Interface (HIF)

Clock	Clock enable bit	Peripheral modules
PCLK	LCDC_SAPB_CKE/CMU_CLKCTL2 register	LCD controller (LCDC) registers
	LCDC_AHB_CKE/CMU_CLKCTL2 register	LCD controller (LCDC) AHB interface and FIFO
	USBREG_CKE/CMU_CLKCTL2 register	USB function controller (USB) registers
	BCU_CKE/CMU_CLKCTL2 register	• SRAM controller (SRAMC) • SDRAM controller (SDRAMC)
	PSC1/2_CKE/CMU_CLKCTL3 register	• Prescaler (PSC Ch.1) → • T8F Ch.0 → USI Ch.0 • T8F Ch.2 → USI Ch.2 • T8F Ch.4 → ADC10 • ADC10 • Prescaler (PSC Ch.2) → • T8F Ch.1 → USI Ch.1 • T8F Ch.3 → USIL • T8F Ch.5 → UART • UART • REMC • T16P Ch.0–1 • GPIO
	T8F_CKE/CMU_CLKCTL3 register	Fine mode 8-bit timer (T8F Ch.0–5)
	UART_CKE/CMU_CLKCTL3 register	UART
	T16P_CKE/CMU_CLKCTL3 register	16-bit audio PWM timer (T16P Ch.0–1)
	USI_CKE/CMU_CLKCTL3 register	Universal serial interface (USI Ch.0–2)
	USIL_CKE/CMU_CLKCTL3 register	Universal serial interface with LCD interface (USIL)
	MISC_CKE/CMU_CLKCTL4 register	Misc registers (MISC)
	IVRAM_ARB_CKE/CMU_CLKCTL4 register	IVRAM arbiter (LCDC)
	A3RAM_CKE/CMU_CLKCTL4 register	DSTRAM
	RTC_SAPB_CKE/CMU_CLKCTL4 register	• Real-time clock (RTC) registers • BBRAM
	WDT_CKE/CMU_CLKCTL4 register	Watchdog timer (WDT)
TCLK	TCLK_CKE/CMU_CLKCTL0 register	• 16-bit PWM timer (T16A6 Ch.0–3) counters • Prescaler (PSC Ch.0) → T16A6 Ch.0–3
LCDC_CLK	LCDC_CKE/CMU_CLKCTL2 register	LCD controller (LCDC)
USB_CLK	USB_CKE/CMU_CLKCTL2 register	USB function controller (USB)
BCLK	BCLK_CKE/CMU_CLKCTL2 register	• IVRAM (Area 3) • DSTRAM (Area 3) • SRAM controller (SRAMC) bus interface and registers • SDRAM controller (SDRAMC) bus interface and registers • Clock management unit (CMU) registers • SAPB bus (Can be stopped in HALT mode.)

Table B.2 lists the clock control conditions and how to suspend/resume the CPU operation.

Table B.2 List of Clock Control Conditions

Current consumption	OSC1	OSC3/PLL	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 (only RTC is running)	halt instruction	1, 2
	Oscillating (System clock)	Stop	Stop	Run	halt instruction	1, 2, 3
	Oscillating (System clock)	Stop	Run	Run		
	Stop	Oscillating (System clock)	Stop	Run	halt instruction	1, 3
	Oscillating	Oscillating (System clock)	Stop	Run	halt instruction	1, 2, 3
High ↓	Oscillating	Oscillating (System clock)	Run (OSC3/1) Run (PLL/1)	Run		

Clearing HALT and SLEEP modes (CPU resuming methods)

1. Resuming by a port/key input interrupt, #RESET, #NMI, or a debug interrupt (issuing an ICD forcible break)
   The CPU resumes operating by occurrence of a cause of port/key input interrupt, #RESET, #NMI, or a forcible break from the debugger.
2. Resuming by the RTC
   The CPU resumes operating by occurrence of a cause of RTC interrupt.
3. Resuming by a peripheral interrupt
   The CPU resumes operating by occurrence of a cause of interrupt in a peripheral whose interrupt is enabled. 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 (LV_{DD}, HV_{DD}, AV_{DD}, $PLLV_{DD}$) off
  If the system uses separated the system power and $RTCV_{DD}$ power sources, it is possible to operate only the $RTCV_{DD}$ system circuits (RTC, OSC1, and BBRAM) with the system power 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 the "Real-Time Clock (RTC)" chapter 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 HV_{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.

Appendix C Mounting Precautions

This section describes various precautions for circuit board design and IC mounting.

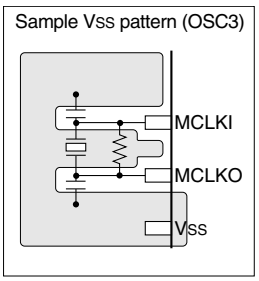
Oscillator circuit

- Oscillation characteristics depend on factors such as components used (resonator, R_f, C_G, C_D) and circuit board patterns. In particular, with ceramic or crystal resonators, select the appropriate external resistor (R_f) and capacitors (C_G, C_D) only after fully evaluating components actually mounted on the circuit board.
- Oscillator clock disturbances caused by noise may cause malfunctions. To prevent such disturbances, consider the following points. The latest devices, in particular, are manufactured by microscopic processes, making them especially susceptible to noise.
  Areas in which noise countermeasures are especially important include the RTCCLKO pin and related circuit components and wiring. RTCCLKI pin handling is equally important. The noise precautions required for the RTCCLKI and RTCCLKO pins are described below. We also recommend applying similar noise countermeasures to the high-speed oscillator circuit, such as the MCLKI and MCLKO pins and wiring.

(1) Components such as a resonator, resistors, and capacitors connected to the RTCCLKI (MCLKI) and RTCCLKO (MCLKO) pins should have the shortest connections possible.

(2) Wherever possible, avoid locating digital signal lines within 3 mm of the RTCCLKI (MCLKI) and RTCCLKO (MCLKO) pins or related circuit components and wiring. Rapidly-switching signals, in particular, should be kept at a distance from these components. Since the spacing between layers of multi-layer printed circuit boards is a mere 0.1 mm to 0.2 mm, the above precautions also apply when positioning digital signal lines on other layers.
Never place digital signal lines alongside such components or wiring, even if more than 3 mm distance or located on other layers. Avoid crossing wires.

(3) Use V_{SS} to shield RTCCLKI (MCLKI) and RTCCLKO (MCLKO) pins and related wiring (including wiring for adjacent circuit board layers). Layers wired should be adequately shielded as shown to the right. Fully ground adjacent layers, where possible. At minimum, shield the area at least 5 mm around the above pins and wiring.
Even after implementing these precautions, avoid configuring digital signal lines in parallel, as described in (2) above. Avoid crossing even on discrete layers, except for lines carrying signals with low switching frequencies.



(4) When an external clock is supplied to the RTCCLKI or MCLKI pin, the clock source should be connected to the RTCCLKI or MCLKI pin in the shortest line. Furthermore, do not connect anything else to the RTCCLKO or MCLKO pin.

(5) After implementing these precautions, check the output clock waveform by running the actual application program within the product. Use an oscilloscope to check outputs from the CMU_CLK pin.
You can check the quality of the OSC3 or PLL output waveform via the CMU_CLK output. Confirm that the frequency is as designed, is free of noise, and has minimal jitter.
You can also check the quality of the OSC1 waveform via the CMU_CLK output. In particular, enlarge the areas before and after the clock rising and falling edges and take special care to confirm that the regions approximately 100 ns to either side are free of clock or spiking noise.

Failure to observe precautions (1) to (3) adequately may lead to jitter in the OSC3 or PLL output and noise in the OSC1 output. Jitter in the OSC3 output will reduce operating frequencies, while noise in the OSC1 output will destabilize timers operated by the OSC1 clock as well as CPU Core operations when the system clock switches to OSC1.

Reset circuit

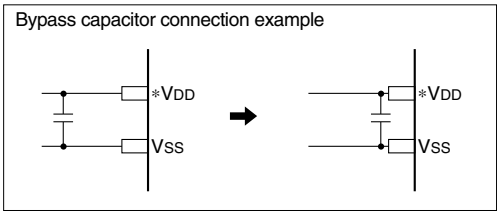
- The reset signal input to the #RESET pin when power is turned on will vary, depending on various factors, such as power supply start-up time, components used, and circuit board patterns. Constants such as capacitance and resistance should be determined through testing with real-world products.
- Components such as capacitors and resistors connected to the #RESET pin should have the shortest connections possible to prevent noise-induced resets.

Power supply circuit

Sudden power supply fluctuations due to noise will cause malfunctions. Consider the following issues.

(1) Connections from the power supply to the LVDD, HVDD, PLLVDD, AVDD, PLLVSS, and VSS pins should be implemented via the shortest, thickest patterns possible. In particular, the power supply for AVDD affects A/D conversion precision.

(2) If a bypass capacitor is connected between ∗VDD and VSS, connections between the ∗VDD and VSS pins should be as short as possible.

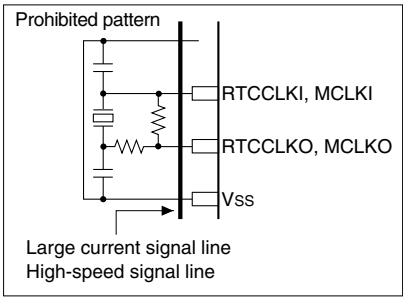


A/D Converter

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

Signal line location

- To prevent electromagnetically-induced noise arising from mutual induction, large-current signal lines should not be positioned close to circuits susceptible to noise, such as oscillators.
- Locating signal lines in parallel over significant distances or crossing signal lines operating at high speed will cause malfunctions due to noise generated by mutual interference. Specifically, avoid positioning crossing signal lines operating at high speed close to circuits susceptible to noise, such as oscillators and analog inputs.

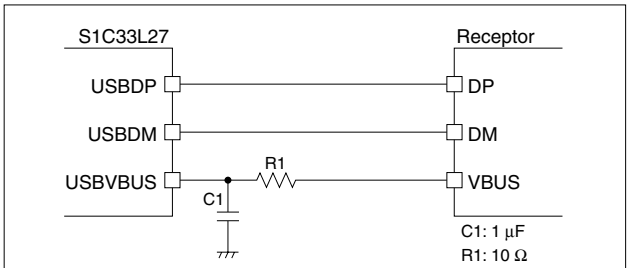


USB

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.



Pull down the USBVBUS, USBDP, and USBDM pins when the USB is not used.

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 malfunctions

Check the following five points if you suspect the presence of noise-induced IC malfunctions.

(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
Low-level noise to this pin will cause a switch to debug mode. The switch to debug mode can be confirmed by the clock output from DCLK and a High signal from the DST2 pin.
For the product version, we recommend connecting the DSIO pin directly to HV_{DD} or pulling up the DISO pin using a resistor not exceeding 10 kΩ. The IC includes an internal pull-up resistor. The resistor has a relatively high impedance of 100 kΩ to 500 kΩ and is not noise-resistant.

(3) #RESET pin
Low-level noise to this pin will reset the IC. Depending on the input waveform, the reset may not proceed correctly. This is more likely to occur if, due to circuit design choices, the impedance is high when the reset input is high.

(4) #NMI pin
Low-level noise to 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) *VDD and VSS power supply
The IC will malfunction at the instant when noise falling below the rated voltage is input. Incorporate countermeasures on the circuit board, including close patterns for circuit board power supply circuits, noise-filtering decoupling capacitors, and surge/noise prevention components on the power supply line.

Perform the inspections described above using an oscilloscope capable of observing waveforms of at least 200 MHz. It may not be possible to observe high-speed noise events with a low-speed oscilloscope.

If you detect potential noise-induced malfunctions while observing the waveform with an oscilloscope, recheck with a low-impedance (less than 1 kΩ) resistor connecting the relevant pin to GND or to the power supply. Malfunctions at that pin are likely if changes are visible, such as the malfunction disappearing, becoming less frequent, or the phenomena changing.

The TEST, DSIO, #RESET, and #NMI input circuits described above detect input signal edges and are susceptible to malfunctions induced by spike noise. This makes these digital signal pins the most susceptible to noise. To reduce potential noise, keep the following two points in mind when designing circuit boards:

(A) It is important to lower the signal-driving impedance, as described above. Connect pins to the power supply or GND, with impedance of 1 kΩ or less, preferably 0 Ω. The signal lines connected should be no longer than approximately 5 mm.

(B) Parallel routing of 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 the digital lines. 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).

Handling of light (for bare chip mounting)

The characteristics of semiconductor components can vary when exposed to light. ICs may malfunction or nonvolatile memory data may be corrupted if ICs are exposed to light.
Consider the following precautions for circuit boards and products in which this IC is mounted to prevent IC malfunctions attributable to light exposure.

(1) Design and mount the product so that the IC is shielded from light during use.

(2) Shield the IC from light during inspection processes.

(3) Shield the IC on the upper, underside, and side faces of the IC chip.

(4) Mount the IC chip within one week of opening the package. If the IC chip must be stored before mounting, take measures to ensure light shielding.

(5) Adequate evaluations are required to assess nonvolatile memory data retention characteristics before product delivery if the product is subjected to heat stress exceeding regular reflow conditions during mounting processes.

Miscellaneous

This product series is manufactured using microscopic processes.
Although it is designed to ensure basic IC reliability meeting EIAJ and MIL standards, minor variations over time may result in electrical damage arising from disturbances in the form of voltages exceeding the absolute maximum rating when mounting the product in addition to physical damage. The following factors can give rise to these variations:

(1) Electromagnetically-induced noise from industrial power supplies used in mounting reflow, reworking after mounting, and individual characteristic evaluation (testing) processes
(2) Electromagnetically-induced noise from a solder iron when soldering

In particular, during soldering, take care to ensure that the soldering iron GND (tip potential) has the same potential as the IC GND.

Appendix D Boot

D.1 Boot Mode

The S1C33L27 supports the six boot modes listed below.

- Large page NAND Flash boot (Large page: 2048 + 64, 4096 + 128 bytes/page)
- Small page NAND Flash boot (Small page: 512 + 16, 1024 + 32 bytes/page)
- Host interface boot
- NOR Flash/external ROM boot (Either 8 bits or 16 bits)
- SPI-EEPROM boot
- PC RS232C boot

The S1C33L27 boots up in the boot mode that can be selected with the BOOT[1:0] and #CE10 pin configuration at initial reset.

Table D.1.1 Boot Mode Setting

Pin configuration			Boot mode	Program execution start address
BOOT1	BOOT0	#CE10		
0	0	1 (Input)	Large page MLC/SLC NAND Flash (>1024 + 32 bytes/page)	The system loads MBR to A0RAM (address 0x0) and starts executing the code loaded.
		0 (Input)	Small page MLC/SLC NAND Flash (≤1024 + 32 bytes/page)	
0	1	Output	Host interface	The system loads MBR to the memory specified by the host device and starts executing the code loaded.
1	0	Output	NOR Flash/external ROM	The system starts executing from the address written at address 0x20000000.
1	1	1 (Input)	SPI-EEPROM	The system loads MBR to A0RAM (address 0x0) and starts executing the code loaded.
		0 (Input)	PC RS232C	

Notes: • The #CE10 pull-up resistor is disabled by default. When using the #CE10 pin as an input pin for NAND Flash, SPI-EEPROM or PC RS232C boot mode, connect an external pull-down resistor to the #CE10 pin to set the pin level to 0 or an external pull-up resistor to the #CE10 pin to set the pin level to 1.

• The BOOT2 pin exists in the PFBGA12U-180 package or chip model. This pin must be connected to VSS.

Revision History

Code No.	Page	Contents
411911900	All	New establishment
411911901	1-1	Overview (Old) The S1C33L27 is a 32-bit application specific RISC controller that delivers more performance ... ... design technology using 0.18 µm low power CMOS process to install these features. (New) The S1C33L27 is a 32-bit application specific RISC controller. It is suitable for applications that ... It provides an interface to communicate with a built-in RAM type LCD driver.
		Features: Technology (Old) Technology • 0.18 µm AL-4-layers mixed analog low power CMOS process technology (New) Deleted
		Features: CCU (Old) • 1K-byte instruction cache and 1K-byte data cache with a four-way associative frame structure (four frames/way, four lines/frame, four words/line) (New) • 1K-byte instruction cache and 1K-byte data cache that adopts a four-way associative method
	1-2	Features: SRAMC (Old) • Allows direct connection of SRAM, ROM, and Flash memories. • Supports little endian access. (New) • Allows connection of SRAM, ROM, and Flash memories. • Little endian access
		Features: SDRAMC (Old) • Supports SDRAM direct interface. ... • Arbitrates ownership of the external bus between the CPU, DMAC, HIF, and LCDC. (New) • Supports SDRAM interface. ... • Arbitrates external bus accesses by AHB-1 (CPU, DMAC, HIF) and AHB-2 (LCDC).
		Features: HIF (Old) • Provides semaphore registers that can be used for token passing and other communication protocol. (New) • Provides semaphore registers.
		Features: ITC (Old) • 34 maskable interrupts (including four software interrupts) (New) • 34 maskable interrupts (including four software exceptions)
	1-3	Features: T16P (Old) • Supports 8, 16, 22.05, 32, 44.1, and 48 kHz sampling rates. ... • Supports fine mode to control pulse widths with high accuracy. (New) • Can support 8, 16, 22.05, 32, 44.1, and 48 kHz sampling rates. ... • Provides fine mode to improve the precision of the pulse width.
		Features: RTC (Old) • Can generate periodic interrupts. (New) • Can generate clock interrupts.
		Features: USI (Old) • Three channels of multi-serial I/O that can be used as a UART, SPI, or I²C module (New) • Three channels of multi-serial interface that can be used as a UART, SPI, or I²C module
		Features: USI (SPI mode) (Old) - Receive data mask function is available (master mode only). (New) Deleted
	1-4	Features: USIL (Old) • Multi-serial I/O that can be used as a UART, SPI, I²C, or built-in RAM LCD interface module (New) • Multi-serial interface that can be used as a UART, SPI, I²C, or built-in RAM LCD interface module
		Features: USIL (UART mode) (Old) - Character length: 7 or 8 bits ... - Supports DMA transfer. (New) - Same features as USI
		Features: USIL (SPI mode) (Old) - Supports both master and slave modes. ... - Supports DMA transfer. (New) - Data length: 8 bits fixed - Other features are the same as USI.
		Features: USIL (I²C mode) (Old) - Supports both master (single master only) and slave modes. ... - Can generate start/stop, data transfer, ACK/NAK transfer, and overrun error interrupts. (New) - Same features as USI
		Features: USIL (LCD parallel interface mode) (Old) - Access timings can be controlled using T8F. The setup cycle (1 to 4), hold cycle (1 to 4), and wait cycle (1 to 16) are configurable. (New) Deleted

REVISION HISTORY

Code No.	Page	Contents
411911901	1-4	Features: UART (Old) • Supports IrDA 1.0 interface. (New) • Conforms to IrDA 1.0.
		Features: I2S (Old) • Operates as master that generates the bit clock, word-select signal, data and master clock. (New) Deleted
		Features: I2S (Old) • Can generate I²S FIFO empty interrupts ... and I²S FIFO full interrupts for the input channel. (New) • Can generate FIFO empty interrupts for the output channel (half empty, whole empty, or one empty) and FIFO full interrupts for the input channel (whole full or one data).
	1-5	Features: SD_MMC (Old) • Variable clock rate from 0 to 30 MHz. • Supports 4-bit (wide bus) and 1-bit interface. (New) • Variable clock rate up to 30 MHz. • Supports 4-bit (wide bus) and 1-bit SD bus interface.
		Features: LCDC (Old) • Supports generic panel resolutions up to VGA, such as 640 × 480 pixels (VGA) and 320 × 240 pixels (QVGA) (can be configured according to the panel used). (New) • Supports various panel resolutions, such as 640 × 480 pixels (VGA) and 320 × 240 pixels (QVGA) (can be configured according to the panel used).
		Features: ADC10 (Old) • 10-bit A/D converter with up to eight ... or four (128- or 144-pin package) analog input channels (New) • 10-bit successive approximation type A/D converter ... • Up to eight analog input channels (chip and PFBGA12U-180 package) • Up to four analog input channels (TQFP24/QFP20-144pin and TQFP15-128pin packages)
		Features: USB (Old) • Supports auto negotiation function. ... • Scratchable variable number of bulk end points (New) • Provides auto negotiation function. ... • Supports four general-purpose endpoints and endpoint 0 (control).
		Features: GPIO (Old) • Maximum 91 I/O ports and eight input ports are available (PFBGA12U-180 package). (New) • Maximum 91 I/O ports and eight input ports are available (chip and PFBGA12U-180 package).
	1-6	Features: Operating voltage (Old) • LV_{DD} (core/internal logic power voltage) (New) • LV_{DD} (internal logic/internal memory power voltage)
		Features: Operating voltage (Old) ... or 1.7 V to 1.9 V (1.8 V typ.) when the USB module is used or when a ceramic resonator is used. (New) ... or 1.7 V to 1.9 V (1.8 V typ.) when a ceramic resonator is used.
	1-7	Block diagram Modified Figure 1.2.1
	1-14 to 1-22	Pin functions (Old) Pad (New) Chip
	1-14	Pin functions: List of power supply pins (LV_{DD}) (Old) Core power supply pin ... (1.7 to 1.9 V when the USB module is used or when a ceramic resonator is used.) (New) Internal logic/internal memory power supply pin ... (1.7 to 1.9 V when a ceramic resonator is used.)
	1-23	USI/USIL pin configuration: USI_DO*x*, USIL_DO Modified Tables 1.3.2.8 and 1.3.2.9
	2-1	CPU (Old) Based on the C33 STD Core CPU features, some useful C33 ADV Core functions/instructions were ... the software assets that the user has accumulated in the past can be effectively utilized. (New) Deleted
	2-5	CPU: Debug mode (Instruction break) (Old) An instruction break can be set at three addresses. (New) An instruction break can be set at two addresses.
	3-2	Memory map: Internal area map Modified Figure 3.2
	4-1	Power supply: Power supply pins Modified Table 4.1.1
		Power supply: Power supply system Modified Figure 4.1.1
	4-2	Power supply: Operating voltage (LV_{DD}) (Old) The CPU core and internal logic circuits operate with a voltage supplied between the LV_{DD} ... or 1.7 to 1.9 V when the USB module is used or when a ceramic resonator is used. (New) The internal logic and internal memory circuits operate with a voltage supplied between the LV_{DD} ... or 1.7 to 1.9 V when a ceramic resonator is used.

Code No.	Page	Contents
411911901	4-3	Power supply: Power-on sequence Modified Figure 4.7.1
		Power supply: Power-on sequence (Old) (2) t_{PR}: Power-on-reset time ... See "Electrical Characteristics" for the power-on-reset time. (New) (2) t_{STA3}: Time at which OSC3 oscillation starts (3) t_{RST}: Minimum reset pulse width ... Keep the #RESET signal low.
		Power supply: Power-off sequence (Old) Power-off: ... 3. LV_{DD}, $PLLV_{DD}$ (and $RTCV_{DD}$) (May be turned off with 1 above at the same time.) Notes: • Applying only HV_{DD} makes a diode circuit on the path from HV_{DD} to AV_{DD} ... • Be sure to avoid applying AV_{DD} ... degraded due to flow-through current of the AV_{DD}. (New) Power-off: ... 3. LV_{DD}, $PLLV_{DD}$ (and $RTCV_{DD}$) (May be turned off with 2 above at the same time.) Note: Be sure to avoid applying HV_{DD} or AV_{DD} ... due to flow-through current of the HV_{DD} or AV_{DD}.
	5-1	Reset and NMI: #RESET pin (Old) Also the internal reset signal is negated when the default OSC3 oscillation stabilization wait time has elapsed after the #RESET pin goes high. (New) Also the internal reset signal is negated when the default OSC3 oscillation stabilization wait cycle has elapsed after the #RESET pin goes high. Modified Figure 5.1.1.1
	5-2	Reset and NMI: Initial reset sequence (Old) Even if the #RESET pin input negates the reset signal after power is turned on, the CPU cannot boot up until the oscillation stabilization waiting time (128 / OSC3 clock frequency) has elapsed. ... Note: The oscillation stabilization time described in this section ... after power is turned on or SLEEP mode is canceled may be longer than that indicated in the figure below. (New) Even if the #RESET pin input negates the reset signal after power is turned on, the CPU cannot boot up until the oscillation stabilization waiting cycle (128 OSC3 clock cycles) has elapsed. ... Note: The oscillation stabilization cycle described in this section ... after power is turned on may be longer than that indicated in the figure below. Modified Figure 5.1.3.1
		Reset and NMI: Initial reset status Modified Table 5.1.4.1
	5-3	Reset and NMI: Precautions to be taken during initial reset (Old) Low-speed (OSC1) oscillator circuit ... The low-speed (OSC1) oscillator circuit requires a longer time for oscillation to stabilize ... the OSC1 clock should not be used only after this stabilization time elapsed. (New) Low-speed (OSC1) oscillator circuit ... The low-speed (OSC1) oscillator circuit requires a longer time for starting oscillation than ... the OSC1 clock should not be used only after this start time elapsed.
	5-4	Reset and NMI: #NMI pin (Old) To generate an NMI (Non-Maskable Interrupt) to the C33 PE Core, the following two conditions ... (2) The #NMI pin is maintained at a low level for three or more system clock cycles. (New) To generate an NMI (Non-Maskable Interrupt) to the C33 PE Core, the #NMI pin must be maintained at a low level for three or more system clock cycles.
	6-3, 6-16	CMU: Stabilization wait time at start of OSC3 oscillation (Old) Example: When OSC3 oscillation start time (max.) = 10 ms and f_{SYSCLK} = 48 MHz OSC3 oscillation stabilization wait time ≥ 480,000 [cycles] OSC3WT[3:0] should be set to 0x3 (OSC3 oscillation stabilization wait time = 524,288 cycles). (New) Example: When OSC3 oscillation start time (max.) = 25 ms and f_{SYSCLK} = 48 MHz OSC3 oscillation stabilization wait time ≥ 1,200,000 [cycles] OSC3WT[3:0] should be set to 0x1 (OSC3 oscillation stabilization wait time = 2,097,152 cycles).
	6-4	CMU: PLL on/off control (Old) Note: Immediately after the PLL is started ... the system can be switched over to the PLL. (New) Notes: • An output stabilization time is required (see "Electric Characteristics") immediately after ... • Switch the system clock source to OSC3 or OSC1 before turning the PLL off.
	6-7	CMU: Power supply for PLL (Old) In order to prevent undesirable effects of noise, the $PLLV_{DD}$ and $PLLV_{SS}$ pins are provided, in addition to the core power supply, to feed power to the PLL. (New) In order to prevent undesirable effects of noise, the $PLLV_{DD}$ and $PLLV_{SS}$ pins are provided, in addition to the internal logic/internal memory power supply, to feed power to the PLL.
	6-9	CMU: System clock source selection (Old) (3/4/1) Stop the peripheral circuits being currently operated (except the RTC). (New) (3/4/1) Stop the peripheral circuit operations (except the RTC).

REVISION HISTORY

Code No.	Page	Contents
411911901	6-9	CMU: System clock source selection (Old) Switching the system clock to PLL from OSC3 1. Configure the PLL input clock and the PLL parameters such as the multiplication rate before activating the PLL. (New) Switching the system clock to PLL from OSC3 1. Configure the PLL parameters, such as the PLL input clock and the multiplication rate, before activating the PLL.
	6-10	CMU: System clock source selection (Old) Note: Do not select the system clock ... mechanism against such system clock selection. (New) Notes: • Do not select the system clock ... mechanism against such system clock selection. • After the system clock is switched, ... settings according to the new clock as necessary.
		CMU: Clock supply control (Old) No description (New) The clock supply circuits other than core clock (CCLK) and bus clock (BCLK) provide a clock ... (when the slp instruction is executed), the clocks stop even if the clock enable bit is set to 1.
	6-12	CMU: Peripheral module clocks (PCLK) (Old) The clock is supplied even in HALT mode when the clock enable bit is set to 1. ... (when the slp instruction is executed), these clocks stop even if the clock enable bit is set to 1. (New) Deleted
		CMU: 16-bit PWM timer counter clock (TCLK) (Old) In HALT mode, TCLK does not stop if TCLK_CKE is set to 1. ... In SLEEP mode (when the slp instruction is executed), TCLK stops even if TCLK_CKE is set to 1. (New) Deleted
	6-13	CMU: LCDC clock (LCDC_CLK) (Old) In HALT mode, LCDC_CLK does not stop if LCDC_CKE is set to 1. ... (when the slp instruction is executed), LCDC_CLK stops even if LCDC_CKE is set to 1. (New) Deleted
		CMU: USB clock (USB_CLK) (Old) USB_CLK is supplied even in HALT mode when USB_CKE is set to 1. ... In SLEEP mode (when the slp instruction is executed), USB_CLK stops even if USB_CKE is set to 1. (New) Deleted
	6-14	CMU: SLEEP mode (Old) Notes: • In SLEEP mode, there is a time lag ... #NMI signal is negated before the clock is supplied. • Before setting the S1C33L27 into SLEEP mode, ... must be disabled. (New) Notes: • In SLEEP mode, there is a time lag ... #NMI signal is negated before the clock is supplied. • Before setting the S1C33L27 into SLEEP mode, ... must be disabled. • If the system clock is PLL, change it to OSC3 or OSC1 before setting SLEEP mode.
	8-1	RTC: RTC module overview (Old) • Periodic interrupts are possible. (New) • Clock interrupts are possible.
	8-9	RTC: WAKEUP and #STBY pins (Old) Note that leakage currents flow from the RTCV$_{DD}$ system ... the #STBY pin is set to a high level. (New) Note that leakage current flows to the RTCV$_{DD}$ system ... the #STBY pin is set to a high level.
	9-1	SRAMC: SRAMC overview (Old) • Supports little endian access. (New) • Little endian access
	10-1	SDRAMC: SDRAMC overview (Old) The SDRAM controller (SDRAMC) allows ... SDRAM to be connected directly to Areas 7 and 19. (New) The SDRAM controller (SDRAMC) allows up to 64MB of SDRAM to be connected to Areas 7 and 19.
	10-12, 10-14	SDRAMC: Self-refresh (Old) No description (New) Note: Be sure to avoid setting SDON ... SDRAMC is not in self-refresh mode.
	11-6, 11-23	HIF: Setting the address and selecting address control mode (Old) Notes: • ... • If an access attempt is made ... an invalid write or invalid read (reading of 0) will occur. (New) Notes: • ... • If an access attempt is made ... an invalid write or invalid read will occur.
	11-14	HIF: Timing chart Modified Figure 11.5.1
	12-3, 12-8	CCU: Selecting area to be cached Modified Tables 12.3.2.1 and 12.7.2
	14-7, 14-13, 14-14	DMAC: DMAC trigger source (Old) No description (New) Modified Tables 14.4.1 and 14.7.2 ∗ Set the T16A6 channel for invoking the DMAC using DMASEL[1:0]/T16A_CTL*x* register.
	16-3, 16-8	T16A6: Interrupts in comparator mode (Old) Note: The intervals of the compare A ... Otherwise, the second interrupt will be omitted by T16A6. (New) Note: The compare A data register (T16A_CCA*x*) ... no cause of interrupt will occur.

Code No.	Page	Contents
411911901	16-6, 16-14	T16A6: Reading counter values (Old) Note: T16SEL[1:0]/T16A_CTL*x* register ... counter value will be read from the T16A_TC0 register. (New) Note: The counter value must be read from ... selected using T16SEL[1:0]/T16A_CTL*x* register.
	16-13	T16A6: T16A6 Ch.*x* Counter Control Registers (T16A_CTL*x*) - (D[5:4]) T16SEL[1:0]: Counter Select Bits (Old) From the T16A_TC*x* register, values of the counter channel selected by T16SEL[1:0] are read out. (New) The counter value must be read from the T16A_TC*x* register of the channel selected using T16SEL[1:0].
	17-4, 17-14	T16P: Split mode (SPLTMD[1:0]) (Old) When a split mode is selected, the split high-order bits (10, 9, or 8 high-order bits) ... compared with the counter data and the two comparison results generate two PWM output signals. (New) When a split mode is selected, the high-order bits (10, 9, or 8 high-order bits) ... compared with the counter data and the two comparison results generate two PWM output signals.
	19-1	UART: UART module overview (Old) No description (New) • Supports full-duplex communication.
	19-3	UART: Enabling data transfers (Old) Note: Do not set RXEN to 0 while the UART is sending or receiving data. (New) Notes: • Do not set RXEN to 0 while the UART is sending or receiving data. • Data transfer only for transmission or reception cannot be enabled.
		UART: Data transmission control (Old) The transmit data buffer size is 1 byte, but a shift register is provided separately to allow data to be written while the previous data is being sent. (New) The transmit data buffer size is 1 byte, but a transmit data buffer and a shift register are provided separately to allow data to be written while the previous data is being sent.
	19-4	UART: Data reception control (Old) The receiver circuit is activated by setting RXEN to 1, enabling data to be received from an external serial device. (New) Deleted
	19-5	UART: Disabling data transfers (Old) Setting RXEN to 0 empties the transmit and receive data buffers, clearing any remaining data. (New) Setting RXEN to 0 empties the transmit data buffer, clearing any remaining data. The receive data buffer is not cleared.
		UART: Overrun error (Old) However, if the receive data buffer is not emptied ... and generate an overrun error. (New) However, if the receive data buffer is not emptied ... and an overrun error will occur.
	19-5, 19-9	UART: Overrun error (OER) (Old) No description (New) Furthermore, set RXEN to 0 and read the receive data buffer successively until the RDRY flag is cleared.
	19-12	UART: UART Control Registers (UART_CTL) - (D0) RXEN: UART Enable Bit (Old) Disabling transfers by writing 0 to RXEN also clears transmit/receive data buffers. (New) Disabling transfers by writing 0 to RXEN also clears transmit data buffer. The receive data buffer is not cleared.
	20-1	USI: USI module overview - SPI master/slave mode (Old) - Receive data mask function is available (master mode only). (New) Deleted
	20-2	USI: List of USI pins (Table 20.2.1) (Old) ∗2: After the pin functions are switched for USI, the pins are configured for input until the interface ... when the pins are already configured for USI, all USI pins are configured for input. (New) ∗2: After the pin functions are switched for USI, the pins are configured for input until the interface ... while USI is operating, switch the pin functions to general-purpose input/output ports as necessary.
	20-5	USI: Settings for SPI mode (Old) When used in SPI master mode, ... and enable or disable the receive data mask function. (New) When used in SPI master mode, select the clock mode and data length.
	20-6	USI: Receive data mask function (Old) Receive data mask function (master mode) The USI in SPI master mode provides a receive data mask (data retransmission) function. ... ... For normal data transfer, set SMSKEN to 0 (default) to disable the receive data mask function. (New) Deleted
	20-7	USI: Data transfer in UART mode - Data reception (Old) If the subsequent receive data is written to the receive data buffer when URDIF is 1, an overrun error occurs. (New) If the next reception is completed ... an overrun error occurs (at the time stop bit has been received).
	20-8	USI: Data receiving timing chart (UART mode) Modified Figure 20.5.1.2
		USI: Data transmission timing chart (SPI mode) Modified Figure 20.5.2.1

REVISION HISTORY

Code No.	Page	Contents
411911901	20-9	USI: Data transfer in SPI mode - Data reception (Old) If the subsequent receive data is written to the receive data buffer when SRDIF is 1, an overrun error occurs. (New) While SRDIF is set to 1, ... overrun error occurs at the time the first bit of the third byte is fetched).
		USI: Data receiving timing chart (SPI mode) Modified Figure 20.5.2.2
	20-10	USI: I²C mode connection example Added Figure 20.5.3.1
	20-11	USI: I²C master data transmission timing chart Modified Figure 20.5.3.3
	20-12	USI: Sending slave address and transfer direction bit (Old) ... In 10-bit mode, data is sent twice under software control. ... To send a 10-bit address, execute this procedure twice as shown in Figure 20.5.3.4. ... (New) ... In 10-bit mode, data is sent twice or three times under software control. ... Modified Figure 20.5.3.5 ... To send a 10-bit address, execute this procedure twice or three times as shown in Figure 20.5.3.5. ...
	20-14	USI: I²C master data receiving timing chart Modified Figure 20.5.3.10
	20-16	USI: Control method in I²C slave mode (Old) ... After an interrupt occurs, ... (ISSTA[2:0]/USI_ISIF*x* register) to check the operation finished. (New) ... After an interrupt occurs, ... This also automatically reset ISSTA[2:0] to 0x0.
	20-16, 20-17	USI: I²C slave data transmission timing chart Modified Figure 20.5.3.13
	20-18, 20-19	USI: I²C slave data receiving timing chart Modified Figure 20.5.3.15
	20-20	USI: Receive errors - Overrun error (Old) Overrun error (all interface modes) If data is received before the previously received data in the receive data buffer has not been read, ... The overrun error flag is reset to 0 by writing 1. (New) Overrun error (all interface modes) UART mode An overrun error occurs if the next reception is completed when URDIF is 1 and the receive data ... I²C master/slave mode ... To reset an overrun error, write 1 to IMEIF/ISEIF and then read the receive data buffer (USI_RD*x* register) twice.
	20-21	USI: Interrupts in UART mode - Receive error interrupt (Old) ... If any of the error flags has the value 1, ... proceed with error recovery. (New) ... If any of the error flags has the value 1, ... proceed with error recovery. To reset an overrun error, ... USIMOD[2:0]/USI_GCFG*x* register) to initialize USI.
	20-22	USI: Interrupts in SPI mode - Receive error interrupt (Old) ... If SEIF is 1, the interrupt handler routine will proceed with error recovery. (New) ... If SEIF is 1, the interrupt handler routine will proceed with error recovery. To reset an overrun error, clear SEIF ... then read the receive data buffer (USI_RD*x* register) twice.
	20-23	USI: Interrupts in I²C master/slave mode - Receive error interrupt (Old) To use this interrupt, ... interrupt requests for this cause will not be sent to the ITC. ... ... If IMEIF is 1, the interrupt handler routine will proceed with error recovery. (New) To use this interrupt, ... interrupt requests for this cause will not be sent to the ITC. An overrun error occurs ... two-byte data has been received without reading the receive data buffer. The USI module sets ... the interrupt handler routine will proceed with error recovery. To reset an overrun error, ... read the receive data buffer (USI_RD*x* register) twice.
	20-24, 20-32, AP-A-3, AP-A-4, AP-A-31, AP-A-32, AP-A-37	USI: 0x30045f/55f/75f USI Ch.*x* SPI Master Mode Receive Data Mask Registers (USI_SMSK*x*) (New) Deleted
	20-26	USI: USI Ch.*x* Receive Data Buffer Registers (USI_RD*x*) - (D[7:0]) RD[7:0]: USI Receive Data Buffer Bits (Old) If receiving the subsequent data is completed ... the new received data overwrites the contents. (New) Deleted
	20-28	USI: USI Ch.*x* UART Mode Interrupt Flag Registers (USI_UIF*x*) - (D2) UOEIF: Overrun Error Flag Bit (Old) An overrun error occurs when the previous received data ... UOEIF is reset by writing 1. (New) An overrun error occurs ... (write 0x0 to USIMOD[2:0]/USI_GCFG*x* register) to initialize USI.
	20-29, 20-30, AP-A-30, AP-A-32, AP-A-36	USI: USI Ch.*x* SPI Master/Slave Mode Configuration Registers (USI_SCFG*x*) (Old) D1 SMSKEN: Receive Data Mask Enable Bit (New) D1 Reserved (Do not set to 1.)

Code No.	Page	Contents
411911901	20-32	USI: USI Ch.*x* SPI Master/Slave Mode Interrupt Flag Registers (USI_SIF*x*) - (D2) SEIF: Overrun Error Flag Bit (Old) An overrun error occurs when the previous received data ... SEIF is reset by writing 1. (New) An overrun error occurs if data are received successively ... USI_RD*x* register twice can be reversed.
	20-34	USI: USI Ch.*x* I²C Master Mode Interrupt Flag Registers (USI_IMIF*x*) - (D[4:2]) IMSTA[2:0]: I²C Master Status Bits (Old) When an operation completion interrupt occurs, ... the operation that has been finished. (New)... the operation that has been finished. IMSTA[2:0] is automatically reset to 0x0 by writing 1 to IMIF.
		USI: USI Ch.*x* I²C Master Mode Interrupt Flag Registers (USI_IMIF*x*) - (D1) IMEIF: Overrun Error Flag Bit (Old) An overrun error occurs when the previous received data ... IMEIF is reset by writing 1. (New) An overrun error occurs ... and then read the receive data buffer (USI_RD*x* register) twice.
	20-36	USI: USI Ch.*x* I²C Slave Mode Interrupt Flag Registers (USI_ISIF*x*) - (D[4:2]) ISSTA[2:0]: I²C Slave Status Bits (Old) When an operation completion interrupt occurs, ... the operation that has been finished. (New)... the operation that has been finished. ISSTA[2:0] is automatically reset to 0x0 by writing 1 to ISIF.
	20-37	USI: USI Ch.*x* I²C Slave Mode Interrupt Flag Registers (USI_ISIF*x*) - (D1) ISEIF: Overrun Error Flag Bit (Old) An overrun errors occurs when the previous received data ... ISEIF is reset by writing 1. (New) An overrun error occurs ... and then read the receive data buffer (USI_RD*x* register) twice.
	21-1	USIL: USIL module overview - SPI master/slave mode (Old) - Receive data mask function is available (master mode only). (New) Deleted
	21-2, 21-3	USIL: List of USIL pins (Old) Table 21.2.1 ∗ When USIL is configured to I²C master or slave mode, ... data input/output pin simultaneously. (New) Modified Table 21.2.1 ∗1: When USIL is configured to I²C master or slave mode, ... data input/output pin simultaneously. ∗2: After the pin functions are switched for USIL, ... general-purpose input/output ports as necessary.
	21-3	USIL: Transfer clock (Old) When the USIL is configured to a UART, SPI master (normal mode), I²C master, or LCD SPI, ... When the USIL is configured to an SPI master (fast mode) or LCD parallel interface, PCLK is used as the source clock. (New) When the USIL is configured to a UART, SPI master (normal mode), I²C master, LCD SPI, or LCD parallel interface, the source clock for transfer is supplied by the fine mode 8-bit timer (T8F Ch.3). ... When the USIL is configured to an SPI master (fast mode), PCLK is used as the source clock.
	21-6	USIL: Settings for SPI mode (Old) When used in SPI master mode, select the clock mode and enable or disable the receive data mask function. (New) When used in SPI master mode, select the clock mode.
	21-7	USIL: Receive data mask function (Old) Receive data mask function (master mode) The USIL in SPI master mode provides a receive data mask (data retransmission) function. ... ... For normal data transfer, set SMSKEN to 0 (default) to disable the receive data mask function. (New) Deleted
	21-11	USIL: Data transfer in UART mode - Data reception (Old) If the subsequent receive data is written to the receive data buffer when URDIF is 1, an overrun error occurs. (New) If the next reception is completed ... an overrun error occurs (at the time stop bit has been received).
		USIL: Data receiving timing chart (UART mode) Modified Figure 21.5.1.2
	21-12	USIL: Data transmission timing chart (SPI mode) Modified Figure 21.5.2.1
		USIL: Data transfer in SPI mode - Data reception (Old) If the subsequent receive data is written to the receive data buffer when SRDIF is 1, an overrun error occurs. (New) While SRDIF is set to 1, ... overrun error occurs at the time the first bit of the third byte is fetched).
	21-13	USIL: Data receiving timing chart (SPI mode) Modified Figure 21.5.2.2
		USI: I²C mode connection example Added Figure 21.5.3.1
	21-15	USIL: I²C master data transmission timing chart Modified Figure 21.5.3.3
	21-15, 21-16	USIL: Sending slave address and transfer direction bit (Old) ... In 10-bit mode, data is sent twice under software control. ... To send a 10-bit address, execute this procedure twice as shown in Figure 21.5.3.4. ... (New) ... In 10-bit mode, data is sent twice or three times under software control. ... Modified Figure 21.5.3.5 ... To send a 10-bit address, execute this procedure twice or three times as shown in Figure 21.5.3.5. ...

REVISION HISTORY

Code No.	Page	Contents
411911901	21-18	USIL: I²C master data receiving timing chart Modified Figure 21.5.3.10
	21-20	USIL: Control method in I²C slave mode (Old) ... After an interrupt occurs, ... (ISSTA[2:0]/USIL_ISIF register) to check the operation finished. (New) ... After an interrupt occurs, ... This also automatically reset ISSTA[2:0] to 0x0.
	21-20, 21-21	USIL: I²C slave data transmission timing chart Modified Figure 21.5.3.13
	21-22, 21-23	USIL: I²C slave data receiving timing chart Modified Figure 21.5.3.15
	21-24	USIL: Data Transmission in LCD SPI Mode (Old) The LSBSY flag indicates the USIL status ... the LCD SPI controller is operating or at standby. (New) The LSBSY flag indicates the USIL status ... LSDMOD[1:0]/USIL_LSDCFG register has completed.
		USIL: Figure 21.5.4.1 (Old) Data Transmission Timing Chart (LCD SPI mode) (New) Data Transmission Timing Chart (LCD SPI mode, 16-bit data format)
		USIL: Data Transfer in LCD Parallel Mode - Data write (Old) To write data to the LCD driver/panel ... the command bit status (LPCMD/USIL_LPCFG register). The LCD parallel interface asserts the chip enable and write signals and outputs the buffer data ... The command bit status is output from the USIL_DI pin. The transmitter circuit includes ... (New) To write data to the LCD driver/panel ... the command bit status (LPCMD/USIL_LPCFG register). The command bit must be set before ... the USIL_DI pin immediately after it is written to the register. The LCD parallel interface asserts the chip enable signal and outputs the buffer data ... The transmitter circuit includes ...
		USIL: Data Transfer in LCD Parallel Mode - Data write (Old) This flag is set to 1 ... when data written to the buffer is output via the USIL_LCD_D[7:0] pins. (New) This flag is set to 1 ... when data written to the buffer is transferred to the shift register.
	21-25	USIL: Data-Write Timing Chart (LCD parallel mode) Modified Figure 21.5.5.1
	21-25, 21-49	USIL: Data Transfer in LCD Parallel Mode - Data read (Old) The LCD parallel interface asserts the chip enable and read signals ... via the USIL_DI pin. (New) Set the command bit ... output from the USIL_DI pin immediately after it is written to the register.
	21-27, 21-28	USIL: Receive errors - Overrun error (Old) Overrun error Overrun error (UART, SPI, I²C modes) If data is received before the previously received data in the receive data buffer has not been read, ... The overrun error flag is reset to 0 by writing 1. (New) Overrun error (UART, SPI, I²C master/slave modes) UART mode An overrun error occurs if the next reception is completed when URDIF is 1 and the receive data ... I²C master/slave mode ... To reset an overrun error, write 1 to IMEIF/ISEIF and then read the receive data buffer (USIL_RD register) twice.
	21-29	USIL: Interrupts in UART mode - Receive error interrupt (Old) ... If any of the error flags has the value 1, ... proceed with error recovery. (New) ... If any of the error flags has the value 1, ... proceed with error recovery. To reset an overrun error, ... USILMOD[2:0]/USIL_GCFG register) to initialize USIL.
	21-30	USIL: Interrupts in SPI mode - Receive error interrupt (Old) ... If SEIF is 1, the interrupt handler routine will proceed with error recovery. (New) ... If SEIF is 1, the interrupt handler routine will proceed with error recovery. To reset an overrun error, clear SEIF ... then read the receive data buffer (USIL_RD register) twice.
	21-30, 21-31	USIL: Interrupts in I²C master/slave mode - Receive error interrupt (Old) To use this interrupt, ... interrupt requests for this cause will not be sent to the ITC. ... ... If IMEIF is 1, the interrupt handler routine will proceed with error recovery. (New) To use this interrupt, ... interrupt requests for this cause will not be sent to the ITC. An overrun error occurs ... two-byte data has been received without reading the receive data buffer. The USIL module sets ... the interrupt handler routine will proceed with error recovery. To reset an overrun error, ... read the receive data buffer (USIL_RD register) twice.
	21-32, 21-40, AP-A-4, AP-A-34	USIL: 0x30065f USIL SPI Master Mode Receive Data Mask Register (USIL_SMSK) (New) Deleted
	21-34	USIL: USIL Receive Data Buffer Register (USIL_RD) - (D[7:0]) RD[7:0]: USIL Receive Data Buffer Bits (Old) If receiving the subsequent data is completed ... the new received data overwrites the contents. (New) Deleted
	21-37	USIL: USIL UART Mode Interrupt Flag Register (USIL_UIF) - (D2) UOEIF: Overrun Error Flag Bit (Old) An overrun error occurs when the previous received data ... UOEIF is reset by writing 1. (New) An overrun error occurs ... (write 0x0 to USILMOD[2:0]/USIL_GCFG register) to initialize USIL.

Code No.	Page	Contents
411911901	21-37, 21-38, AP-A-34	USIL: USIL SPI Master/Slave Mode Configuration Register (USIL_SCFG) (Old) D1 SMSKEN: Receive Data Mask Enable Bit (New) D1 Reserved (Do not set to 1.)
	21-39, 21-40	USIL: USIL SPI Master/Slave Mode Interrupt Flag Register (USIL_SIF) - (D2) SEIF: Overrun Error Flag Bit (Old) An overrun error occurs when the previous received data ... SEIF is reset by writing 1. (New) An overrun error occurs if data are received successively when SRDIF is 1. While SRDIF is set ... The procedure that writes 1 to SEIF and reads USIL_RD register twice can be reversed.
	21-42	USIL: USIL I²C Master Mode Interrupt Flag Register (USIL_IMIF) - (D[4:2]) IMSTA[2:0]: I²C Master Status Bits (Old) When an operation completion interrupt occurs, ... the operation that has been finished. (New)... the operation that has been finished. IMSTA[2:0] is automatically reset to 0x0 by writing 1 to IMIF.
		USIL: USIL I²C Master Mode Interrupt Flag Register (USIL_IMIF) - (D1) IMEIF: Overrun Error Flag Bit (Old) An overrun error occurs when the previous received data ... IMEIF is reset by writing 1. (New) An overrun error occurs ... and then read the receive data buffer (USIL_RD register) twice.
	21-44	USIL: USIL I²C Slave Mode Interrupt Flag Register (USIL_ISIF) - (D[4:2]) ISSTA[2:0]: I²C Slave Status Bits (Old) When an operation completion interrupt occurs, ... the operation that has been finished. (New)... the operation that has been finished. ISSTA[2:0] is automatically reset to 0x0 by writing 1 to ISIF.
		USIL: USIL I²C Slave Mode Interrupt Flag Register (USIL_ISIF) - (D1) ISEIF: Overrun Error Flag Bit (Old) An overrun errors occurs when the previous received data ... ISEIF is reset by writing 1. (New) An overrun error occurs ... and then read the receive data buffer (USIL_RD register) twice.
	21-46	USIL: USIL LCD SPI Mode Interrupt Flag Register (USIL_LSIF) - (D1) LSBSY: Transfer Busy Flag Bit (Old) It is cleared to 0 once the transfer is completed. (New) It is cleared to 0 after data transfer ... LSDMOD[1:0]/USIL_LSDCFG register has completed.
	22-3	I²S: Pin mode Modified Figure 22.4.1
		I²S: I²S interface clocks Modified Figure 22.4.2
	22-5, 22-28	I²S: Sampling clock cycle Modified Figures 22.4.3 and 22.8.13
	22-13	I²S: Data output timing chart Modified Figure 22.5.2
	22-14	I²S: Receive FIFO data and interrupts Modified Figure 22.6.1
	22-25	I²S: Data input timing Modified Figures 22.8.10 to 22.8.12
	24-1	CARD: CARD module overview (Old) • The data and address signals of the device can be connected directly to the external bus of the SRMAC. ... - Supports accelerated error correction. (New) • The data and address signals of the device can be connected to the external bus. ... - Supports error correction. (Error Correction Code)
	25-1	SD_MMC: SD_MMC module overview (Old) In addition to the card (NAND Flash) interface, the S1C33L27 is equipped with an SD/MMC ... (New) The S1C33L27 is equipped with an SD/MMC interface (SD_MMC) module ...
		SD_MMC: SD_MMC module overview (Old) • Variable clock rate: 0 to 30 MHz • Supports 4-bit interface (wide bus) and 1-bit (standard bus) interface. (New) • Variable clock rate: Max. 30 MHz • Supports 4-bit (wide bus) and 1-bit (standard bus) SD bus interface.
		SD_MMC: SD_MMC module overview (Old) • Supports SD/MMC mode. (New) Deleted
	26-7, 26-18	GPIO: Interrupt mode and polarity selection (port input interrupt) (Old) In SLEEP mode, the CMU senses the port interrupt ... if edge trigger mode is selected. (New) SLEEP mode can be canceled by causing a port input interrupt regardless of how the GPIO interrupt mode (edge trigger/level trigger) is set.
	26-10, 26-26	GPIO: Interrupt mode and polarity selection (key input interrupt) (Old) In SLEEP mode, the CMU senses the port interrupt ... if edge trigger mode is selected. (New) SLEEP mode can be canceled by causing a key input interrupt regardless of how the GPIO interrupt mode (edge trigger/level trigger) is set.
	27-1	ADC: ADC10 module overview (Old) • Input channels: Max. 8 channels (180-pin package) Max. 4 channels (128- or 144-pin package) (New) • Input channels: Max. 8 channels (Chip and PFBGA12U-180 package) Max. 4 channels (TQFP24/QFP20-144pin and TQFP15-128pin package)
	27-2	ADC: List of A/D converter input pins Modified Table 27.2.1

REVISION HISTORY

Code No.	Page	Contents
411911901	27-3	ADC: Selecting A/D conversion start and end channels (Old) Examples: • One A/D conversion sequence in the 180-pin package model ... • One A/D conversion sequence in the 128-pin or 144 pin package model ... Note: ... does not exist in the 128-pin or 144-pin package is specified. (New) Examples: • One A/D conversion sequence in the chip and PFBGA12U-180 package models ... • One A/D ... in the TQFP24/QFP20-144pin and TQFP15-128pin package models ... Note: ... does not exist in the TQFP24/QFP20-144pin and TQFP15-128pin packages is specified.
	28-1 to 28-45, AP-A-65	LCDC (Old) Generic HR-TFT, HR-TFT (New) TFT
	28-1	LCDC: LCDC module overview (Internal bus interface and VRAM) (Old) • The UMA (Unified Memory Access) method ... Bus Arbiter and IVRAM interface is implemented. (New) • The UMA (Unified Memory Architecture) method ... Bus Arbiter and IVRAM interface is implemented.
		LCDC: LCDC module overview (Internal bus interface and VRAM) (Old) • The LCDC registers are mapped into area 6 and 32-bit accesses are possible. (New) Deleted
	28-2	LCDC: Block diagram Modified Figure 28.2.1
		LCDC: Block diagram (Old) Look-up table In monochrome mode, ... is used as the look-up table to set up gray scale data to be displayed. (New) In monochrome mode, the look-up table included in the LCDC is used.
	28-11	LCDC: TFT panel timing parameters Modified Figure 28.5.3.1
	28-13	LCDC: TFT Panel Timing Parameters (Old) LCFC_FPFR register (New) LCDC_FPFR register
	28-20	LCDC: LUT bypass mode for TFT panel (Old) When a TFT panel is used, LUT bypass mode is effective if the conditions shown below are all met. ... • Color mode is selected (COLOR/LCDC_DISPMOD register = 1). (New) When a TFT panel is used, make the following settings: ... • Select color mode (COLOR/LCDC_DISPMOD register = 1).
	28-30	LCDC: Inverting and blanking the display (Old) This is accomplished by inverting the display data output from the look-up tables, rather than by inverting the pixel data in the display memory. (New) This is accomplished by inverting the display data output from the LCDC.
	28-32	LCDC: Sub-window configuration example (Old) If LCDC_MAINADR = 0x1003a97c, ... as the virtual main screen VRAM ends at that word. (New) Deleted
		LCDC: Sub-window configuration example (Old) If LCDC_SUBADR =0x101257fc or 0x1015fffc, ... as the virtual sub-screen VRAM ends at that word. (New) Deleted
	28-45	LCDC: LCDC Display Mode Register (LCDC_DISPMOD) - (D25) SWINV: Software Video Invert Bit (Old) Inverse operation is applied to output of the look-up tables, and does not affect the display memory. (New) Inverse operation is applied to the display data output from the LCDC, and does not affect the display memory.
	29-2	USB: USB operating clocks (Old) This clock can be stopped after USB control register settings have been finished. (New) Deleted
	29-15	USB: Device attach timing values Modified Table 29.5.1.5
	29-27	USB: SIE_IntStat (SIE Interrupt Status) - (D6) NonJ (Old) This bit is valid when the InSUSPEND bit of the USB_Control register is set to 1. (New) This bit is valid when the InSUSPEND bit of the USB_Control register is set to 1. This bit is valid during snooze as well.
	31-1, 31-2, 31-6, 31-13 to 31-16	CALC (Old) ... operation (Type 2), ... operation (Type 3) (New) Deleted (Type2) and (Type 3)
	31-1	CALC: Reserved resources (Old) • Stack 4 bytes/r4 r12 (New) Stack 4 bytes / General-purpose registers (r4–r12)
	32-1	Electrical characteristics: Absolute maximum rating (Old) No description (New) ∗2) The maximum input voltage range of the #STBY pin is V_{SS} - 0.3 V to 4.0 V.
		Electrical characteristics: Recommended operating conditions (Old) No description (New) ∗1) $HV_{DD}/AV_{DD} \geq LV_{DD}/RTCV_{DD}/PLLV_{DD}$ $LV_{DD} = RTCV_{DD} = PLLV_{DD}$ ∗2) The recommended input voltage range of the #STBY pin is V_{SS} - 0.3 V to 3.6 V.

Code No.	Page	Contents
411911901	32-2	Electrical characteristics: DC characteristics - Low level input voltage (LVCMOS) (Old) Min. – (New) Min. -0.3 V
	32-3	Electrical characteristics: DC characteristics (Old) Pull-up resistor (TYPE1) Min. 24 kΩ Pull-down resistor (TYPE1) Min. 24 kΩ Pull-up resistor (TYPE2) Min. 48 kΩ Pull-down resistor (TYPE2) Min. 48 kΩ (New) Pull-up resistor (TYPE1) Min. 25 kΩ Pull-down resistor (TYPE1) Min. 25 kΩ Pull-up resistor (TYPE2) Min. 50 kΩ Pull-down resistor (TYPE2) Min. 50 kΩ
	32-24, 32-25	Electrical characteristics: TFT panel timing (Old) Generic HR-TFT (New) TFT
	32-25	Electrical characteristics: #STBY AC characteristics Modified the characteristic table
	33-1	Basic external connection diagram: #STBY Modified the figure
	AP-A-74, AP-A-76	I/O register: SD_MMC Clock Control Register (MMC_CLKCTL) SD_MMC Interrupt Control Register (MMC_INT) (Old) No description (New) ∗ Not initialized by SYNRST/MMC_FUNCTL register.
	AP-B-2	Power Saving: List of clock control conditions Modified Table B.2
	AP-C-2	Mounting precautions: Prohibited pattern Modified the figure
	AP-C-3	Mounting precautions: Precautions on VBUS Modified the figure

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.
20/F, Harbour Centre, 25 Harbour Road,
Wanchai, Hong Kong
Phone: +852-2585-4600 Fax: +852-2827-4346
Telex: 65542 EPSCO HX

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: 411911901
First Issue April 2010
Revised December 2010 in JAPAN Ⓛ