TOSHIBA 32-bit RISC Microprocessor

TX03 Series

TMPM320C1DFG

Under development/Tentative

Rev1.05

This data sheet is an interim version that includes contents under editing.

TOSHIBA CORPORATION

Semiconductor Company

Contents

RESTRICTIONS ON PRODUCT USE

- The information contained herein is subject to change without notice.
- TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
  In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the “Handling Guide for Semiconductor Devices,” or “TOSHIBA Semiconductor Reliability Handbook” etc.
- The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury (“Unintended Usage”). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc.. Unintended Usage of TOSHIBA products listed in his document shall be made at the customer’s own risk.
- The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations.
- The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patents or other rights of TOSHIBA or the third parties.
- Please contact your sales representative for product-by-product details in this document regarding RoHS compatibility. Please use these products in this document in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances. Toshiba assumes no liability for damage or losses occurring as a result of noncompliance with applicable laws and regulations.

****

ARM, ARM Powered, ARM7TDMI, ARM9TDMI, ADK, ARMulator, Embedded-ICE, Jazelle, MicroPack, ModelGen, MOVE, Multi-ICE, PrimeCell, StrongARM, The Architecture for the Digital World, and Thumb are registered trademarks of ARM Limited in the EU and other countries.

ACT, AMBA, Integrator, MultiTrace, PrimXsys, RealView, and SecurCore are trademarks of ARM Limited in the EU and other countries.

****

- Introduction - Notes on the registers -

This device has SFR (Special Function Register) each IP (Peripheral circuits). SFR is shown as following in this data book.

a) IP lists

- IP lists show the register name, address and easy descriptions.
- 32bit address is assigned to all registers. It shows as **[base address + (specific) address].**

base address = 0x0000_0000

Register Name	Address (base+)	Description
SAMPLE	0x0001	Sample register
...	...	...

Note1: Case of this register (SAMPLE): 00000001 address because 00000000 address (hex)+0001 address (hex)

Note2: This register is sample register. There is not this data book.

b) SFR (register) description

- Basically, each register is structured 32 bit register. (There is a part of exception.)
- Each description shows Bit, Bit Symbol, Type, Reset value and Description.

Address = (0x0000_0000) + (0x0001)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:6]	SAMPLE76	R/W	0y00	Sample setting 0y00: Set to Sample mode 0 0y01: Set to Sample mode 1 0y10: Set to Sample mode 2 0y11: Set to Sample mode 3
...	...	...	...	...

Note1: Basically 3types.

R/W(READ/WRITE) :	Enable Read/Write
RO(READ ONLY) :	Enable Read only
WO(WRITE ONLY) :	Enable Write only

There are exception types (USB device controller and SD host controller).
Please refer to those sections.

Note2: Bit state description:

Hexadecimal:	0x00FF = 255 (Decimal)
Binary:	0y0101 = 5 (Decimal)

Note3: 1 Word = 32 bit.

32-bit RISC Microprocessor

TMPM320C1DFG

1. Overview and Features

The TMPM320C1D is a 32-bit RISC microprocessor with a built-in ARM Cortex-M3 core, which is suitable for microcontrollers and built-in processors for applications.

The function overview and features are as follows:

(1) Cortex-M3 manufactured by ARM is used.

- ARMv7-M architecture
- Thumb-2 command set
- 3-stage pipeline

(2) Maximum operating frequency: 144 MHz (12 MHz × 12 multiplied)

(3) Built-in program memory: Data memory

- Built-in RAM
  RAM: 320 Kbytes (can be used as programs or data)
  eDRAM: 1 Mbyte (can be used as programs or data)

(4) Memory controller

- Chip-select output: 2 channels
- Supports non-synchronous memories (such as SRAM and NOR Flash memory).
- Separate bus system:

External address	23 bits: A0-A22 (Access space of minimum 1 MB to maximum 16 MB × 2ch)
External data bus	16 bits: D0-D15

(5) 16-bit timer: 8 channels

- Supports 4-system PWM output by using timers of 4 out of 8 channels.

(6) Synchronous serial bus interface (SSP): 4 channels

- Supports the formats SPI, SSI, and MicroWire.

(7) I^2C bus interface: 2 channels

(8) UART: 4 channels

- Channel0/Channel1: Supports Full UART.
- Channel2/Channel3: Supports only 2 pins: TXD and RXD.

(9) I^2S (Inter-IC Sound) bus interface: 2 channels (Input: 1 channel, output: 1 channel)

- Can control input and output independently.
- Supports the formats of each front-aligned, back-aligned, and I2S.
- Supports 16, 18, 20, and 24 bit data length.
- Supports 32, 48, and 64 slots.

Note: For more information on audio standard output frequency errors, refer to Section 3.20.

(10) USB host controller: 1 channel

- Compliant with the USB2.0 Specification revision 2.0
- Enhanced Host Controller Interface (EHCI) Specification revision 1.0
- Complies with Open Host Controller Interface (OHCI) for USB release 1.0a.
- Supports High Speed (480 Mbps) and Full Speed (12 Mbps).

(11) 10-bit AD converter (with a built-in sample-and-hold circuit): 4 channels

(12) Watchdog timer

(13) Interrupt function: 28 types

- External 4 types INT0 to INT3 (edge: rise and fall, level: High and Low)
- Internal 24 types 16-bit timer × 8, A/D converter × 1, UART × 4, I2C × 2, SSP × 4, USB × 1, SD host controller × 1, DMAC × 2, WDT × 1

(14) Input/output port: 55 terminals

(15) DMA controller: 8 channels

(16) SD host controller: 1 channel

- Supports SD association(SDA) specifications version 2.00.
- Supports SD card I/F mode (4-bit parallel).
- Built-in 512-byte FIFO buffer

(17) Clock control function

- Two blocks of built-in clock multiple circuit (PLL) enable an external 12 MHz oscillator to supply USB clock frequency of 480 MHz and clock frequency of 144 MHz to the CPU.

(18) Debugging function

- Supports on-chip debugging.
- Supports SWD (Serial wire debug) as the debugging function.
- Supports ETM (Embedded trace macrocell) as the tracing function.

(19) Operating voltage

- Power supply for built-in logic DVCC12 = 1.2V ± 0.1V
- Power supply for PLL DVCC12PLL = 1.2V ± 0.1V
- Power supply for eDRAM DVCC33DRM = 3.3V ± 0.3V, DVCC12DRM =1.2V ± 0.1V
- External I/O DVCC33IO = 3.3V ± 0.3V
- Power supply for AD converter AVDD33ADC = 3.3V ± 0.3V
- Power supply for USB AVDD33USB = 3.3V ± 0.3V
  DVDD12USB = 1.2 ± 0.1V

(20) Temperature range

- -40°C to 85°C

(21) Package

- 144-pin QFP: LQFP144-P-2020-0.50E

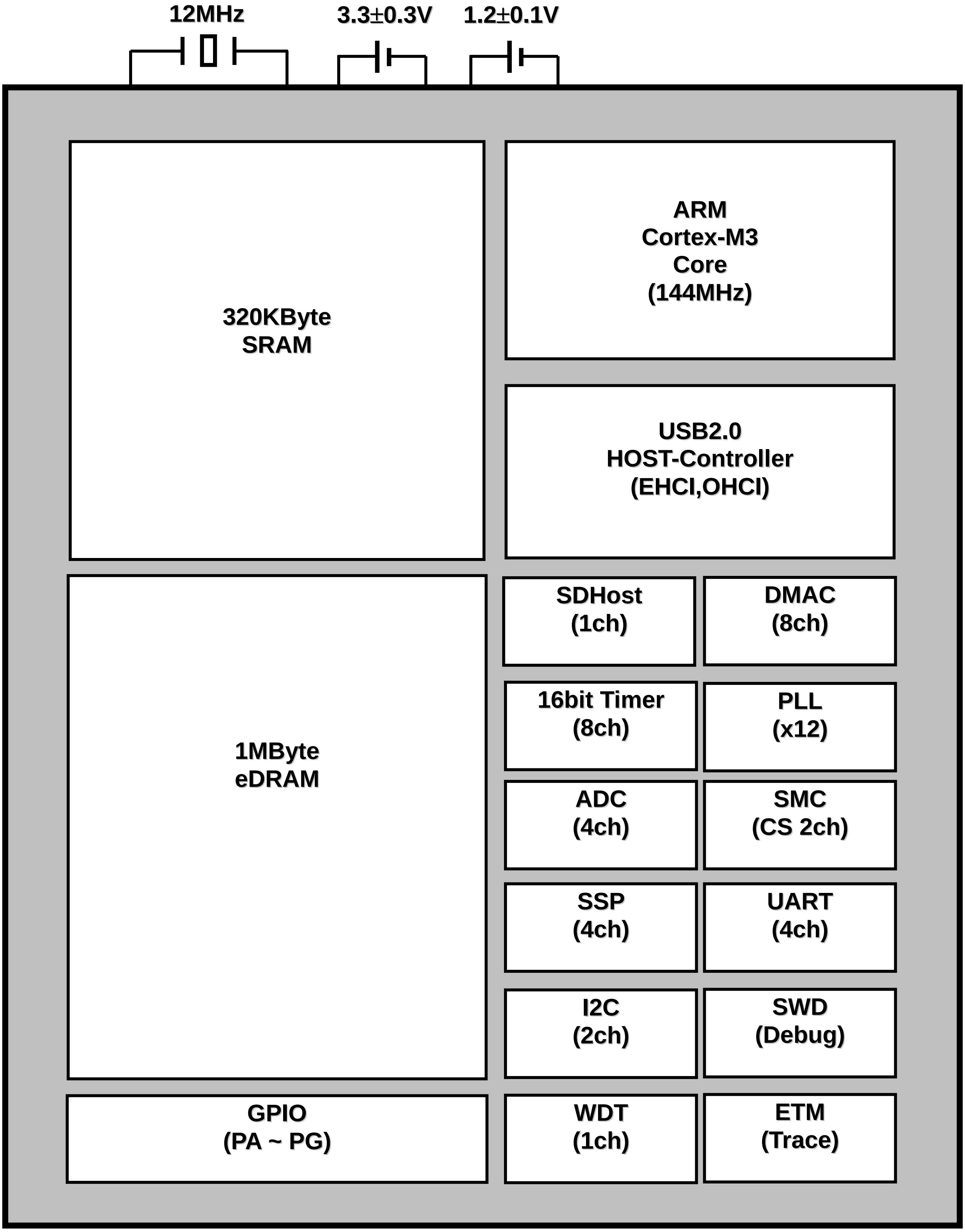


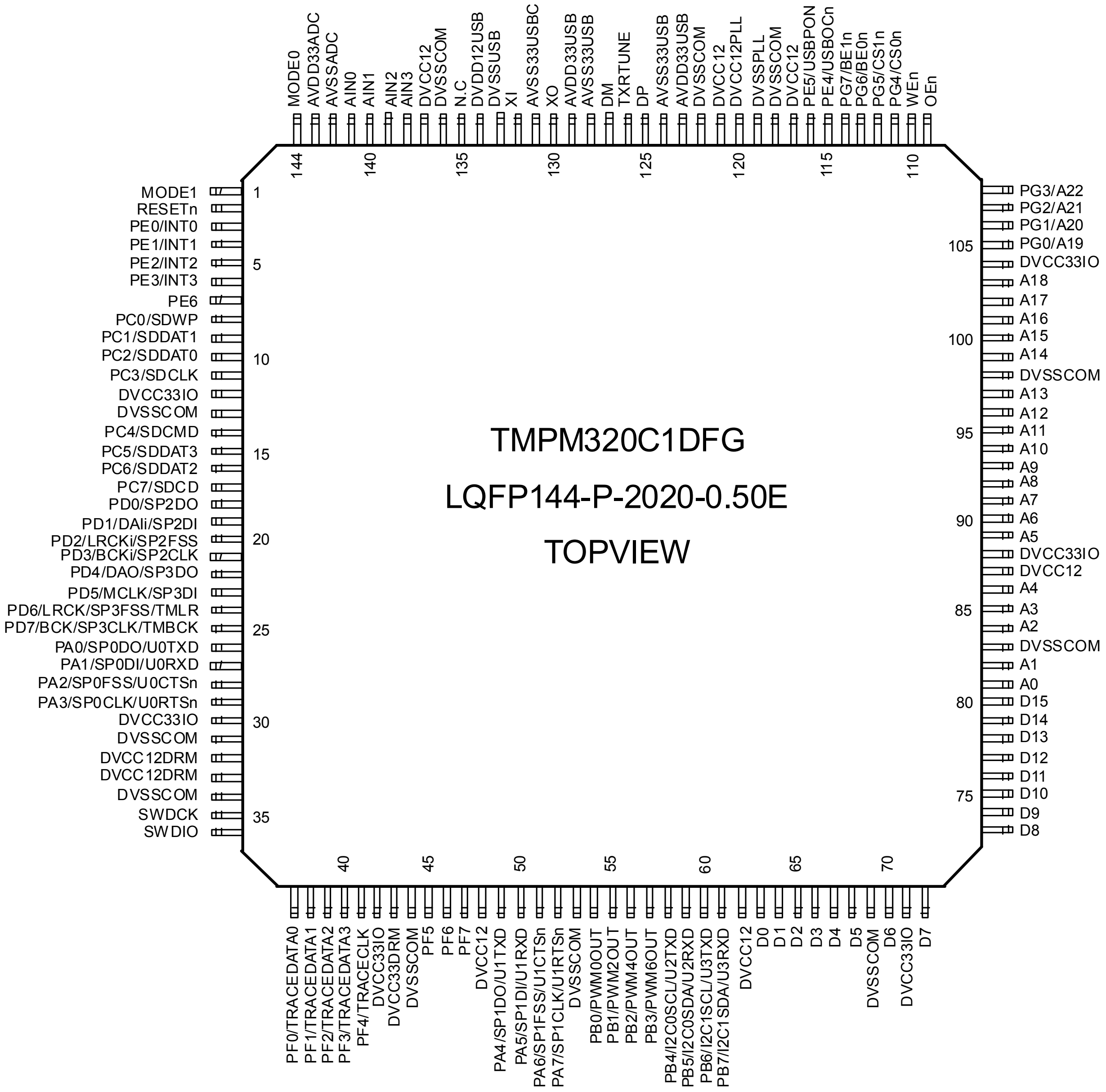
Figure 1.1 TMPM320C1D Block Diagram

2. Pin Configuration and Function

This section provides a pin configuration diagram of the TMPM320C1D, names of I/O pins, and brief description of their functions.

2.1 Pin Configuration Diagram (Top View)

Figure 2.1.1 shows the TMPM320C1DFG pin configuration.



Note) Open N.C pins are acceptable.

Figure 2.1.2 Pin configuration diagram

2.2 Pin Name and Function

Names and functions of I/O pins are shown below.

Table 2.2.1 Pin names and functions (1/4)

Pin name	Number of pins	Input/Output	Function	Remarks
D0 ~ D7	8	Input/Output	Data: Data bus D0 to D7	
D8 ~ D15	8	Input/Output	Data: Data bus D8 to D15	
A0 ~ A7	8	Output	Address: Address bus A0 to A7	
A8 ~ A15	8	Output	Address: Address bus A8 to A15	
A16 (A18	3	Output	Address: Address bus A16 to A18	
OEn	1	Output	Out-enable signal for NOR FLASH/SRAM	
WEn	1	Output	Write-enable signal for NOR FLASH/SRAM	
PG0 ~ PG3 A19 ~ A22	4	Input/Output Output	Port G0 to G3: I/O port Address: Address bus A19 to A22	
PG4 CS0n	1	Input/Output Output	Port G4: I/O port Chip select 0 for NOR-FLASH/SRAM	
PG5 CS1n	1	Input/Output Output	Port G5: I/O port Chip select 1 for NOR-FLASH/SRAM	
PG6 BE0n	1	Input/Output Output	Port PG6: I/O port Byte enable signal (D0 to D7) for SRAM	
PG7 BE1n	1	Input/Output Output	Port PG7: I/O port Byte enable signal (D8 to D15) for SRAM	
PA0 SP0DO U0TXD	1	Input/Output Output Output	Port A0: I/O port Data output pin for SSP0 UART function 0 transmission data	
PA1 SP0DI U0RXD	1	Input/Output Input Input	Port A1: I/O port Data input pin for SSP0 UART function 0 transmission data	
PA2 SP0FSS U0CTSn	1	Input/Output Input/Output Input	Port A2: I/O port FSS pin for SSP0 UART function 0 CTS data input (Clear to send)	
PA3 SP0CLK U0RTSn	1	Input/Output Input/Output Output	Port A3: I/O port Clock pin for SSP0 UART function 0 output modem control line RTS (Request To Send)	
PA4 SP1DO U1TXD	1	Input/Output Output Output	Port A4: I/O port Data output pin for SSP1 UART function 1 transmission data	
PA5 SP1DI U1RXD	1	Input/Output Input Input	Port A5: I/O port Data input pin for SSP1 UART function 1 transmission data	
PA6 SP1FSS U1CTSn	1	Input/Output Input/Output Output	Port A6: I/O port FSS pin for SSP1 UART function 1 receive data	
PA7 SP1CLK U1RTSn	1	Input/Output Input/Output Output	Port A7: I/O port Clock pin for SSP1 UART function 1 output modem control line RTS (Request To Send)	

Table 2.2.2 Pin names and functions (2/4)

Pin name	Number of pins	Input/Output	Function	Remarks
PB0 PWM0OUT	1	Input/Output Output	Port B0: I/O port Timer PWM0 output pin	
PB1 PWM2OUT	1	Input/Output Output	Port B1: I/O port Timer PWM1 output pin	
PB2 PWM4OUT	1	Input/Output Output	Port B2: I/O port Timer PWM4 output pin	
PB3 PWM6OUT	1	Input/Output Output	Port B3: I/O port Timer PWM6 output pin	
PB4 I2C0SCL U2TXD	1	Input/Output Input/Output Output	Port B4: I/O port Clock input/output pin for I2C0 UART function 2 transmission data	
PB5 I2C0SDA U2RXD	1	Input/Output Input/Output Input	Port B5: I/O port Data input/output pin for I2C0 UART function 2 receive data	
PB6 I2C1SCL U3TXD	1	Input/Output Input/Output Output	Port B6: I/O port Clock input/output pin for I2C1 UART function 3 transmission data	
PB7 I2C1SDA U3RXD	1	Input/Output Input/Output Input	Port B7: I/O port Data input/output pin for I2C1 UART function 3 receive data	
PC0 SDWP	1	Input/Output Input	Port C0: I/O port SD: Write-protect signal	
PC1 SDDAT1	1	Input/Output Input/Output	Port C1: I/O port SD: Data 1 input/output pin	
PC2 SDDAT0	1	Input/Output Input/Output	Port C2: I/O port SD: Data 0 input/output pin	
PC3 SDCLK	1	Input/Output Output	Port C3: I/O port SD: Clock output pin	
PC4 SDCMD	1	Input/Output Output	Port C4: I/O port SD: Command output pin	
PC5 SDDAT3	1	Input/Output Input/Output	Port C5: I/O port SD: Data 3 input/output pin	
PC6 SDDAT2	1	Input/Output Input/Output	Port C6: I/O port SD: Data 2 input/output pin	
PC7 SDCD	1	Input/Output Input	Port C7: I/O port SD: Card detection pin	

Table 2.2.3 Pin names and functions (3/4)

Pin name	Number of pins	Input/Output	Function	Remarks
PD0 SP2DO	1	Input/Output Output	Port D0: I/O port Data output pin for SSP2	
PD1 SP2DI DAIi	1	Input/Output Input Input	Port D1: I/O port Data input pin for SSP2 DAIi data input pin for I2S	
PD2 SP2FSS LRCKi	1	Input/Output Input/Output Input	Port D2: I/O port FSS pin for SSP2 LRCKi clock input pin for I2S	
PD3 SP2CLK BCKi	1	Input/Output Input/Output Input	Port D3: I/O port Clock pin for SSP2 BCKi clock input pin for I2S	
PD4 SP3DO DAO	1	Input/Output Output Output	Port D4: I/O port Data output pin for SSP2 DAO data output pin for I2S	
PD5 SP3DI MCLK	1	Input/Output Input Output	Port D5: I/O port Data input pin for SSP3 MCLK clock output pin for I2S	
PD6 SP3FSS TMLRCK LRCK	1	Input/Output Input/Output Output Output	Port D6: I/O port FSS pin for SSP3 SSP_LRCK output LRCK clock output pin for I2S	
PD7 SP3CLK TMBCK BCK	1	Input/Output Input/Output Output Output	Port D7: I/O port Clock pin for SSP3 SSP_BCK output BCK clock output pin for I2S	
PE0 INT0	1	Input/Output Input	Port PE0: I/O port External interrupt request pin 0	
PE1 INT1	1	Input/Output Input	Port PE1: I/O port External interrupt request pin 1	
PE2 INT2	1	Input/Output Input	Port PE2: I/O port External interrupt request pin 2	
PE3 INT3	1	Input/Output Input	Port PE3: I/O port External interrupt request pin 3	
PE4 USBOCn	1	Input/Output Input	Port PE4: I/O port USB OC (over-current) input pin	
PE5 USBPON	1	Input/Output Output	Port PE5: I/O port USB PON (power-on) output pin	
PE6	1	Input/Output	Port PE6: I/O port	
PF0 TRACEDATA0	1	Input/Output Output	Port PF0: I/O port ETM: Trace data output pin 0	
PF1 TRACEDATA1	1	Input/Output Output	Port PF1: I/O port 1 ETM: Trace data output pin 1	
PF2 TRACEDATA2	1	Input/Output Output	Port PF2: I/O port ETM: Trace data output pin 2	
PF3 TRACEDATA3	1	Input/Output Output	Port PF3: I/O port ETM: Trace data output pin 3	
PF4 TRACECLK	1	Input/Output Output	Port PF4: I/O port ETM: Trace clock output pin	
PF5	1	Input/Output	Port PF5: I/O port	
PF6	1	Input/Output	Port PF6: I/O port	
PF7	1	Input/Output	Port PF7: I/O port	

Table 2.2.4 Pin names and functions (4/4)

Pin name	Number of pins	Input/Output	Function	Remarks
DP	1	Input/Output	USB pin (D+)	
TXRTUNE	1	Input	Connect to the AVDD33USB at the resistance of 44.2Ω.	
DM	1	Input/Output	USB pin (D-)	
AIN0	1	Input	Analog input 0: AD converter input pin	
AIN1	1	Input	Analog input 1: AD converter input pin	
AIN2	1	Input	Analog input 2: AD converter input pin	
AIN3	1	Input	Analog input 3: AD converter input pin	
SWDCK	1	Input	SWD: Clock input pin for SWD (Serial Wire Debug)	
SWDIO	1	Input/Output	SWD: Data input/output pin for SWD (Serial Wire Debug)	
RESETn	1	Input	Reset: Initializes TMPM320C1D (with Schmitt input)	
MODE0	1	Input	Startup mode input pin 0	
MODE1	1	Input	Startup mode input pin 1	
XI	1	Input	High-frequency oscillator connection input pin	
XO	1	Output	High-frequency oscillator connection output pin	
DVCC12	6	Power supply	VCC power supply for internal logic	
DVCC33IO	6	Power supply	VCC power supply for external I/O	
DVSSCOM	11	Power supply	Common VSS power supply (GND)	
DVCC33DRM	1	Power supply	VCC power supply for the internal eDRAM	
DVCC12DRM	2	Power supply	VCC power supply for the internal eDRAM	
DVCC12PLL	1	Power supply	VCC power supply for the internal PLL	
DVSSPLL	1	Power supply	VSS power supply for the internal PLL (GND)	
AVDD33USB	2	Power supply	VCC power supply for USB	
AVSS33USB	2	Power supply	VSS power supply for USB (GND)	
AVSS33USBC	1	Power supply	VSS power supply for high-frequency oscillator (GND)	
DVSSUSB	1	Power supply	VSS power supply for USB	
DVDD12USB	1	Power supply	VCC power supply for USB	
AVDD33ADC	1	Power supply	VCC power supply for A/D converter	
AVSSADC	1	Power supply	VSS power supply for A/D converter (GND)	

3. Operation Description

3.5 System

3.5.1 Reset operation

Before resetting TMPM320C1D, make sure that the power supply voltage is within the operating range, oscillation from the internal oscillator is stable at 20 system clock cycles (1.67 μs @ X1 = 12 MHz) at least, and the RESETn input pin is set to the "L" level.

After reset, the PLL stops and the PLL output becomes unselected (FCSEL=0).

Therefore, the system clock operates at 12 MHz (X1 = 12 MHz) after reset.

If the reset instruction is accepted, the built-in I/O, I/O ports, and other pins are initialized.

Initializing the internal I/O registers

(Refer to the chapter on ports and pins for initial values.)

Note 1) This LSI has a built-in RAM (SRAM and eDRAM), but its data may be lost as a result of reset operation. Initialize data in the built-in RAM after the reset operation.

The Cortex-M3 core interrupt uses the vector method, which sets the 32-bit data (address) stored in the exception address of 0x0000_0004 into the program counter when reset operation has just been performed.

Also, a warm-up time is required for the eDRAM, PLL and USBHC after the reset operation is cancelled. Refer to the chapter on each peripheral for details.

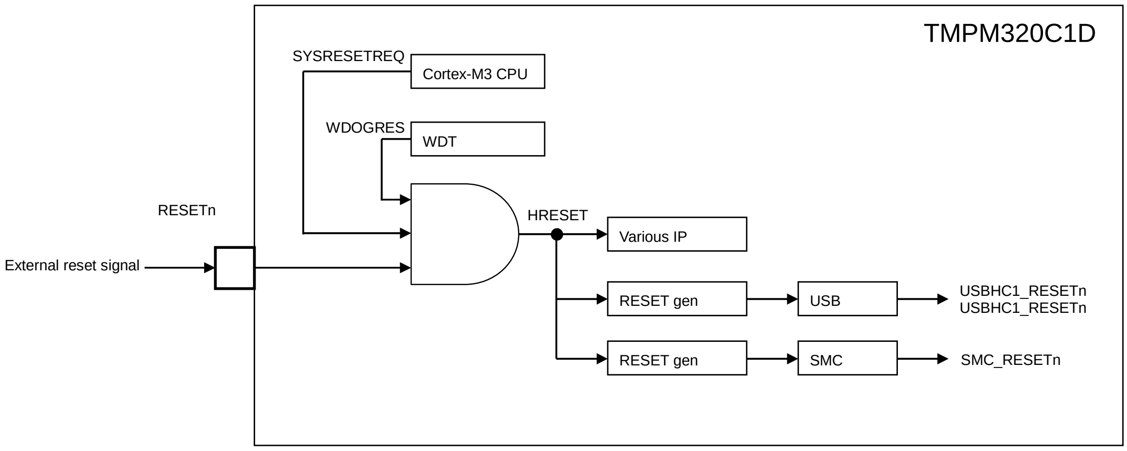
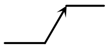


Figure 3.1.1-1 Block Diagram of Reset Input

3.5.2 Startup Mode

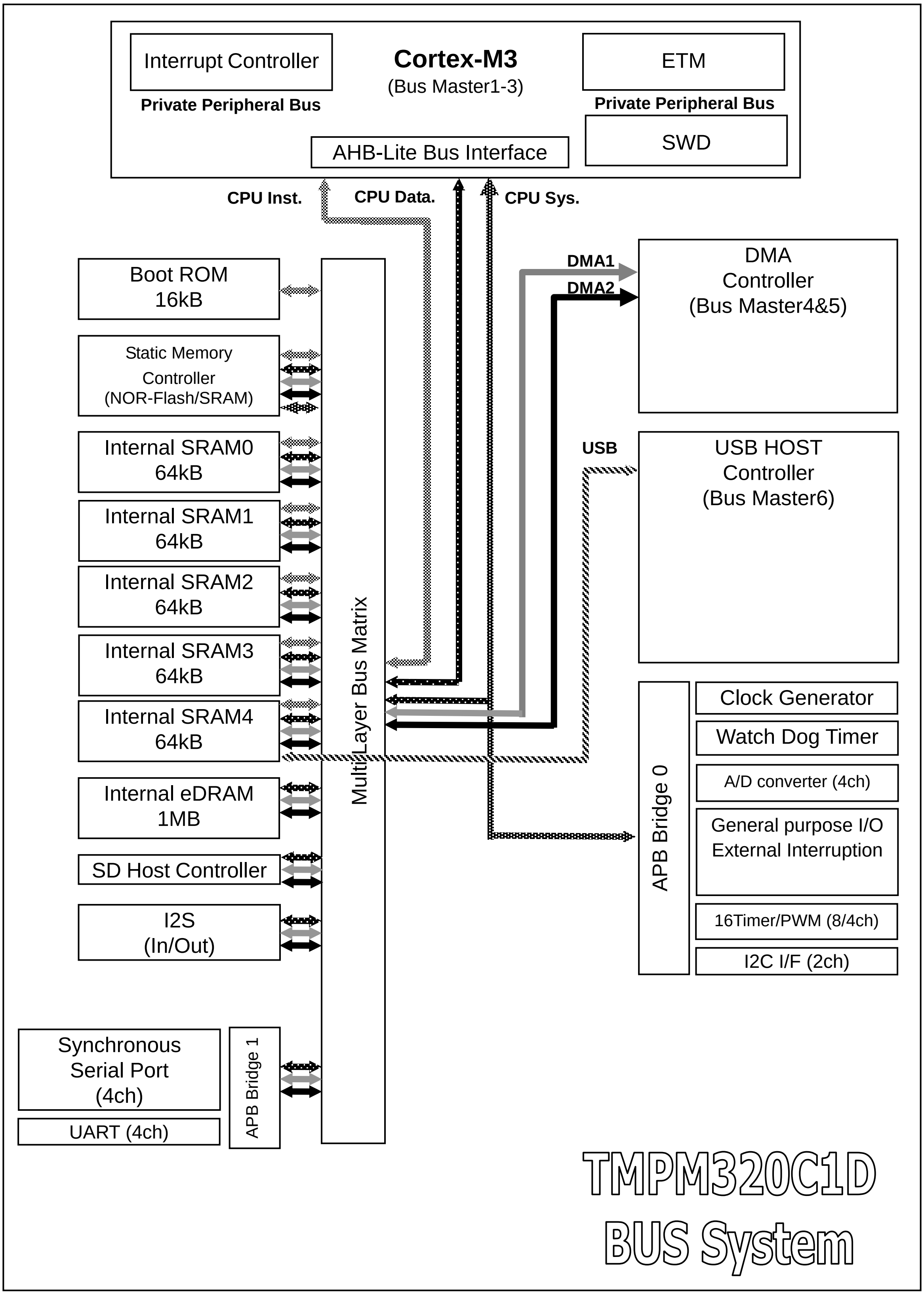
This microcomputer can use external pin settings to select a startup mode.

1. Startup Memory Setting

Mode setting pin			Operation mode
RESETn	MODE1	MODE0	
	0	0	Boot mode: Starts from the built-in boot ROM
	0	1	Normal mode: Starts from the external memory
	1	0	Setting prohibited
	1	1	Setting prohibited

3.5.3 Bus Configuration

The TMPM320C1D uses the AHB-Lite bus system in the 6-layer configuration.



3.2 Debug Interface

3.2.1 Overview of Specifications

The TMPM320C1D contains the SWD (Serial Wire Debug) unit as the debugging interface for connection to the ICE (In-Circuit Emulator), and the ETM (Embedded Trace Macrocell) unit for tracing and outputting internal programs. ETM outputs signals to dedicated pins (TRACEDATA[0] to [3]) through the TPIU (Trace Port Interface Unit) in the microprocessor.

When using the tracing function, you need to enable the corresponding bits in the port function control register GPIOFFR1 of PORTF and the TRACECLKIN output enable bit, CG_PLLCTRL6< TRACECLKINEN >, in the clock control register 6.

For more information on SWD, ETM, and TPIU, please refer to the documents published by ARM.

Note that the TMPM320C1D does not support connections based on the JTAG (Joint Test Action Group) Standards. Use SWD-compatible tools when connecting the TMPM320C1D to the ICE.

3.2.2 SWD Features

- Supports 2-pin debug interfaces (SWDCK and SWDIO)

3.2.3 ETM Features

- Supports trace output using 4 data signal pins (TRACEDATA[0] to [3]) and 1 clock signal pin (TRACECLK).

(Note 1) Disable the watchdog timer during debug operation.

(Note 2) Switch the port function settings before using TRACE.

(Note 3) For TRACECLK, you can change the output frequency using CG_PLLCTRL6<FTRACEDIV[17:16]>. For more information, refer to section 3.5, "CG_PLLCTRL6 (CG PLL Control Register 6)."

(Note 4) Note that a user program is executed for several tens to hundreds of ms (depending on the SWDCK speed) in order to establish SWD communication with the TMPM320C1D after the reset output from the ICE is cancelled.

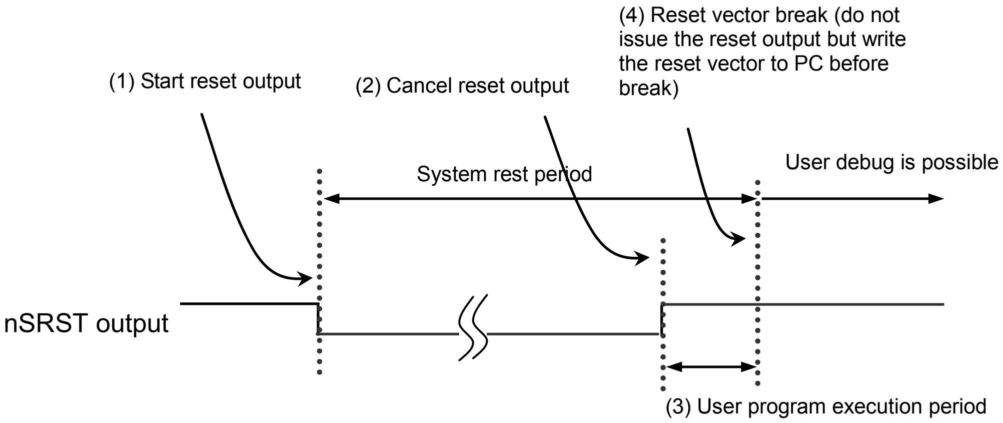

(4) Reset vector break (do not issue the reset output but write the reset vector to PC before break)
(1) Start reset output
(2) Cancel reset output
System rest period
User debug is possible
nSRST output
(3) User program execution period

Figure 3.2.3-1 SWD Communication Establishment Sequence with ICE after Reset Output

3.2.4 Connection example

An example of connections between the TMPM320C1D and the ICE (SWD connection) is shown below.

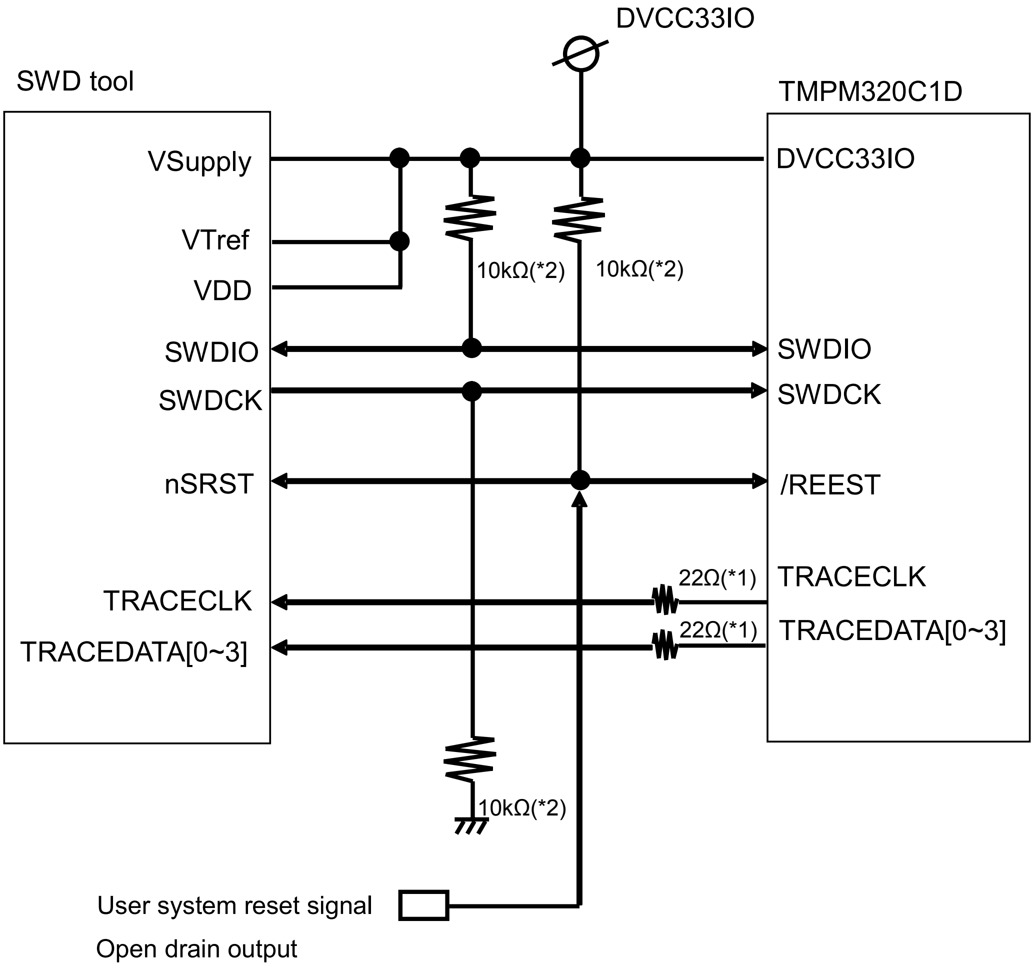


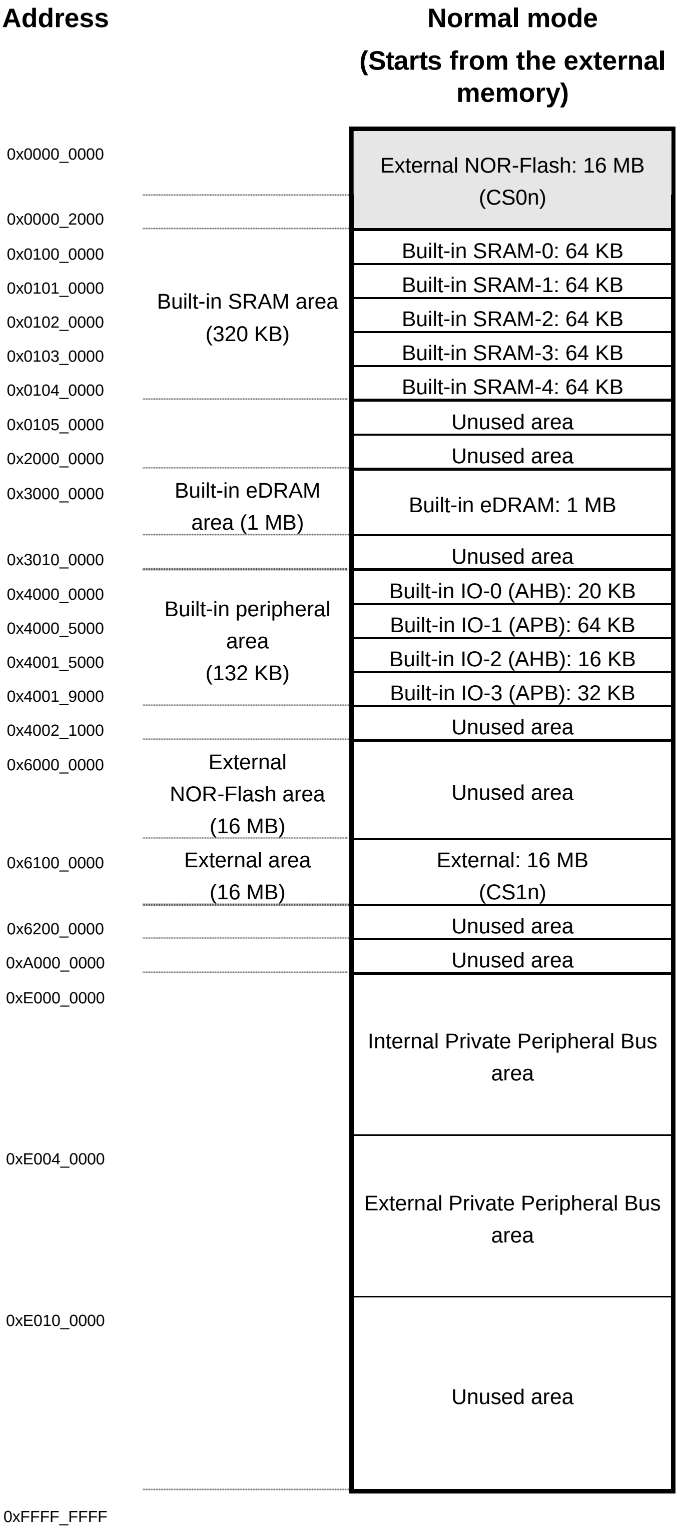
Figure 3.2.4-2 Connection example with the SWD tool

(*1) The resistance is a recommended value. Please select an appropriate resistance value according to the user's needs.

(*2) To avoid malfunction of the MPU when the ICE is not connected, pull down at the resistance of 10 kohm.

3.3 Memory Map

The memory map of the TMPM320C1D is as follows:



Note 1) Unused areas must not be accessed.

Note 2) Refer to the documentations published by ARM for details.

Figure 3.3.1 Memory map (details of startup mode, external areas, and internal areas)

The following list shows the access relationship between the bus master and slaves (IPs) in normal mode:

Connection relationship between the Bus Master and the Slave
○: Accessible, Δ: The default slave responds
-: Unavailable for access, x: Must not be accessed

Address	**Normal mode** **(Starts from the external memory)**		**CPU(I)** **M1**	**CPU(D)** **M2**	**CPU(S)** **M3**	**DMA1** **M4**	**DMA2** **M5**	**USB** **M6**
0x0000_0000 0x0000_2000	**SMCCS0n** External NOR-Flash: 16 MB		○	○	–	○	○	Δ
0x0100_0000	Built-in SRAM-0: 64 KB	Built-in SRAM area (320 KB)	○	○	–	○	○	Δ
0x0101_0000	Built-in SRAM-1: 64 KB		○	○	–	○	○	Δ
0x0102_0000	Built-in SRAM-2: 64 KB		○	○	–	○	○	Δ
0x0103_0000	Built-in SRAM-3: 64 KB		○	○	–	○	○	Δ
0x0104_0000	Built-in SRAM-4: 64 KB		○	○	–	○	○	○
0x0105_0000	Unused area		Δ	Δ	–	Δ	Δ	Δ
0x2000_0000	Unused area		–	–	Δ	Δ	Δ	Δ
0x3000_0000	Built-in eDRAM: 1 MB	Built-in eDRAM area (1 MB)	–	–	○	○	○	Δ
0x3010_0000	Unused area		–	–	Δ	Δ	Δ	Δ
0x4000_0000	Built-in IO-0 (AHB): 20 KB	Built-in I/O area (132 KB)	–	–	○	Δ	Δ	Δ
0x4000_5000	Built-in IO-1 (APB): 64 KB		–	–	○	Δ	Δ	Δ
0x4001_5000	Built-in IO-2 (AHB): 16 KB		–	–	○	Δ	Δ	Δ
0x4001_9000	Built-in IO-3 (APB): 32 KB		–	–	○	○	○	Δ
0x4002_1000	Unused area		–	–	Δ	Δ	Δ	Δ
0x6000_0000	Unused area	External NOR-Flash area (16 MB)	–	–	Δ	Δ	Δ	Δ
0x6100_0000	SMCCS1n External: 16 MB	External area (16 MB)	–	–	○	○	○	Δ
0x6200_0000	Unused area		–	–	Δ	Δ	Δ	Δ
0xA000_0000	Unused area		–	–	Δ	Δ	Δ	Δ
0xE000_0000	Internal Private Peripheral Bus area		–	–	–	Δ **(Note 1)**		
0xE004_0000	External Private Peripheral Bus area		–	–	–	Δ **(Note 2)**		
0xE010_0000	Unused area		–	–	Δ	Δ	Δ	Δ
0xFFFF_FFFF								

Note 1) Internal Private Peripheral Bus (PPB) access area

Note 2) External Private Peripheral Bus (PPB) access area

Figure 3.3.2 Memory map
(details of external areas, and connection relationship between the Bus Master and the Slave)

The following list shows the relationship between the IPs and the register access size:

Address	Bus	IP	Register Access Size
0x4000_0000	Built-in IO-0 (AHB): 20 KB	DMAC	8,16,32bit
0x4000_1000		eDRAMC	8,16,32bit
0x4000_2000		USB(EHCI)	8,16,32bit
0x4000_3000		USB(OHCI)	8,16,32bit
0x4000_4000		SMC	8,16,32bit
0x4000_5000	Built-in IO-1 (APB): 64 KB	CG&PLL	32bit
0x4000_6000		WDT	32bit
0x4000_7000		ADC	32bit
0x4000_8000		Port A	32bit
0x4000_9000		Port B	32bit
0x4000_A000		Port C	32bit
0x4000_B000		Port D	32bit
0x4000_C000		Port E	32bit
0x4000_D000		Port F	32bit
0x4000_E000		Port G	32bit
0x4000_F000		Timer01/PWM0	32bit
0x4001_0000		TImer23/PWM2	32bit
0x4001_1000		Timer45/PWM4	32bit
0x4001_2000		Timer67/PWM6	32bit
0x4001_3000		I2C0	32bit
0x4001_4000		I2C1	32bit
0x4001_5000	Built-in IO-2 (AHB): 16 KB	SDHost(DATA)	16bit
0x4001_6000		I2S	32Bit
0x4001_7000		reserved	-
0x4001_8000		SDHost(registers)	16bit
0x4001_9000	Built-in IO-3 (APB): 32 KB	UART0	32bit
0x4001_A000		UART1	32bit
0x4001_B000		UART2	32bit
0x4001_C000		UART3	32bit
0x4001_D000		SSP0	32bit
0x4001_E000		SSP1	32bit
0x4001_F000		SSP2	32bit
0x4002_0000		SSP3	32bit
0x4002_1000			

Note) Access to the SDHost Controller is fixed to 16-bit access including the registers

Figure 3.3.3 Memory map (details of peripherals)

3.4 CPU

The TMPM320C1D contains the 32-bit RISC processor core (Cortex-M3 core) made by ARM.
This section describes the overview of the Cortex-M3 and the configuration information in the TMPM320C1D. For more information on the Cortex-M3, please refer to the documents published by ARM.

http://www.arm.com/

The following shows the diagram of the Cortex-M3 core:

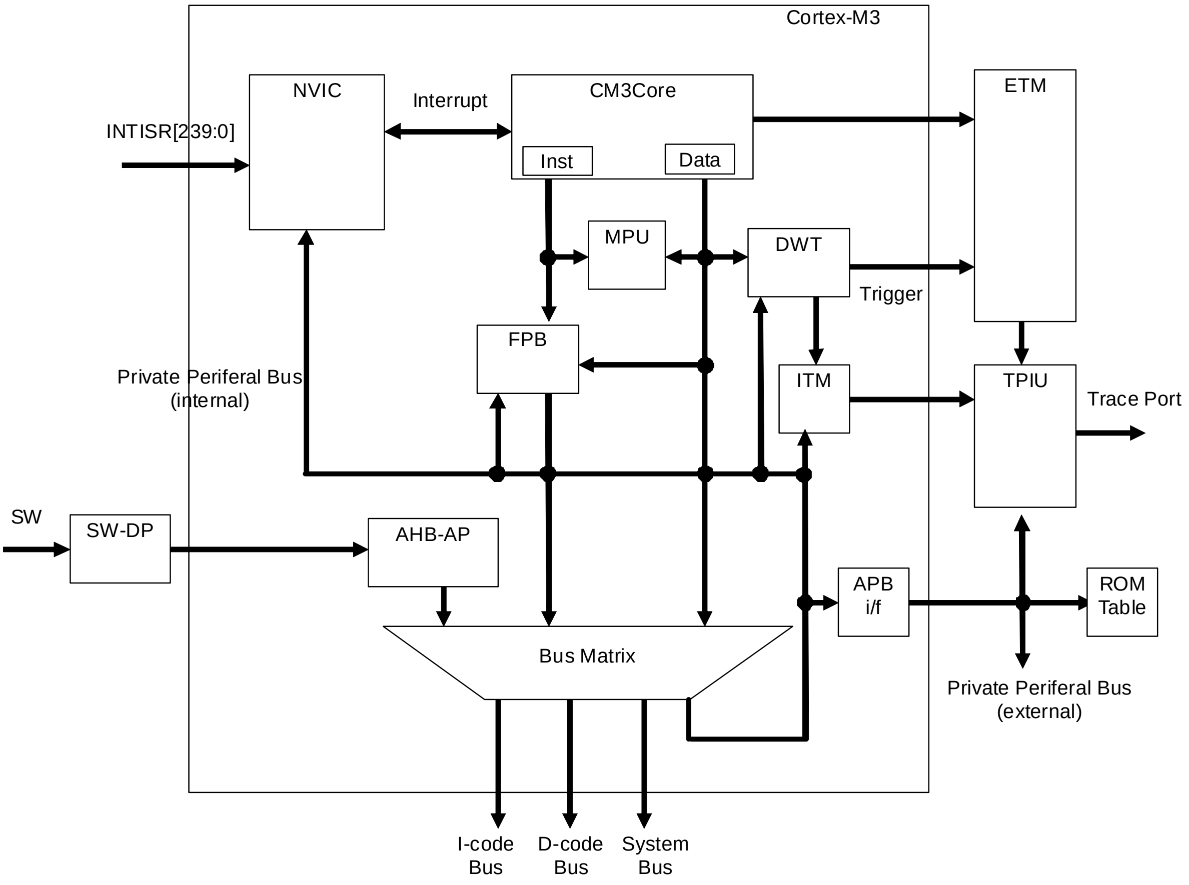


Figure 3.4.1 Schematic diagram of Cortex-M3 core

3.4.1 Core Configuration

The following shows the Cortex-M3 core configuration selected in the TMPM320C1D:

Configuration	Description
Number of interrupts	29 channels
Interrupt priority level (Note 1)	4 bits (16 levels)
MPU (Memory Protection Unit)	Installed
SW / SWJ-DP	SW-DP only (SWD connection only)
ETM (Embedded Trace Macrocell)	Installed
Endian	Little endian

(Note 1) Refer to pages 102/384, Cortex-M3 Technical Reference Manual.

3.4.2 Exceptions

The following list shows the Cortex-M3 exception types:
INTWDT and the subsequent areas are the vector areas unique to the TMPM320C1D. For more information, refer to section 3.7, "Interrupts."

Exception	Address	Remarks
Top of Stack	0x00000000	Beginning of a stack
Reset	0x00000004	Reset
reserved	0x00000008	-
Hard Fault	0x0000000C	Hard fault
MPU Fault	0x00000010	Memory management
Bus Fault	0x00000014	Bus fault
Usage Fault	0x00000018	Usage fault
reserved	0x0000001C	-
reserved	0x00000020	-
reserved	0x00000024	-
reserved	0x00000028	-
SVCall	0x0000002C	Supervisor call
Debug Monitor	0x00000030	Debug monitor
PendSV	0x00000038	Software pending request
SysTick	0x0000003C	SysTick interrupt
INTWDT :	0x00000040 :	Watchdog timer interrupt :

3.5 Clock Controller

3.5.1 Overview

This block is a circuit that controls the clock for the overall TMPM320C1D. It has the following features:

a. Writing to registers inside the clock controller is prohibited.

b. Supplies and stops a clock on an IP basis.

c. Selects and sets a SYSTICK_CLK clock (HCLK/2, HCLK/4).

d. Selects and sets a TRACECLKIN clock (HCLK/2, HCLK/4, HCLK/8, HCLK/16).

e. Selects and sets an SMC clock (external memory controller) (HCLK/2, HCLK/4).

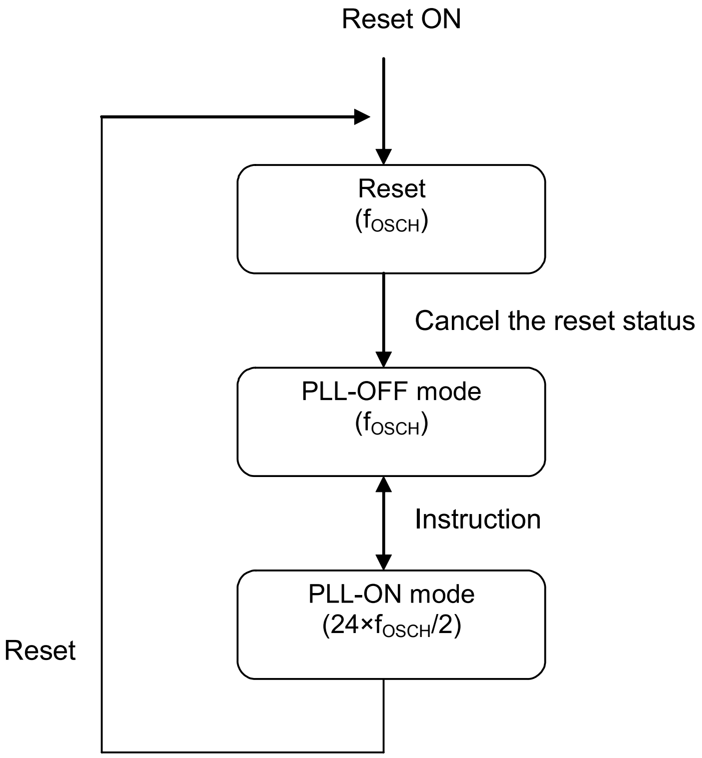


Figure 3.5-1 Clock mode status transition

3.5.2 Overview of a Block

The following shows the block diagram of the clock controller:

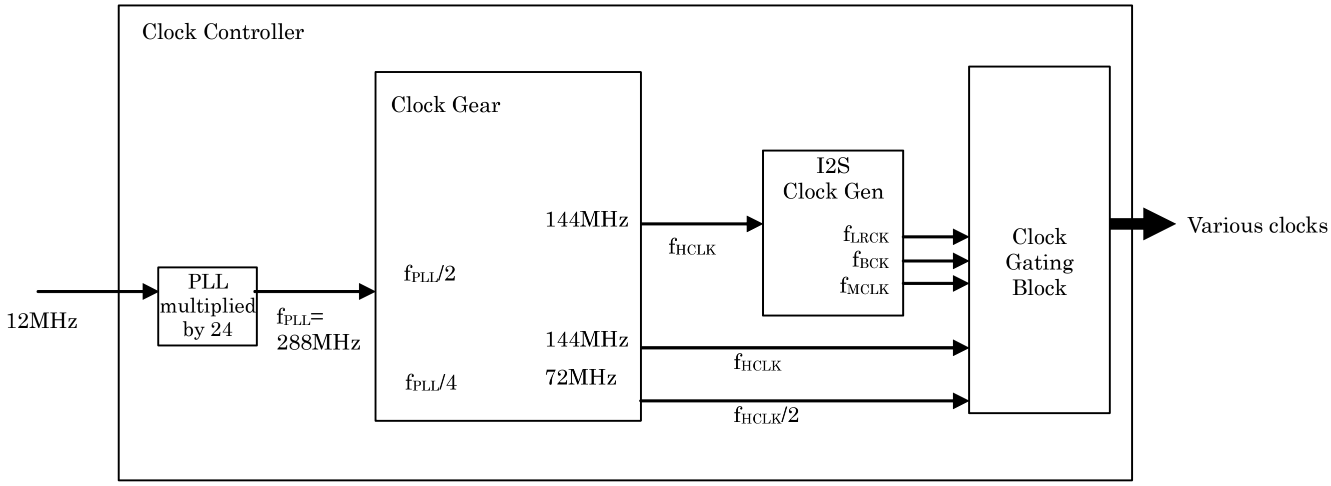


Figure 3.5-2 Diagram of whole block

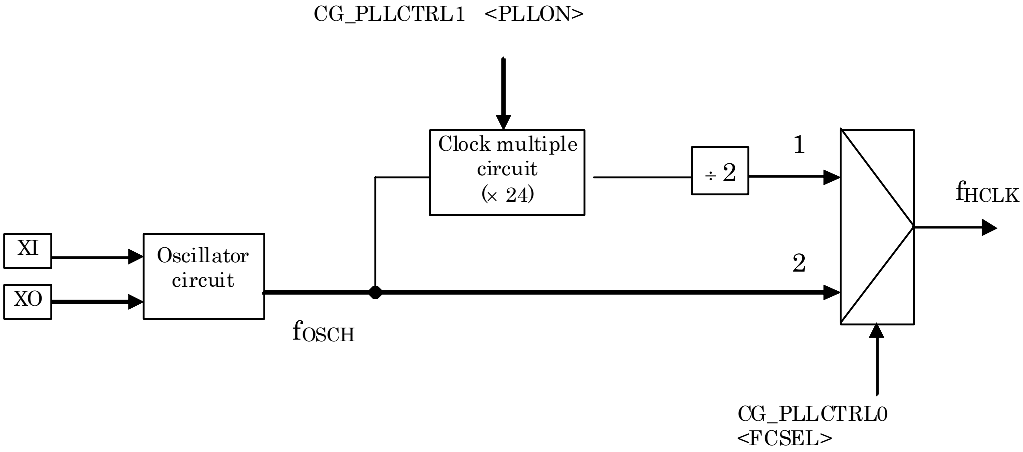


Figure 3.5-3 PLL and clock gear configuration

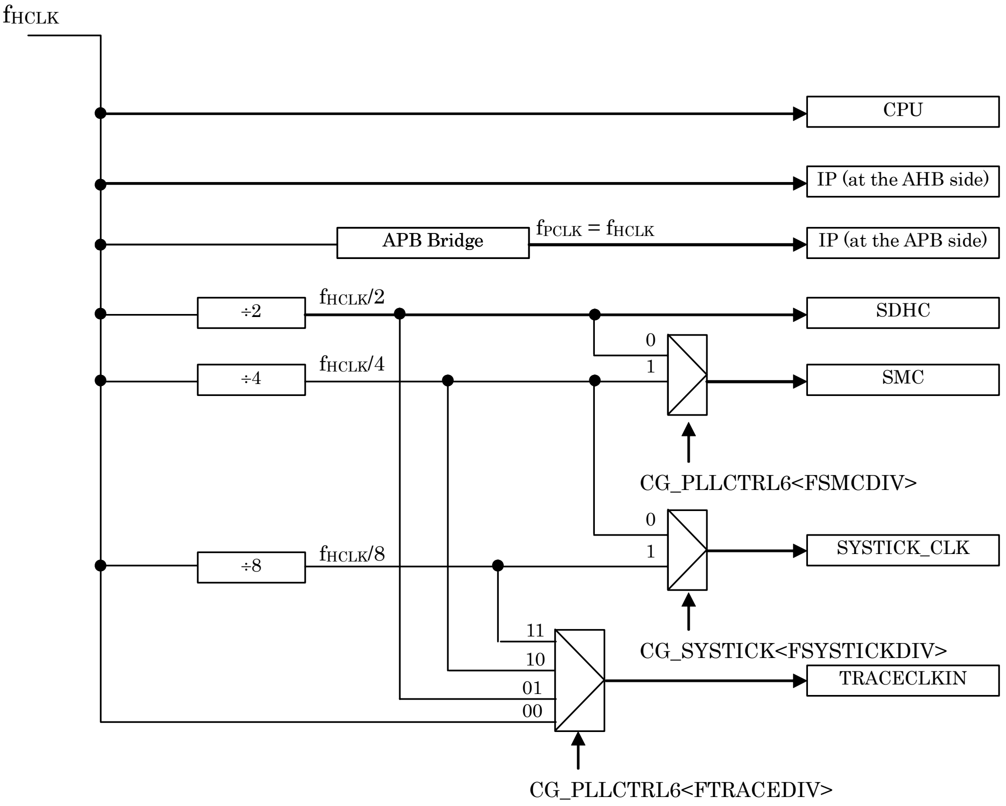


Figure 3.5-4 Clock supply configuration for each block

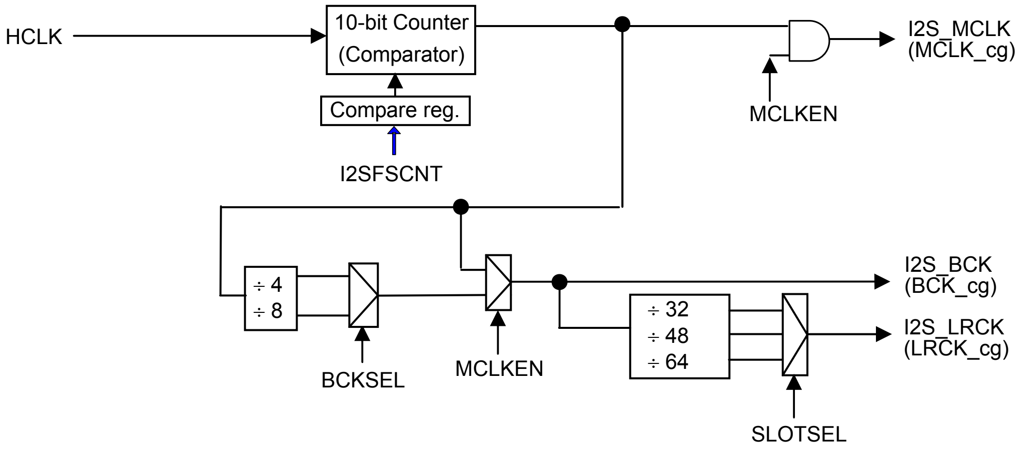


Figure 3.5-5 I2S clock generator

The clock frequency input from the X1 and X2 pins via the USB PHY block shown in Figure 3.5-3 is defined as f_{OSCH}, and the clock for peripheral IP connecting the clock selected by CG_PLLCTRL0 <FCSEL> to the CPU and AHB bus is defined as f_{HCLK}. In addition, the peripheral IP to be connected to the APB bus is defined as f_{PCLK} (This has the same frequency as f_{HCLK} but a different name).

Furthermore, a clock obtained by dividing f_{HCLK} by 2 or f_{HCLK} by 4 can be selected for the SMC (external memory controller). For the reference clock for SYSTICK, a clock obtained by dividing f_{HCLK} by 4 or f_{HCLK} by 8 can be selected.

The clock supply to TRACECLKIN when using the tracing function (TPIU) can also be selected from the following four types: f_{HCLK} without frequency division, f_{HCLK} divided by 2, f_{HCLK} divided by 4, or f_{HCLK} divided by 8.

The TRACECLK output frequency is driven at the rising edge of TRACECLKIN and thus the output frequency from the pin is as follows: TRACECLK = TRACECLKIN divided by 2.

(Note) For more information on TRACECLKIN and SYSTICK, please refer to the documents published by ARM.

3.5.3 Register Functions

The built-in registers and their functions are listed below.

base address = 0x4000_5000

Register Name	Address (base+)	Description
CG_PLLCTRL0	0x0000	CG PLL Control Register 0
CG_PLLCTRL1	0x0004	CG PLL Control Register 1
–	0x0008	reserved
CG_PLLCTRL3	0x000C	CG PLL Control Register 3
CG_PLLCTRL4	0x0010	CG PLL Control Register 4
CG_PLLCTRL5	0x0014	CG PLL Control Register 5
CG_PLLCTRL6	0x0018	CG PLL Control Register 6
CG_CLKDIS	0x0020	CG Clock Disable Register
CG_FSCTRL	0x0100	CG FS Control Register
CG_I2SFSCTRL	0x0104	CG I2S FS Control Register
–	0x0108	reserved
CG_BSIFCTRL	0x0200	CG Bit Stream Interface Control Register
CG_DMASELR	0x0300	DMA Request Select Control Register
–	0x0304	reserved
–	0x0308	reserved
–	0x030C	reserved
CG_SYSTICK	0x0310	SYSTICK Control Register

1. CG_PLLCTRL0 (CG PLL Control Register 0)

Address = (0x4000_5000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	undefined	Read undefined. Write as zero.
[0]	FCSEL	R/W	0y0	Selection to use PLL output clock 0y0: Selects PLL input clock 0y1: Selects PLL output clock

[Explanation]

a. <FCSEL>

Selects whether to use PLL output clock or not. By default after reset operation, PLL input clock is selected. After PLL initialize sequence is complete (be sure to check that the output becomes stable after the PLL lockup period has passed), select the PLL output clock whenever possible.

(Note) Do not set to "FCSEL=1" before the multiplied PLL output clock is stabilized.

2. CG_PLLCTRL1 (CG PLL Control Register 1)

Address = (0x4000_5000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	undefined	Read undefined. Write as zero.
[0]	PLLON	R/W	0y0	Signal for controlling PLL operation (Inverts and inputs the setting value to the BP pin) 0y0: OFF (bypass mode) 0y1: ON (operation mode)

[Explanation]

a. <PLLON>

By default after reset operation, the PLL bypass mode is set. During the PLL initialize sequence, switch the operation mode to "PLLON=1" if possible after the system reset is cancelled and the bypass period (100 μsec) has passed.

3. CG_PLLCTRL3 (CG PLL Control Register 3)

Address = (0x4000_5000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	undefined	Read undefined. Write as zero.
[0]	PROTECT	R/W	0y0	PLL control register protect flag 0y0: Protect OFF 0y1: Protect ON

[Explanation]

b. <PROTECT>

Indicates the access prohibition (protect ON) status for CG_PLLCTRL0 and CG_PLLCTRL1, which is set by the protect settings in CG_PLLCTRL4 and CG_PLLCTRL5.

By default after reset operation, this indicates Protect OFF.

4. CG_PLLCTRL4 (CG Control Register 4)

Address = (0x4000_5000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	ProtectCode0	R/W	0x00000000	CG_PLLCTRL0/1 protect code 0 Any value: Protect execution condition 0x0ACCE550: protect cancellation condition

[Explanation]

a. <ProtectCode0>

Sets the protect execution condition for inside of the control circuit, which is used to prohibit register access to CG_PLLCTRL0 and CG_PLLCTRL1. By default after reset operation, this sets the protect cancellation condition. Write any value in this register to set the protect execution condition. To actually execute the protect operation, write any value other than the cancellation condition in CG_PLLCRTL5 sequentially after the value written in that register. To cancel the protect operation, write the cancellation condition of "0x0ACCE550" in that register, and sequentially write the cancellation code in CG_PLLCTRL5.

5. CG_PLLCTRL5 (CG Control Register 5)

Address = (0x4000_5000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	ProtectCode1	R/W	0x00000000	CG_PLLCTRL0/1 protect code 1 Any value: Executes the protect operation 0x1ACCE551: Cancels the protect operation

[Explanation]

a. < ProtectCode1>

Executes the protect operation in the control circuit to prohibit the register access to CG_PLLCTRL0 and CG_PLLCTRL1. By default after reset operation, this sets the protect cancellation condition. To execute the protect operation, write any value in this register. To actually execute the protect operation, any value other than the cancellation condition must be written in CG_PLLCTRL4 before access to this register. To cancel the protect operation, write the cancellation condition in CG_PLLCTRL4, and sequentially write the cancellation code of "0x1ACCE551."

6. CG_PLLCTRL6 (CG PLL Control Register 6)

Address = (0x4000_5000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:19]	–	–	undefined	Read undefined. Write as zero.
[18]	TRACECLKINEN	R/W	0y0	TRACECLKIN output enable 0y0: Output disabled 0y1: Output enabled
[17:16]	FTRACEDIV	R/W	0y00	TRACECLKIN clock frequency division select 0y00: No division (fHCLK/2) 0y01: Divided by 2 (fHCLK/4) 0y10: Divided by 4 (fHCLK/8) 0y11: Divided by 8 (fHCLK/16) Note: The frequencies shown in the parentheses are the frequencies output from the TRACECLK pin.
[15:6]	–	–	undefined	Read undefined. Write as zero.
[5]	FSMCDIV	R/W	0y0	Selects the number of divisions for the SMCCLK clock 0y0: Divides by 2 0y1: Divides by 4
[4:0]	–	–	undefined	Read undefined. Write as zero.

[Explanation]

a. <TRACECLKINEN>

Controls the output of the TRACECLKIN output clock supplied for debug use. After reset, output is disabled by default.

b. <FTRACEDIV>

Selects the frequency-division circuit for the TRACECLK output clock supplied for debug use.

After reset, "No division" is selected by default. For example, if fHCLK operates at 144 MHz in this setting, the TRACECLK output clock is output at 72 MHz.

c. <FSMCDIV>

Selects a divider circuit for the SCM_CLK output clock to be supplied to the SMC. By default after reset operation, "Divides by 2" is selected.

7. CG_CLKDIS (CG Clock Disable Register)

Address = (0x4000_5000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:20]	–	–	undefined	Read undefined. Write as zero.
[19]	DMACHDIS	R/W	0y0	Disables the bus clock HCLK output for the DMAC 0y0: Enables the output 0y1: Disables the output
[18]	USBHDIS	R/W	0y0	Disables the bus clock HCLK output for the USB 0y0: Enables the output 0y1: Disables the output
[17]	Reserved	R/W	0y0	Read undefined. Write as one.
[16]	SDHCHDIS	R/W	0y0	Disables the bus clock HCLK output for the SDHC 0y0: Enables the output 0y1: Disables the output
[15]	I2SHDIS	R/W	0y0	Disables the bus clock HCLK output for I2S 0y0: Enables the output 0y1: Disables the output
[14]	VZOPDIS	R/W	0y0	Disables the bus clock PCLK output for the clock generator dedicated for the BSIF mode 0y0: Enables the output 0y1: Disables the output
[13]	TMR67PDIS	R/W	0y0	Disables the bus clock PCLK output for the TimerCH6/CH7 0y0: Enables the output 0y1: Disables the output
[12]	TMR45PDIS	R/W	0y0	Disables the bus clock PCLK output for the TimerCH4/CH5 0y0: Enables the output 0y1: Disables the output
[11]	TMR23PDIS	R/W	0y0	Disables the bus clock PCLK output for the TimerCH2/CH3 0y0: Enables the output 0y1: Disables the output
[10]	TMR01PDIS	R/W	0y0	Disables the bus clock PCLK output for the TimerCH0/CH1 0y0: Enables the output 0y1: Disables the output
[9]	I2C1PDIS	R/W	0y0	Disables the bus clock PCLK output for the I2C CH1 0y0: Enables the output 0y1: Disables the output
[8]	I2C0PDIS	R/W	0y0	Disables the bus clock PCLK output for the I2C CH0 0y0: Enables the output 0y1: Disables the output

Bit	Bit Symbol	Type	Reset Value	Description
[7]	SSP3PDIS	R/W	0y0	Disables the bus clock PCLK output for the SSPCH3 0y0: Enables the output 0y1: Disables the output
[6]	SSP2PDIS	R/W	0y0	Disables the bus clock PCLK output for the SSPCH2 0y0: Enables the output 0y1: Disables the output
[5]	SSP1PDIS	R/W	0y0	Disables the bus clock PCLK output for the SSPCH1 0y0: Enables the output 0y1: Disables the output
[4]	SSP0PDIS	R/W	0y0	Disables the bus clock PCLK output for the SSPCH0 0y0: Enables the output 0y1: Disables the output
[3]	UART3PDIS	R/W	0y0	Disables the bus clock PCLK output for the UARTCH3 0y0: Enables the output 0y1: Disables the output
[2]	UART2PDIS	R/W	0y0	Disables the bus clock PCLK output for the UARTCH2 0y0: Enables the output 0y1: Disables the output
[1]	UART1PDIS	R/W	0y0	Disables the bus clock PCLK output for the UARTCH1 0y0: Enables the output 0y1: Disables the output
[0]	UART0PDIS	R/W	0y0	Disables the bus clock PCLK output for the UARTCH0 0y0: Enables the output 0y1: Disables the output

[Explanation]

a. <DMACHDIS>

Controls the HCLK output, which is a bus clock for the DMAC. By default after reset operation, the output is enabled.

b. <USBHDIS>

Controls the HCLK output, which is a bus clock for the USB. By default after reset operation, the output is enabled.

c. <SDHCHDIS>

Controls the HCLK output, which is a bus clock for the SDHC. By default after reset operation, the output is enabled.

d. <I2SHDIS>

Controls the HCLK output, which is a bus clock for I2S. By default after reset operation, the output is enabled.

e. <VZOPDIS>

Controls the PCLK output, which is a bus clock for the VZO. By default after reset operation, the output is enabled.

f. <TMR67PDIS>

Controls the PCLK output, which is a bus clock for the Timer CH6/CH7. By default after reset operation, the output is enabled.

g. <TMR45PDIS>

Controls the PCLK output, which is a bus clock for the Timer CH4/CH5. By default after reset operation, the output is enabled.

h. <TMR23PDIS>

Controls the PCLK output, which is a bus clock for the Timer CH2/CH3. By default after reset operation, the output is enabled.

i. <TMR01PDIS>

Controls the PCLK output, which is a bus clock for the Timer CH0/CH1. By default after reset operation, the output is enabled.

j. <I2C1PDIS>

Controls the PCLK output, which is a bus clock for the I2C CH1. By default after reset operation, the output is enabled.

k. <I2C0PDIS>

Controls the PCLK output, which is a bus clock for the I2C CH0. By default after reset operation, the output is enabled.

l. <SSP3PDIS>

Controls the PCLK output, which is a bus clock for the SSP CH3. By default after reset operation, the output is enabled.

m. <SSP2PDIS>

Controls the PCLK output, which is a bus clock for the SSP CH2. By default after reset operation, the output is enabled.

n. <SSP1PDIS>

Controls the PCLK output, which is a bus clock for the SSP CH1. By default after reset operation, the output is enabled.

o. <SSP0PDIS>

Controls the PCLK output, which is a bus clock for the SSP CH0. By default after reset operation, the output is enabled.

p. <UART3PDIS>

Controls the PCLK output, which is a bus clock for the UART CH3. By default after reset operation, the output is enabled.

q. <UART2PDIS>

Controls the PCLK output, which is a bus clock for the UART CH2. By default after reset operation, the output is enabled.

r. <UART1PDIS>

Controls the PCLK output, which is a bus clock for the UART CH1. By default after reset operation, the output is enabled.

s. <UART0PDIS>

Controls the PCLK output, which is a bus clock for the UART CH0. By default after reset operation, the output is enabled.

(Note 1)

The bus clock for each IP can be stopped (output can be disabled) by this register setting without conditions if the reset operation has just been cancelled and no request and interrupt occurs to the IP to be stopped. However, if a request or interrupt occurs in the normal operation, all of those factors must be cleared before stopping the clock (disabling the output). After the clock is stopped, any register settings in the stopped IP that have already been used will be held.

(Note 2)

Before accessing each IP register, enable the clock output of each IP. When accessing the IP whose clock was stopped by this register setting, the default slave returns an error response and an exceptional process occurs.

8. CG_FSCTRL (CG FS Control Register)

Address = (0x4000_5000) + 0x0100

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	undefined	Read undefined. Write as zero.
[3]	BCKSEL	R/W	0y0	BCK frequency division select (Enabled when MCLKEN=1) 0y0: MCLK divided by 4 0y1: MCLK divided by 8
[2]	MCLKEN	R/W	0y0	MCLK use select 0y0: MCLK not used 0y1: MCLK used
[1:0]	SLOTSEL	R/W	0y00	LRCK output clock use select 0y00: 64 Slot 0y01: 48 Slot 0y10: 32 Slot 0y11: 64 Slot

[Explanation]

a. <BCKSEL>

Selects the frequency division ratio of the BCK clock when generating a clock for I2S and using the MCLK clock. After reset, "MCLK clock divided by 4" is selected by default. To enable this bit, MCLKEN=1 needs to be set.

b. <MCLKEN>

Writes "1" to this bit when generating a clock for I2S and using the MCLK clock. After reset, "MCLK not used (MCLKEN=0)" is selected by default.

c. <SLOTSEL>

Sets the number of slots for the LRCK output clock when generating a clock for I2S.

After reset, "64 Slot (SLOTSEL=0y00)" is selected by default.

When 0y11 is written for this bit, 64 slots is set like the case of 0y00.

9. CG_I2SFSCTRL (CG I2S FS Control Register)

Address = (0x4000_5000) + 0x0104

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	undefined	Read undefined. Write as zero.

[9:0]	I2SFSCNT	R/W	0y0000000000	Selects the generation clock by setting the I2S 1/2FS generation counter and setting the MCLKEN bit of CG_FSCTRL.
				MCLKEN=0: BCK count setting
				MCLKEN=1: MCLK count setting

[Explanation]

a. <I2SFSCNT>

Sets the frequency division counter value (10-bit) for 1/2FS clock generation to this register from the system clock when generating the basic clock for I2S. After reset, "I2SFSCNT=0" is selected by default. That is:

When MCLKEN=0

1/(I2SFSCNT * 2) = BCK frequency

When MCLKEN=1

1/(I2SFSCNT * 2) = MCLK frequency

For more information on how to calculate the counter value, and the setting values, refer to Section 3.20 "I2S."

10. CG_BSIFCTRL (CG Bit Stream Interface Control Register)

Address = (0x4000_5000) + 0x0200

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	undefined	Read undefined. Write as zero.
[6]	VZLRCKOEN	R/W	0y0	Enables LRCK output for the BSIF mode 0y0: Selects SP3 FSS 0y1: Selects LRCK
[5:4]	VZFSSEL	R/W	0y00	Selects FS for the BSIF mode 0y00: 64 Slot 0y01: 48 Slot 0y10: 32 Slot 0y11: 64 Slot
[3]	VZLRCKTGL	R/W	0y0	Inverts LRCK output for the BSIF mode 0y0: Positive logic 0y1: Negative logic (inverted)
[2]	VZBCKTGL	R/W	0y0	Inverts BCK output for the BSIF mode 0y0: Positive logic 0y1: Negative logic (inverted)
[1]	VZINIPHS	R/W	0y0	Sets an initial LRCK phase for the BSIF mode 0y0: "0" when stopped 0y1: "1" when stopped
[0]	VZSTART	R/W	0y0	LRCK generation module start for BSIF mode 0y0: Stopped 0y1: Starts the operation

[Explanation]

a. <VZLRCKOEN>

Selects the LRCK output for BSIF mode. By default after reset operation, the SP3FSS signal is selected. When 1 is written for this bit, the LRCK output clock for the BSIF mode is selected. In this case, an appropriate value must be set for each bit of this register.

b. <VZFSSEL>

To generate the LRCK output clock (FS) for the BSIF mode, set the number of slots for the LRCK output clock. By default after reset operation, VZFSSEL=0y00 (64 Slot) is set. When 0y11 is written for this bit, 64 slots is set like the case of 0y00.

c. <VZLRCKTGL>

To output the LRCK output clock (FS) for the BSIF mode, set the output logic of the LRCK output clock. By default after reset operation, the output is of positive logic.

d. <VZBCKTGL>

To output the BCK output clock (FS) for the BSIF mode, set the output logic of the BCK output clock. By default after reset operation, the output is of positive logic.

e. <VZINIPHS>

To output the LRCK output clock (FS) for the BSIF mode, set the initial phase of the LRCK output clock. By default after reset operation, the output is 0.

f. <VZSTART>

To output the LRCK output clock (FS) for the BSIF mode, set the output start timing of the LRCK output clock. By default after reset operation, the output is stopped.

11. CG_DMASELR (DMA request Select Control Register)

Address = (0x4000_5000) + 0x0300

Bit	Bit Symbol	Type	Reset Value	Description
[31:20]	–	–	undefined	Read undefined. Write as zero.
[19:16]	DMASEL	R/W	0y0000	Selects 2 out of 4 modules: SSP CH2/CH3 and UART CH2/CH3 *Refer to Table 2.
[15:2]	Reserved	–	undefined	Write as zero. Read as zero.
[1]	SP1U1DMA	R/W	0y0	Selects a DMA request for SSP CH1/UART CH1 0y0: Selects the SSP CH1 DMA request 0y1: Selects the UART CH1 DMA request
[0]	SP0U0DMA	R/W	0y0	Selects a DMA request for SSP CH0/UART CH0 0y0: Selects the SSP CH0 DMA request 0y1: Selects the UART CH0 DMA request

[Explanation]

d. <DMASEL>

Sets to select 2 out of 4 modules (SSP CH2/CH3 and UART CH2/CH3) that share the DMA request signal. By default after reset operation, SSP CH2 and SSP CH3 are selected as shown in Table 2 below. There are restrictions on combinations of the selections as shown in Table 1. Selections that are not in the table cannot be selected.

Table 1: Combinations of 4 to 2 selections

Setting ->	NG	NG	NG	OK	NG	OK	OK	NG	NG	OK	OK	NG	OK	NG	NG	NG
SSP2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
SSP3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
UART2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
UART3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

▨ : It is also impossible to select nothing or a single factor only.

Table 2: DMASEL correspondence table

DMASEL	out port2	out port3
0000	SSP2	SSP3
0001		
0010	SSP2	UART2
0011	SSP3	UART2
0100	SSP2	UART3
0101	SSP3	UART3
0110	UART2	UART3
0111		
1000	SSP3	SSP2
1001		
1010	UART2	SSP2
1011	UART2	SSP3
1100	UART3	SSP2
1101	UART3	SSP3
1110	UART3	UART2
1111		

e. <SP1U1DMA>

Sets to select 1 out of 2 modules (SSP CH1 and UART CH1) that share the DMA request signal. By default after reset operation, SSP CH1 is selected.

f. <SP0U0DMA>

Sets to select 1 out of 2 modules (SSP CH0 and UART CH0) that share the DMA request signal. By default after reset operation, SSP CH0 is selected.

12. CG_SYSTICK (SYSTICK Control Register)

Address = (0x4000_5000) + 0x0310

Bit	Bit Symbol	Type	Reset Value	Description
[31:28]	–	–	undefined	Read undefined. Write as zero.
[27]	FSYSTICKDIV	R/W	0y0	Selects magnification ratio of the SYSTICK clock 0y0: Divides the system clock by 4 0y1: Divides the system clock by 8
[26]	SYSTICKDIS	R/W	0y0	Disables the output of the SYSTICK clock 0y0: Disables the output 0y1: Enables the output
[25]	NOREF	R/W	0y0	Sets the reference of the SYSTICK clock 0y0: External clock used (with reference = SYSTICK_CLK) 0y1: System clock used (without reference = CPU clock)
[24]	SKEW	R/W	0y0	Sets the skew of the SYSTICK clock 0y0: Without skew 0y1: With skew
[23:0]	TENMS	R/W	0x000000	Calibration reload value for SYSTICK

[Explanation]

a. <FSYSTICKDIV>

To generate the SYSTICK clock (for the system timer), select the frequency division rate from the system clock. By default after reset operation, "Divides the system clock by 4" is selected. To use the SYSTICK clock, SYSTICKDIS=1 and NOREF=1 must be set.

b. <SYSTICKDIS>

Controls the output of the SYSTICK clock. By default after reset operation, the output is disabled.

c. <NOREF>

Sets the reference of the SYSTICK clock. By default after reset operation, "With reference (Uses the external clock)" is selected. After reset, "With reference (External clock used)" is selected by default, where CG_SYSTICK< FSYSTICKDIV> is enabled and the system clock divided by 4 or 8 is used as the SYSTICK clock. If "1" is set, the system clock (HCLK = CPU clock) will be directly used as the SYSTICK clock.

d. <SKEW>

Sets the skew of the SYSTICK clock to the calibration value. By default after reset operation, "Without skew" is selected.

e. <TENMS>

Sets the calibration reload value for the SYSTICK clock. By default after reset operation, 0x000000 is set, which disables the counter interrupt by the expected SYSTICK clock and the COUNTFLAG operation.

3.5.4 PLL Initialize Sequence

This section explains the PLL initialize sequence. When the PLL macro stored in this block is used, the following initialize sequence is required during the system power-on.

■ **PLL Initialize Sequence**

1) Power On Reset

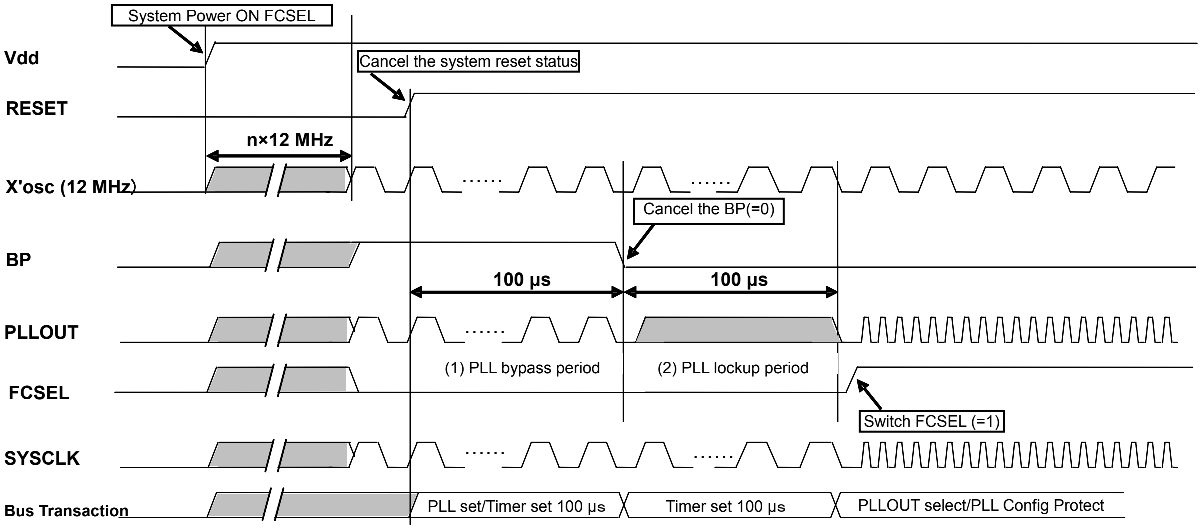


Figure 3.5-6 PLL initialize sequence (during power-on)

After the input clock from the external oscillator is stabilized, FCSEL=0 and BP=1 (default values when reset) are set while RESET is set to 0. When the system reset is cancelled (RESET=1), a PLL bypass period of approx. 100 μsec is taken. At the same time, 100 μsec is counted by the built-in timer function* (settings must be made in the software separately). The system clock at this time operates at the clock frequency from the external oscillator (12 MHz). After 100 μsec has passed, set BP to 0 (CG_PLLCTRL1=1) to move to the PLL lockup period. Sequentially, 100 μsec is counted by the built-in timer as described above* (also, settings must be made in the software separately). When the PLLOUT (output clock) is stabilized at the target frequency, set FCSEL to 1 (CG_PLLCTRL0=1) to switch the system clock to PLLOUT (output clock).

Note that when the system operates normally and the system reset is executed due to a internal factor of the system, the following initialize sequence is required.

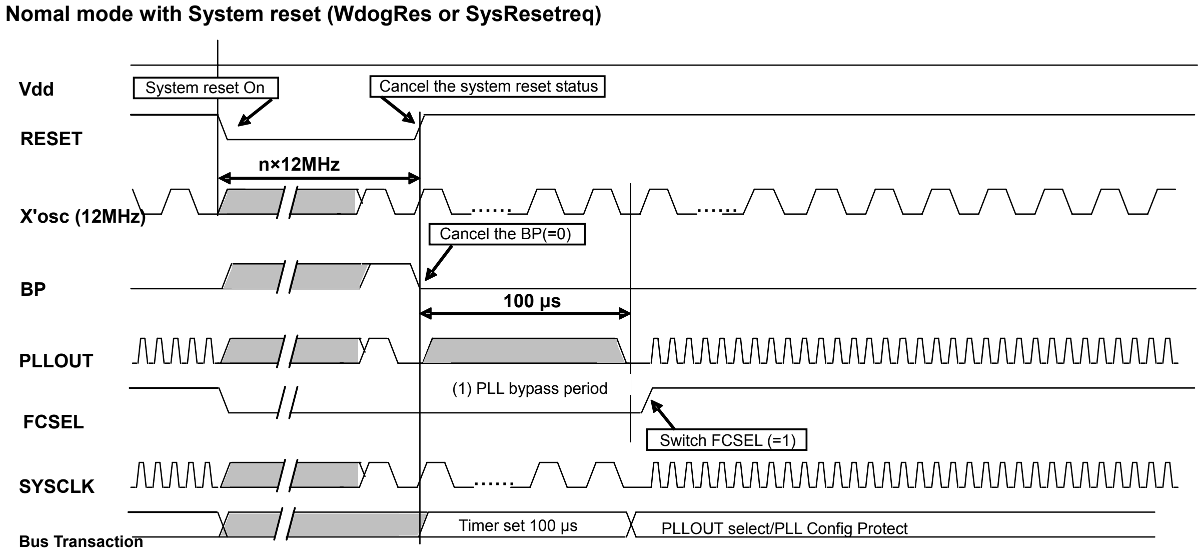


Figure 3.5-7 PLL initialize sequence (when an internal reset factor occurs during normal operation)

When the system reset is asserted due to an internal reset factor, FCSEL=0 and BP=1 (default values when reset) are set while RESET is set to 0. In this case, the PLL bypass period (approx. 100 μsec) can be omitted because the power is already on and stable. When the system reset is cancelled (RESET=1) and BP is set to 0 (CG_PLLCTRL1=1), the PLL lockup period of approx. 100 μsec is taken. During this time, 100 μsec is counted by the built-in timer* (settings must be made in the software separately). When the PLLOUT (output clock) is stabilized at the target frequency, set FCSEL to 1 (CG_PLLCTRL0=1) to switch the system clock to PLLOUT (output clock).

3.5.5 Register for Controlling the Protect Operation

In this block, write access to 3 control registers for PLL setting (CG_PLLCTRL0, CG_PLLCTRL1) can be protected by sequentially writing any values in the CG_PLLCTRL4 and CG_PLLCTRL5 registers (if access to a module other than the CG modules occurs between CG_PLLCTRL4 and CG_PLLCTRL5, both the protect execution and cancellation operations become available). Once write access to the CG_PLLCTRL0/1/2 registers is protected, write access remains unavailable unless the code values used for cancellation are sequentially written to CG_PLLCTRL4 and CG_PLLCTRL5 (if a value is written during the protect mode, it will not be valid and the value set before the protect operation will be held). To determine whether or not the protect operation is valid, read a value from the CG_PLLCTRL3 (read only) protect status register (1: Protect status, 0: Non-protect status). The correspondence table is shown below.

Table 3.5.5-1 List of setting values for controlling the protect operation

Control register name	Write data for protection	Write data for cancelling protection
CG_PLLCTRL4	Any 32-bit value other than the value for cancellation	0 × 0ACCE550
CG_PLLCTRL5	Any 32-bit value other than the value for cancellation	0 × 1ACCE551

The sequential access timings to the protect control registers, CG_PLLCTRL4 and CG_PLLCTRL5, (when protected and when not protected) and their circuit configurations are shown below. The following shows the access timings.

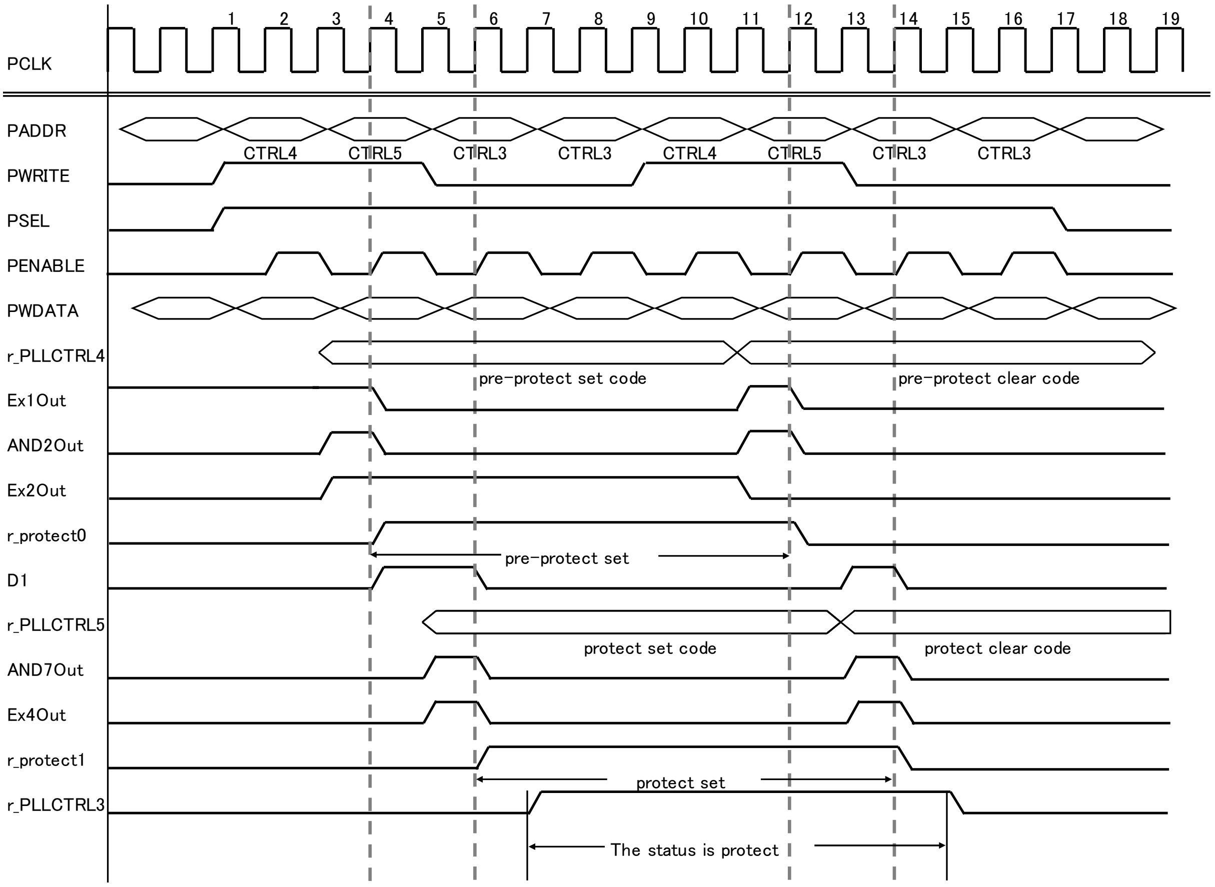


Figure 3.5-8 Protect control register access timings

As described above, CG_PLLCTRL3 (read only) is provided as the register dedicated to check the protect status. It requires an additional 1 cycle for the register to detect the protect status after it becomes valid. Therefore, the status bit in CG_PLLCTRL3 that is read right after the protect operation does not reflect the actual status. To obtain the actual status, execute polling to wait for a while, or wait for 2 bus cycles before access.

3.5.6 Clock Gating

This block has the gating function for each clock output divided by using CG. Outputs can be controlled via the control register (CG_CLKDIS) as required. By default after reset operation, all clocks are output. The control signals selected in the control registers are used as the output signals for external, which are used for recognition during clock gating.

3.5.7 PLL ON/OFF Sequence

A sample sequence during PLL ON/OFF setting is shown below.

1. PLL ON sequence

```
 // PLL initialization (Initializes PLL)
 Timer0Load    = 0x000004bc; // 1212 cycle set (Counts a period of 100 μs by the built-in timer with fOSCH=12
MHz)
 Timer0Control = 0x000000a3; // timer enable

 // PLL BP's period = 100 us
 // Wait for interrupt, it will be expected to TIMINT0 as 0
 __WFI(); (Waits for interrupt by timer count-down & Core sleep mode)

 // BP set to 0
 CG_PLLCTRL1 = 0x00000001; (Cancels the PLL bypass mode)

 // 2nd Timer0 Set for dummy 1212 cycle count
 Timer0Load    = 0x000004bc; // 1212 cycle set (Counts a period of 100 μs by the built-in timer with fOSCH=12
MHz)
 Timer0Control = 0x000000a3; // timer enable

 // PLL Lockup's period = 100us
 // Wait for interrupt, it will be expected to TIMINT0 as 0
 __WFI(); (Waits for interrupt by timer count-down & Core sleep mode)

 // HCLK is switched to PLLCK
 CG_PLLCTRL0 = 0x00000001; (Switches to PLL output)
```

2. PLL OFF sequence

```
 // PLL output switch HCLK to OSCH (Switches PLL outputs)
 CG_PLLCTRL0 = 0x00000000; (Switches to PLL output)

 // BP set to 1
 CG_PLLCTRL1 = 0x00000000; (Turns ON the PLL bypass mode)

 // Timer0 Set for dummy  M cycle count
 Timer0Load     = 0xXXXXXXXX; // M cycle set (Counts a period of N μs by the built-in timer with $f_{OSCH}$=12
MHz)
 Timer0Control = 0x000000a3; // timer enable

 // PLL BP's period = Nus
 // Wait for interrupt, it will be expected to TIMINT0 as 0
 __WFI(); (Waits for interrupt by timer count-down & Core sleep mode)
```

3.7 Interrupts (exceptions)

3.7.1 Function Overview

- Supports 29 types of interrupt sources.
- Fixes to the default priority out of 29 stages in the H/W (valid when the S/W levels are the same).
- Can set the interrupt level out of 16 stages.
- The offset register can change the base address of the vector table ("0x00000000" by default).

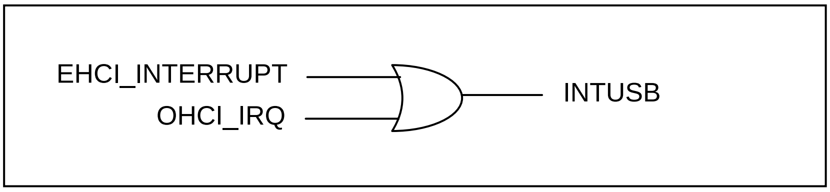
Refer to the documentations published by ARM for details on the interrupt.

3.7.2 Interrupt Source List

Table 3.7.1 Interrupt source list

INTISR No.	Interrupt name	Description	Offset Address (base+)
INTSR[0]	INTWDT	Watchdog timer interrupt	0x00000040
INTSR[1]	INTDMACERR	DMA transfer error interrupt	0x00000044
INTSR[2]	INTDMACTC	DMA transfer end Interrupt	0x00000048
INTSR[3]	INT0	External pin interrupt 0	0x0000004C
INTSR[4]	INT1	External pin interrupt 1	0x00000050
INTSR[5]	INT2	External pin interrupt 2	0x00000054
INTSR[6]	INT3	External pin interrupt 3	0x00000058
INTSR[7]	INTTMR0	Timer 0 interrupt	0x0000005C
INTSR[8]	INTTMR1	Timer 1 interrupt	0x00000060
INTSR[9]	INTTMR2	Timer 2 interrupt	0x00000064
INTSR[10]	INTTMR3	Timer 3 interrupt	0x00000068
INTSR[11]	INTTMR4	Timer 4 interrupt	0x0000006C
INTSR[12]	INTTMR5	Timer 5 interrupt	0x00000070
INTSR[13]	INTTMR6	Timer 6 interrupt	0x00000074
INTSR[14]	INTTMR7	Timer 7 interrupt	0x00000078
INTSR[15]	INTSSP0	SSPch0 integrated interrupt	0x0000007C
INTSR[16]	INTSSP1	SSPch1 integrated interrupt	0x00000080
INTSR[17]	INTSSP2	SSPch2 integrated interrupt	0x00000084
INTSR[18]	INTSSP3	SSPch3 integrated interrupt	0x00000088
INTSR[19]	INTUART0	UARTch0 integrated interrupt	0x0000008C
INTSR[20]	INTUART1	UARTch1 integrated interrupt	0x00000090
INTSR[21]	INTUART2	UARTch2 integrated interrupt	0x00000094
INTSR[22]	INTUART3	UARTch3 integrated interrupt	0x00000098
INTSR[23]	INTI2C0	I2Cch0 interrupt	0x0000009C
INTSR[24]	INTI2C1	I2Cch1 interrupt	0x000000A0
INTSR[25]	INTUSB	USB (EHCI, OHCI) integrated interrupt	0x000000A4
INTSR[26]	INTSDHC	SDHost interrupt	0x000000A8
INTSR[27]	INTI2S	I2S interrupt	0x000000AC
INTSR[28]	–	reserved	0x000000B0
INTSR[29]	INTADC	ADC interrupt	0x000000B4
INTSR[30]	Reserved	–	0x000000B8
:	:	:	:
INTSR[239]	Reserved	–	0x000003FC

*) The USB interrupt source is a merged interrupt where the interrupt requests from the masters of each OHCI and EHCI are ORed.



3.7.3 SFR

The following lists the SFRs:

Table 3.7.2 SFR list (1/2) base address = 0xE000_0000

Register Name	Address (base+)	Description
Reserved	0xE000	Reserved
Interrupt Control Type Register	0xE004	Interrupt Control Type Register
SysTick Control and Status Register	0xE010	SysTick Control and Status Register
SysTick Reload Value Register	0xE014	SysTick Reload Value Register
SysTick Current Value Register	0xE018	SysTick Current Value Register
SysTick Calibration Value Register	0xE01C	SysTick Calibration Value Register
Reserved	0xE020 - E0FF	Reserved
Irq 0 to 31 Set Enable Register	0xE100	Irq 0 to 31 Set Enable Register
Reserved	0xE104 - E17F	Reserved
Irq 0 to 31 Clear Enable Register	0xE180	Irq 0 to 31 Clear Enable Register
Reserved	0xE184 - E1FF	Reserved
Irq 0 to 31 Set Pending Register	0xE200	Irq 0 to 31 Set Pending Register
Reserved	0xE204 - E27F	Reserved
Irq 0 to 31 Clear Pending Register	0xE280	Irq 0 to 31 Clear Pending Register
Reserved	0xE284 - E2FF	Reserved
Irq 0 to 31 Active Bit Register	0xE300	Irq 0 to 31 Active Bit Register
Reserved	0xE304 - E3FF	Reserved
Irq 0 to 31 Priority Register	0xE400	Irq 0 to 3 Priority Register
Irq 0 to 31 Priority Register	0xE404	Irq 4 to 7 Priority Register
Irq 0 to 31 Priority Register	0xE408	Irq 8 to 11 Priority Register
Irq 0 to 31 Priority Register	0xE40C	Irq 12 to 15 Priority Register
Irq 0 to 31 Priority Register	0xE410	Irq 16 to 19 Priority Register
Irq 0 to 31 Priority Register	0xE414	Irq 20 to 23 Priority Register
Irq 0 to 31 Priority Register	0xE418	Irq 24 to 27 Priority Register
Irq 0 to 31 Priority Register	0xE41C	Irq 28 to 31 Priority Register
Reserved	0xE420 - ECFF	Reserved
CPUID Base Register	0xED00	CPUID Base Register
Interrupt Control State Register	0xED04	Interrupt Control State Register
Vector Table Offset Register	0xED08	Vector Table Offset Register
Application Interrupt/Reset Control Register	0xED0C	Application Interrupt/Reset Control Register
System Control Register	0xED10	System Control Register
Configuration Control Register	0xED14	Configuration Control Register
System Handlers 4-7 Priority Register	0xED18	System Handlers 4-7 Priority Register
System Handlers 8-11 Priority Register	0xED1C	System Handlers 8-11 Priority Register
System Handlers 12-15 Priority Register	0xED20	System Handlers 12-15 Priority Register
System Handler Control and State Register	0xED24	System Handler Control and State Register
Configurable Fault Status Registers	0xED28	Configurable Fault Status Registers
Hard Fault Status Register	0xED2C	Hard Fault Status Register
Debug Fault Status Register	0xED30	Debug Fault Status Register
Mem Manage Address Register	0xED34	Mem Manage Address Register
Bus Fault Address Register	0xED38	Bus Fault Address Register
Auxiliary Fault Status Register	0xED3C	Auxiliary Fault Status Register

Table 3.7.2 SFR list (2/2)

Register Name	Address (base+)	Description
Processor Feature register0	0xED40	Processor Feature register0
Processor Feature register1	0xED44	Processor Feature register1
Debug Feature register0	0xED48	Debug Feature register0
Auxiliary Feature register0	0xED4C	Auxiliary Feature register0
Memory Model Feature register0	0xED50	Memory Model Feature register0
Memory Model Feature register1	0xED54	Memory Model Feature register1
Memory Model Feature register2	0xED58	Memory Model Feature register2
Memory Model Feature register3	0xED5C	Memory Model Feature register3
ISA Feature register0	0xED60	ISA Feature register0
ISA Feature register1	0xED64	ISA Feature register1
ISA Feature register2	0xED68	ISA Feature register2
ISA Feature register3	0xED6C	ISA Feature register3
ISA Feature register4	0xED70	ISA Feature register4
Reserved	0xED74 - EEFF	Reserved
Software Trigger Interrupt Register	0xEF00	Software Trigger Interrupt Register
Reserved	0xEF04 - EFCF	Reserved
Peripheral identification register	0xEFD0	Vector Priority 16 Register
Peripheral identification register	0xEFD4	Vector Priority 17 Register
Peripheral identification register	0xEFD8	Vector Priority 18 Register
Peripheral identification register	0xEFDC	Reserved
Peripheral identification register Bits 7:0	0xEFE0	Vector Priority 20 Register
Peripheral identification register Bits 15:8	0xEFE4	Vector Priority 21 Register
Peripheral identification register Bits 23:16	0xEFE8	Vector Priority 22 Register
Peripheral identification register Bits 31:24	0xEFEC	Vector Priority 23 Register
Component identification register Bits 7:0	0xEFF0	Reserved
Component identification register Bits 15:8	0xEFF4	Reserved
Component identification register Bits 23:16	0xEFF8	Vector Priority 26 Register
Component identification register Bits 31:24	0xEFFC	Vector Priority 27 Register

3.8 DMA Controller (DMAC)

The DMA of the TMPM320C1D is controlled by the internal DMA controller and DMA request select registers (See the CG_DMASELR register in Section 3.5).

3.8.1 Function Overview

The table below lists its major functions.

Table 3.8.1 DMA controller functions

Item	Function		Overview
Number of channels	8ch		
	Hardware start		Supports 14 types of DMA requests for peripheral IPs. See Table 3.8.2.
	Software start		Started with a write to the DMACSoftBReq register.
Bus master	32bit×2 (AHB)		DMA1, DMA2
Priority	DMA channel 0 (high) to DMA channel 7 (low)		Fixed by hardware
FIFO	4word × 8ch		
Bus width	8/16/32bit		Settable individually for transfer source and destination
Burst size	1/4/8/16/32/64/128/256		
Number of transfers	up to 4095		
Address	Transfer source address	incr / no-incr	It is possible to specify whether Source and Destination addresses should increment or should not increment (should be fixed). (Address wrapping is not supported.)
	Transfer destination address	incr / no-incr	
Endian	Only little endian is supported.		
Transfer type	Peripheral circuit (register) → peripheral circuit (register) Peripheral circuit (register) → memory Memory → peripheral circuit (register) Memory → memory		When "memory → memory" is selected, hardware start for DMA startup is not supported. See the DMACCxConfiguration register for more information.
Interrupt function	Transfer end interrupt Error interrupt		
Special Function	Scatter/gather function		

* 1 word = 32 bits

3.8.2 Block Diagram

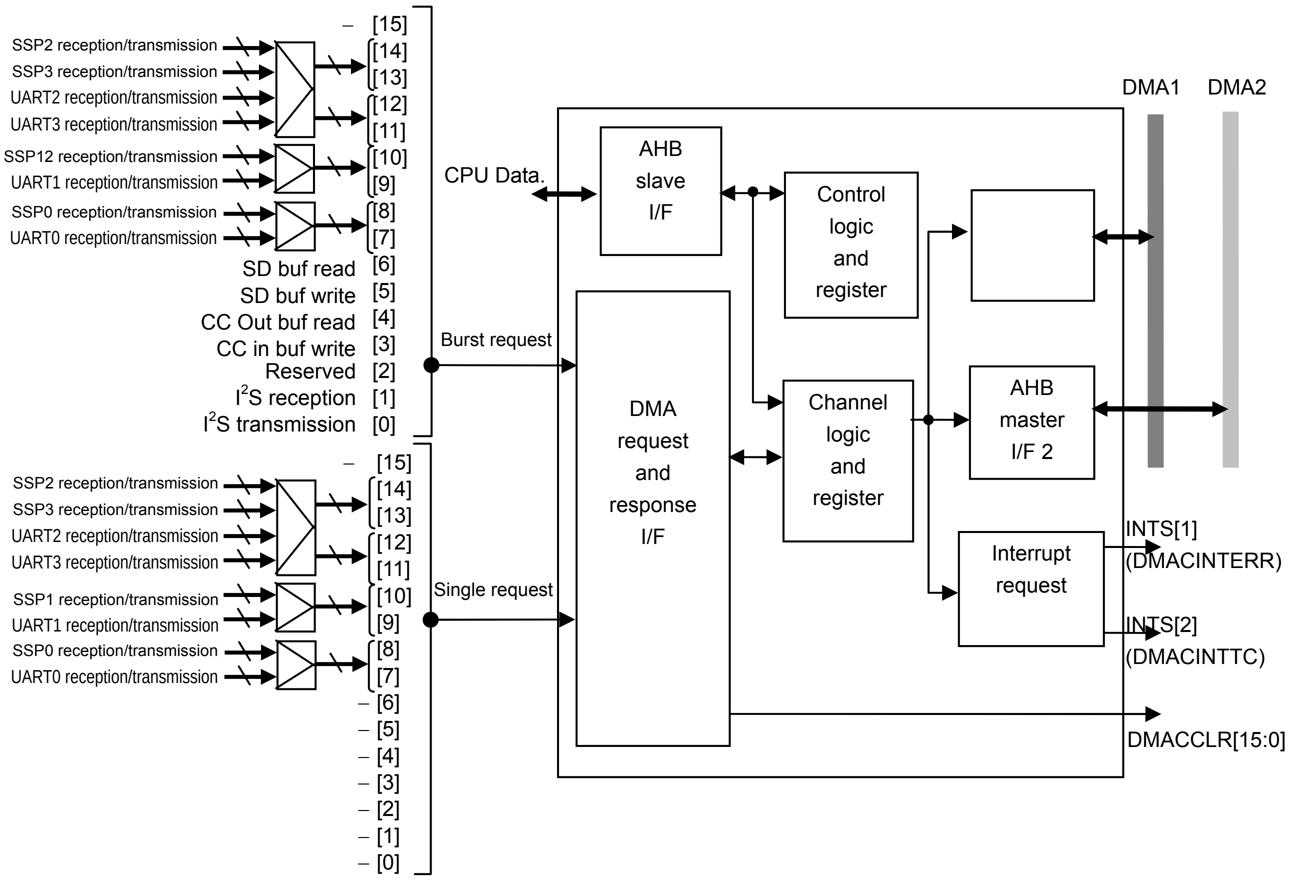

SSP2 reception/transmission
SSP3 reception/transmission
UART2 reception/transmission
UART3 reception/transmission
SSP12 reception/transmission
UART1 reception/transmission
SSP0 reception/transmission
UART0 reception/transmission
SD buf read
SD buf write
CC Out buf read
CC in buf write
Reserved
I2S reception
I2S transmission
– [15]
[14]
[13]
[12]
[11]
[10]
[9]
[8]
[7]
[6]
[5]
[4]
[3]
[2]
[1]
[0]
SSP2 reception/transmission
SSP3 reception/transmission
UART2 reception/transmission
UART3 reception/transmission
SSP1 reception/transmission
UART1 reception/transmission
SSP0 reception/transmission
UART0 reception/transmission
– [15]
[14]
[13]
[12]
[11]
[10]
[9]
[8]
[7]
– [6]
– [5]
– [4]
– [3]
– [2]
– [1]
– [0]
CPU Data.
Burst request
Single request
AHB slave I/F
Control logic and register
DMA request and response I/F
Channel logic and register
AHB master I/F 2
Interrupt request
DMA1
DMA2
INTS[1] (DMACINTERR)
INTS[2] (DMACINTTC)
DMACCLR[15:0]

Table 3.8.2 DMA request number chart

DMA request No.	Corresponding peripheral	
	Burst	Single
0	I²S transmission	–
1	I²S reception	–
2	Reserved	–
3	SDHost CC input buffer write request	–
4	SDHost CC output buffer read request	–
5	SDHost SD buffer write request	–
6	SDHost SD buffer read request	–
7	SSP0/UART0 transmission (*)	SSP0/UART0 transmission (*)
8	SSP0/UART0 reception (*)	SSP0/UART0 reception (*)
9	SSP1/UART1 transmission (*)	SSP1/UART1 transmission (*)
10	SSP1/UART1 reception (*)	SSP1/UART1 reception (*)
11	SSP2/SSP3/UART2/UART3 transmission (*)	SSP2/SSP3/UART2/UART3 transmission (*)
12	SSP2/SSP3/UART2/UART3 reception (*)	SSP2/SSP3/UART2/UART3 reception (*)
13	SSP2/SSP3/UART2/UART3 transmission (*)	SSP2/SSP3/UART2/UART3 transmission (*)
14	SSP2/SSP3/UART2/UART3 reception (*)	SSP2/SSP3/UART2/UART3 reception (*)
15	–	–

*) For more information, see the CG_DMASELR register in Section 3.5.

3.8.2.1 Details of DMA Transfer Types

Table 3.8.3 DMA Transfer Types

<table>
<tr><th></th><th>DMA transfer direction</th><th>Circuit of DMA request output</th><th>Usable DMA Request
Note 3:</th><th colspan="3">Description</th></tr>
<tr><td>1</td><td>Memory → peripheral circuit</td><td>Peripheral circuit</td><td>Burst request</td><td colspan="3">1) Uses burst requests in all.
2) For transfer, set DMA burst to 1 for single request.</td></tr>
<tr><td>2</td><td>Peripheral circuit → memory</td><td>Peripheral circuit</td><td>Burst request / single request
Note 1:</td><td colspan="3">If the amount of data transfer is not an integral multiple of the burst size, both burst and single transfers are used.
Amount of remaining transfer data ≥ Burst size
• Uses burst transfer.
Amount of remaining transfer data < Burst size
• Uses single transfer.</td></tr>
<tr><td>3</td><td>Memory → memory</td><td>DMAC</td><td>None</td><td colspan="3">Start condition:
Enabling the DMAC starts data transfer with no DMAC request required.
Use condition:
Transfer of all transfer data is complete.
The DMAC channel is disabled.
Note 2:</td></tr>
<tr><td rowspan="4">4</td><td rowspan="4">Peripheral circuit → peripheral circuit</td><td rowspan="2">Transfer source peripheral circuit</td><td rowspan="2">Burst request / single request
Note 1:</td><td>Transfer size</td><td>Transfer source side</td><td>Transfer destination side</td></tr>
<tr><td>1) An integral multiple of the burst transfer</td><td>Burst request</td><td rowspan="3">Burst request</td></tr>
<tr><td rowspan="2">Transfer destination peripheral circuit</td><td rowspan="2">Burst request</td><td>2) Single transfer</td><td>Single request</td></tr>
<tr><td>3) Not an integral multiple of the burst transfer</td><td>Burst request / single request</td></tr>
</table>

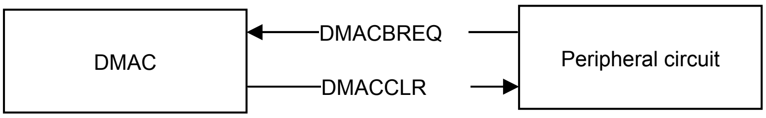
Note 1) Peripheral circuits compatible with single request: UART and SSP

Note 2) Recommendation:
When transferring (large) data using "memory → memory," using a channel with low priority (DMAC6 and 7) is recommended. Other AHB Masters can take bus ownership in the middle of transfer. When using channels other than DMAC6 and 7, taking the bus ownership by other AHB Masters needs to wait until the transfer has been completed.

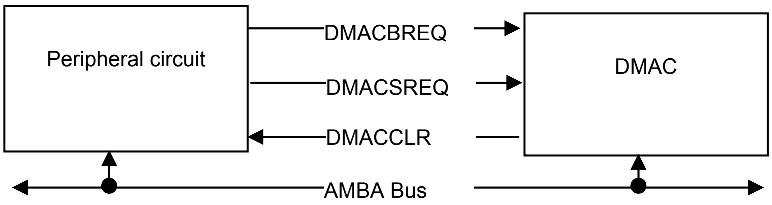
Note 3) Usable DMA request
See the next page.

3.8.2.2 Usable DMA Request

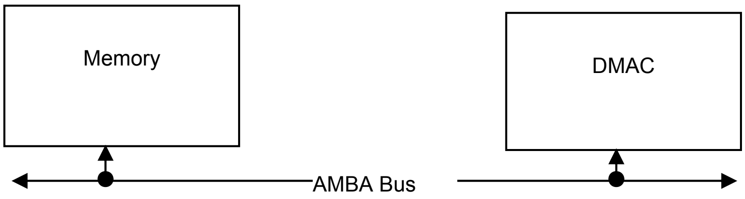
1. Memory → peripheral circuit



2. Peripheral circuit → memory

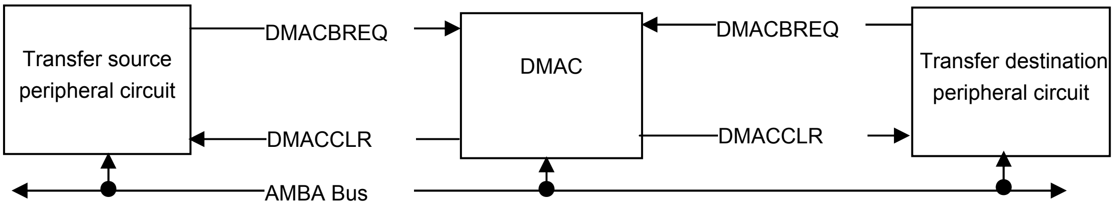


3. Memory → memory

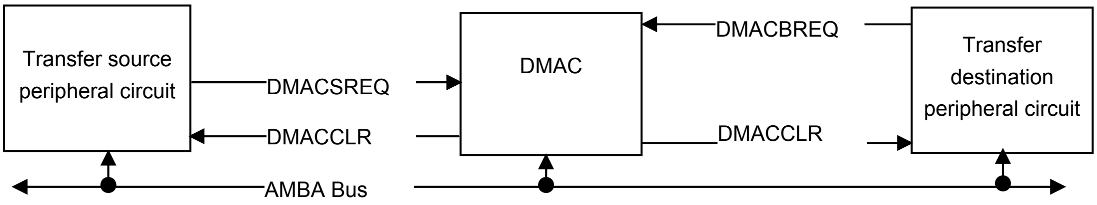


4. Peripheral circuit → peripheral circuit

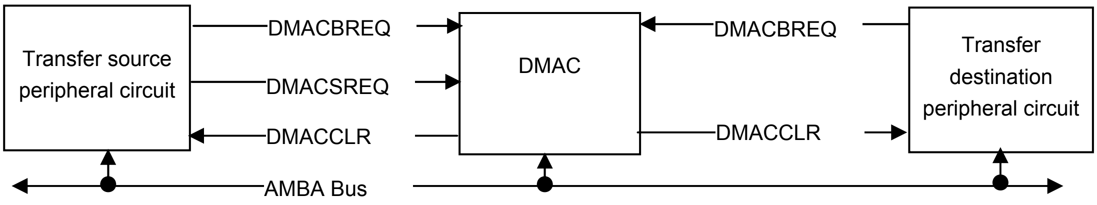
4-1) An integral multiple of the burst transfer


Transfer source peripheral circuit
DMACBREQ
DMACCLR
DMAC
DMACBREQ
DMACCLR
Transfer destination peripheral circuit
AMBA Bus

4-2) Single transfer


Transfer source peripheral circuit
DMACSREQ
DMACCLR
DMAC
DMACBREQ
DMACCLR
Transfer destination peripheral circuit
AMBA Bus

4-3) Not an integral multiple of the burst transfer


Transfer source peripheral circuit
DMACBREQ
DMACSREQ
DMACCLR
DMAC
DMACBREQ
DMACCLR
Transfer destination peripheral circuit
AMBA Bus

The DMA transfer request in the TMPM320C1D includes, as shown below, the 12 (15) DMA transfer requests from peripheral circuits and the 16 software DMA transfer requests initiated with a write to the register of the DMA controller.

Generated by hardware

DMA transfer request from peripheral circuits: 14 requests

Generated by software

Started with a write to the DMACSoftBReq register: 16 requests

These DMA transfer requests are used by assigning them to the eight channels from DMA channels 0 to 7 provided in the DMA controller. From DMA channels 0 to 7, priorities are fixed by hardware, where DMA channel 0 has the highest priority. The DMA controller has two AHB bus masters, and the following is set for each channel:

- Selection of DMA transfer request source
- Selection of AHB bus master
- Selection of transfer type
- Setting of transfer method (AHB protocol)
- Interrupts (transfer and error)

Setting and enabling a DMA channel sets the DMA controller to enter the standby state to wait for DMA transfer requests from peripheral circuits. When a DMA transfer request is asserted from a peripheral circuit, the DMA controller starts up a set AHB bus master so that the AHB bus master reads data from a set transfer source address to make a write to the transfer destination address (For each of the addresses, "fixed" or "increment" can be selected). After data of a set number of beats has been transferred, the DMA controller asserts a request clear signal to the peripheral circuit which output the DMA transfer request. This deasserts the DMA transfer request from the peripheral circuit.

At this time, if the amount of transferred data has not reached the total number of transfers, the DMA channel holds the setting to allow the DMA controller to keep staying in a standby state.

After the total transfer amount is reached after repeating a transfer each time the peripheral circuit asserts a DMA transfer request, the DMA controller asserts a request clear signal to the peripheral circuit and asserts a DMA transfer end interrupt, if enabled, to disable the DMA channel.

* For the address areas accessible by the DMA controller, see "Memory Map" in Section 3.3.

3.8.3 Description of Registers

The following lists the SFRs and their functions:

Table 3.8.4 SFR list

base address = 0x4000_0000

Register Name	Address (base+)	Description
DMACIntStaus	0x0000	DMAC Interrupt Status Register
DMACIntTCStatus	0x0004	DMAC Interrupt Terminal Count Status Register
DMACIntTCClear	0x0008	DMAC Interrupt Terminal Count Clear Register
DMACIntErrorStatus	0x000C	DMAC Interrupt Error Status Register
DMACIntErrClr	0x0010	DMAC Interrupt Error Clear Register
DMACRawIntTCStatus	0x0014	DMAC Raw Interrupt Terminal Count Status Register
DMACRawIntErrorStatus	0x018	DMAC Raw Error Interrupt Status Register
DMACEnbldChns	0x01C	DMAC Enabled Channel Register
DMACSoftBReq	0x020	DMAC Software Burst Request Register
DMACSoftSReq	0x024	DMAC Software Single Request Register
–	0x028	Reserved
–	0x02C	Reserved
DMACConfiguration	0x030	DMAC Configuration Register
–	0x034	Reserved
DMACC0SrcAddr	0x100	DMAC Channel0 Source Address Register
DMACC0DestAddr	0x104	DMAC Channel0 Destination Address Register
DMACC0LLI	0x108	DMAC Channel0 Linked List Item Register
DMACC0Control	0x10C	DMAC Channel0 Control Register
DMACC0Configuration	0x110	DMAC Channel0 Configuration Register
DMACC1SrcAddr	0x120	DMAC Channel1 Source Address Register
DMACC1DestAddr	0x124	DMAC Channel1 Destination Address Register
DMACC1LLI	0x128	DMAC Channel1 Linked List Item Register
DMACC1Control	0x12C	DMAC Channel1 Control Register
DMACC1Configuration	0x130	DMAC Channel1 Configuration Register
DMACC2SrcAddr	0x140	DMAC Channel2 Source Address Register
DMACC2DestAddr	0x144	DMAC Channel2 Destination Address Register
DMACC2LLI	0x148	DMAC Channel2 Linked List Item Register
DMACC2Control	0x14C	DMAC Channel2 Control Register
DMACC2Configuration	0x150	DMAC Channel2 Configuration Register
DMACC3SrcAddr	0x160	DMAC Channel3 Source Address Register
DMACC3DestAddr	0x164	DMAC Channel3 Destination Address Register
DMACC3LLI	0x168	DMAC Channel3 Linked List Item Register
DMACC3Control	0x16C	DMAC Channel3 Control Register
DMACC3Configuration	0x170	DMAC Channel3 Configuration Register
DMACC4SrcAddr	0x180	DMAC Channel4 Source Address Register
DMACC4DestAddr	0x184	DMAC Channel4 Destination Address Register
DMACC4LLI	0x188	DMAC Channel4 Linked List Item Register
DMACC4Control	0x18C	DMAC Channel4 Control Register
DMACC4Configuration	0x190	DMAC Channel4 Configuration Register
DMACC5SrcAddr	0x1A0	DMAC Channel5 Source Address Register
DMACC5DestAddr	0x1A4	DMAC Channel5 Destination Address Register
DMACC5LLI	0x1A8	DMAC Channel5 Linked List Item Register
DMACC5Control	0x1AC	DMAC Channel5 Control Register
DMACC5Configuration	0x1B0	DMAC Channel5 Configuration Register

Note) Access the registers by using word reads and word writes.

Register Name	Address (base+)	Description
DMACC6SrcAddr	0x1C0	DMAC Channel6 Source Address Register
DMACC6DestAddr	0x1C4	DMAC Channel6 Destination Address Register
DMACC6LLI	0x1C8	DMAC Channel6 Linked List Item Register
DMACC6Control	0x1CC	DMAC Channel6 Control Register
DMACC6Configuration	0x1D0	DMAC Channel6 Configuration Register
DMACC7SrcAddr	0x1E0	DMAC Channel7 Source Address Register
DMACC7DestAddr	0x1E4	DMAC Channel7 Destination Address Register
DMACC7LLI	0x1E8	DMAC Channel7 Linked List Item Register
DMACC7Control	0x1EC	DMAC Channel7 Control Register
DMACC7Configuration	0x1F0	DMAC Channel7 Configuration Register
−	0xFE0	Reserved
−	0xFE4	Reserved
−	0xFE8	Reserved
−	0xFEC	Reserved
−	0xFF0	Reserved
−	0xFF4	Reserved
−	0xFF8	Reserved
−	0xFFC	Reserved
−	0x500	Reserved
−	0x504	Reserved
−	0x508	Reserved
−	0x50C	Reserved

Note) Access the registers by using word reads and word writes.

Table 3.8.5 SFR

base address = 0x4000_5300

Register Name	Address	Description
CG_DMASELR	0x4000_5300	DMA request Select control Register

Note) Access the registers by using word reads and word writes.

1. DMACIntStatus (DMAC Interrupt Status Register)

Address = (0x4000_0000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	IntStatus7	RO	0y0	Status of DMAC channel 7 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[6]	IntStatus6	RO	0y0	Status of DMAC channel 6 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[5]	IntStatus5	RO	0y0	Status of DMAC channel 5 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[4]	IntStatus4	RO	0y0	Status of DMAC channel 4 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[3]	IntStatus3	RO	0y0	Status of DMAC channel 3 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[2]	IntStatus2	RO	0y0	Status of DMAC channel 2 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[1]	IntStatus1	RO	0y0	Status of DMAC channel 1 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[0]	IntStatus0	RO	0y0	Status of DMAC channel 0 interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested

[Explanation]

a. <IntStatus[7:0]>

Status of the DMAC interrupt generation after passing through the transfer end interrupt enable register and error interrupt enable register. An interrupt is requested when there is a transfer error or when the counter completes counting.

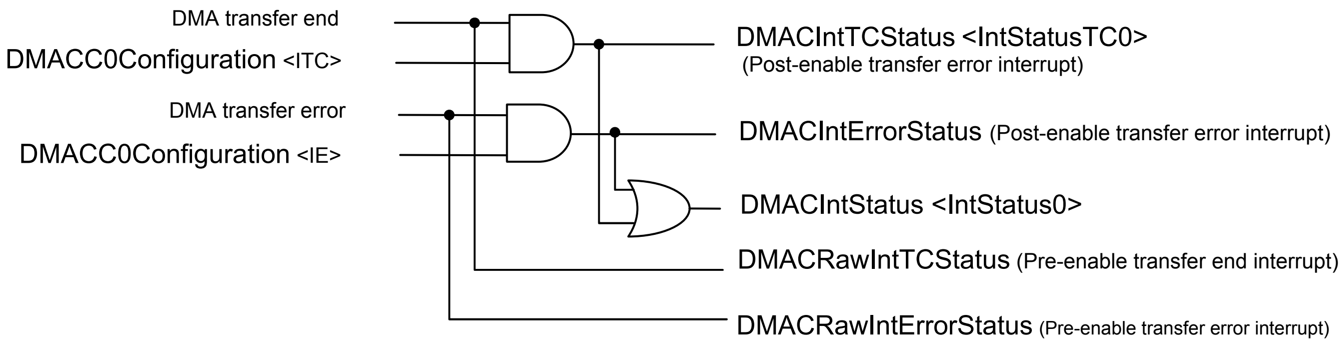

DMA transfer end
DMACC0Configuration <ITC>
DMA transfer error
DMACC0Configuration <IE>
DMACIntTCStatus <IntStatusTC0>
(Post-enable transfer error interrupt)
DMACIntErrorStatus (Post-enable transfer error interrupt)
DMACIntStatus <IntStatus0>
DMACRawIntTCStatus (Pre-enable transfer end interrupt)
DMACRawIntErrorStatus (Pre-enable transfer error interrupt)

Figure 3.8.1 Interrupt-related block diagram

2. DMACIntTCStatus (DMAC Interrupt Terminal Count Status Register)

Address = (0x4000_0000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	IntStatusTC7	RO	0y0	Status of DMAC channel 7 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[6]	IntStatusTC6	RO	0y0	Status of DMAC channel 6 transfer end interrupt generation. 1: Interrupt requested 0: Interrupt not requested
[5]	IntStatusTC5	RO	0y0	Status of DMAC channel 5 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[4]	IntStatusTC4	RO	0y0	Status of DMAC channel 4 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[3]	IntStatusTC3	RO	0y0	Status of DMAC channel 3 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[2]	IntStatusTC2	RO	0y0	Status of DMAC channel 2 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[1]	IntStatusTC1	RO	0y0	Status of DMAC channel 1 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[0]	IntStatusTC0	RO	0y0	Status of DMAC channel 0 transfer end interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested

[Explanation]

a. <IntTStatusTC[7:0]>

The status of post-enable transfer end interrupt generation.

3. DMACIntTCClear (DMAC Interrupt Terminal Count Clear Register)

Address = (0x4000_0000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	IntTCClear7	WO	0y0	Clear DMAC channel 7 transfer end interrupt 0y1: Clear 0y0: Do nothing
[6]	IntTCClear6	WO	0y0	Clear DMAC channel 6 transfer end interrupt 0y1: Clear 0y0: Do nothing
[5]	IntTCClear5	WO	0y0	Clear DMAC channel 5 transfer end interrupt 0y1: Clear 0y0: Do nothing
[4]	IntTCClear4	WO	0y0	Clear DMAC channel 4 transfer end interrupt 0y1: Clear 0y0: Do nothing
[3]	IntTCClear3	WO	0y0	Clear DMAC channel 3 transfer end interrupt 0y1: Clear 0y0: Do nothing
[2]	IntTCClear2	WO	0y0	Clear DMAC channel 2 transfer end interrupt 0y1: Clear 0y0: Do nothing
[1]	IntTCClear1	WO	0y0	Clear DMAC channel 1 transfer end interrupt 0y1: Clear 0y0: Do nothing
[0]	IntTCClear0	WO	0y0	Clear DMAC channel 0 transfer end interrupt 0y1: Clear 0y0: Do nothing

[Explanation]

a. <IntTCClearCH[7:0]>

The DMACINTTCS register bit of the channel that corresponds to the bit to which "1" was written will clear the interrupt.

4. DMACIntErrorStatus (DMAC Interrupt Error Status Register)

Address = (0x4000_0000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	IntErrStatus7	RO	0y0	Status of DMAC channel 7 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[6]	IntErrStatus6	RO	0y0	Status of DMAC channel 6 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[5]	IntErrStatus5	RO	0y0	Status of DMAC channel 5 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[4]	IntErrStatus4	RO	0y0	Status of DMAC channel 4 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[3]	IntErrStatus3	RO	0y0	Status of DMAC channel 3 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[2]	IntErrStatus2	RO	0y0	Status of DMAC channel 2 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[1]	IntErrStatus1	RO	0y0	Status of DMAC channel 1 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[0]	IntErrStatus0	RO	0y0	Status of DMAC channel 0 error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested

[Explanation]

a. <IntErrStatus[7:0]>
Error interrupt status after enabled

5. DMACIntErrClr (DMAC Interrupt Error Clear Register)

Address = (0x4000_0000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	IntErrClr7	WO	0y0	Clear DMAC channel 7 error interrupt 0y1: Clear 0y0: Do nothing
[6]	IntErrClr6	WO	0y0	Clear DMAC channel 6 error interrupt 0y1: Clear 0y0: Do nothing
[5]	IntErrClr5	WO	0y0	Clear DMAC channel 5 error interrupt 0y1: Clear 0y0: Do nothing
[4]	IntErrClr4	WO	0y0	Clear DMAC channel 4 error interrupt 0y1: Clear 0y0: Do nothing
[3]	IntErrClr3	WO	0y0	Clear DMAC channel 3 error interrupt 0y1: Clear 0y0: Do nothing
[2]	IntErrClr2	WO	0y0	Clear DMAC channel 2 error interrupt 0y1: Clear 0y0: Do nothing
[1]	IntErrClr1	WO	0y0	Clear DMAC channel 1 error interrupt 0y1: Clear 0y0: Do nothing
[0]	IntErrClr0	WO	0y0	Clear DMAC channel 0 error interrupt 0y1: Clear 0y0: Do nothing

[Explanation]

a. <IntErrClr[7:0]>
"1": Clears an error interrupt request.

6. DMACRawIntTCStatus (DMAC Raw Interrupt Terminal Count Status Register)

Address = (0x4000_0000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	RawIntTCS7	RO	0y0	Status of DMAC channel 7 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested
[6]	RawIntTCS6	RO	0y0	Status of DMAC channel 6 pre-enable transfer end interrupt generation 1: Interrupt requested 0: Interrupt not requested
[5]	RawIntTCS5	RO	0y0	Status of DMAC channel 5 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested
[4]	RawIntTCS4	RO	0y0	Status of DMAC channel 4 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested
[3]	RawIntTCS3	RO	0y0	Status of DMAC channel 3 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested
[2]	RawIntTCS2	RO	0y0	Status of DMAC channel 2 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested
[1]	RawIntTCS1	RO	0y0	Status of DMAC channel 1 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested
[0]	RawIntTCS0	RO	0y0	Status of DMAC channel 0 pre-enable transfer end interrupt generation 0y1: Interrupt requested 0y0: Interrupt not requested

7. DMACRawIntErrorStatus (DMAC Raw Error Interrupt Status Register)

Address = (0x4000_0000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	RawIntErrS7	RO	0y0	Status of DMAC channel 7 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[6]	RawIntErrS6	RO	0y0	Status of DMAC channel 6 pre-enable error interrupt generation. 1: Interrupt requested 0: Interrupt not requested
[5]	RawIntErrS5	RO	0y0	Status of DMAC channel 5 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[4]	RawIntErrS4	RO	0y0	Status of DMAC channel 4 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[3]	RawIntErrS3	RO	0y0	Status of DMAC channel 3 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[2]	RawIntErrS2	RO	0y0	Status of DMAC channel 2 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[1]	RawIntErrS1	RO	0y0	Status of DMAC channel 1 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested
[0]	RawIntErrS0	RO	0y0	Status of DMAC channel 0 pre-enable error interrupt generation. 0y1: Interrupt requested 0y0: Interrupt not requested

8. DMACEnbldChns (DMAC Enabled Channel Register)

Address = (0x4000_0000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	EnabledCH7	RO	0y0	DMA channel 7 enable status 0y1: Enable 0y0: Disable
[6]	EnabledCH6	RO	0y0	DMA channel 6 enable status 0y1: Enable 0y0: Disable
[5]	EnabledCH5	RO	0y0	DMA channel 5 enable status 0y1: Enable 0y0: Disable
[4]	EnabledCH4	RO	0y0	DMA channel 4 enable status 0y1: Enable 0y0: Disable
[3]	EnabledCH3	RO	0y0	DMA channel 3 enable status 0y1: Enable 0y0: Disable
[2]	EnabledCH2	RO	0y0	DMA channel 2 enable status 0y1: Enable 0y0: Disable
[1]	EnabledCH1	RO	0y0	DMA channel 1 enable status 0y1: Enable 0y0: Disable
[0]	EnabledCH0	RO	0y0	DMA channel 0 enable status 0y1: Enable 0y0: Disable

[Explanation]

a. <EnabledCH[7:0]>

"0": The bits of the appropriate channel are cleared when DMA transfer is complete.

"1": The appropriate channel DMA is enabled.

9. DMACSoftBReq (DMAC Software Burst Request Register)

Address = (0x4000_0000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:15]	–	–	Undefined	Read undefined. Write as zero.
[14]	SoftBReq12	R/W	0y0	DMA burst request by software at SSP2/SSP3/UART2/UART3 reception. 0y1: DMA burst requested 0y0: Disabled (WR)
[13]	SoftBReq11	R/W	0y0	DMA burst request by software at SSP2/SSP3/UART2/UART3 transmission. 0y1: DMA burst requested 0y0: Disabled (WR)
[12]	SoftBReq12	R/W	0y0	DMA burst request by software at SSP2/SSP3/UART2/UART3 reception. 0y1: DMA burst requested 0y0: Disabled (WR)
[11]	SoftBReq11	R/W	0y0	DMA burst request by software at SSP2/SSP3/UART2/UART3 transmission. 0y1: DMA burst requested 0y0: Disabled (WR)
[10]	SoftBReq10	R/W	0y0	DMA burst request by software at SSP1/UART1 reception. 0y1: DMA burst requested 0y0: Disabled (WR)
[9]	SoftBReq9	R/W	0y0	DMA burst request by software at SSP1/UART1 transmission. 0y1: DMA burst requested 0y0: Disabled (WR)
[8]	SoftBReq8	R/W	0y0	DMA burst request by software at SSP0/UART0 reception. 0y1: DMA burst requested 0y0: Disabled (WR)
[7]	SoftBReq7	R/W	0y0	DMA burst request by software at SSP0/UART0 transmission. 0y1: DMA burst requested 0y0: Disabled (WR)
[6]	SoftBReq6	R/W	0y0	DMA burst request by software at SDHost SD buffer read. 0y1: DMA burst requested 0y0: Disabled (WR)
[5]	SoftBReq5	R/W	0y0	DMA burst request by software at SDHost SD buffer write. 0y1: DMA burst requested 0y0: Disabled (WR)
[4]	SoftBReq4	R/W	0y0	DMA burst request by software at SDHost CC output buffer read. 0y1: DMA burst requested 0y0: Disabled (WR)
[3]	SoftBReq3	R/W	0y0	DMA burst request by software at SDHost CC input buffer write. 0y1: DMA burst requested 0y0: Disabled (WR)

Bit	Bit Symbol	Type	Reset Value	Description
[3]	SoftBReq3	R/W	0y0	DMA burst request by software at SDHost CC input buffer write. 0y1: DMA burst requested 0y0: Disabled (WR)
[2]	SoftBReq2	R/W	0y0	Read undefined. Write as zero.
[1]	SoftBReq1	R/W	0y0	I^{2}S reception DMA burst request by software. 0y1: DMA burst requested 0y0: Disabled (WR)
[0]	SoftBReq0	R/W	0y0	I^{2}S transmission DMA burst request by software. 0y1: DMA burst requested 0y0: Disabled (WR)

[Explanation]

a. <SoftBReq[14:0]>

Sets a DMA burst transfer request by software. When the DMA burst transfer by software is complete, the appropriate bits in SoftBReq[14:0] are cleared.

Note) Do not execute DMA requests by software and hardware peripheral at the same time.

10. DMACSoftSReq (DMAC Software Single Request Register)

Address = (0x4000_0000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:15]	–	–	Undefined	Read undefined. Write as zero.
[14]	SoftSReq12	R/W	0y0	DMA single request by software at SSP2/SSP3/UART2/UART3 reception. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[13]	SoftSReq11	R/W	0y0	DMA single request by software at SSP2/SSP3/UART2/UART3 transmission. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[12]	SoftSReq12	R/W	0y0	DMA single request by software at SSP2/SSP3/UART2/UART3 reception. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[11]	SoftSReq11	R/W	0y0	DMA single request by software at SSP2/SSP3/UART2/UART3 transmission. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[10]	SoftSReq10	R/W	0y0	DMA single request by software at SSP1/UART1 reception. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[9]	SoftSReq9	R/W	0y0	DMA single request by software at SSP1/UART1 transmission. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[8]	SoftSReq8	R/W	0y0	DMA single request by software at SSP0/UART0 reception. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[7]	SoftSReq7	R/W	0y0	DMA single request by software at SSP0/UART0 transmission. 0y1: Generate a DMA single request 0y0: Disabled (WR)
[6:0]	Reserved	–	Undefined	Read undefined. Write as zero.

[Explanation]

a. <SoftSReq[14:7]>

Sets a DMA single transfer request by software. When the DMA single transfer by software is complete, the appropriate bits in SoftSReq[14:7] are cleared.

Note) Do not execute a DMA request by software when a DMA request by hardware peripheral is generated.

11. DMACConfiguration (DMAC Configuration Register)

Address = (0x4000_0000) + 0x0030

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	M2	R/W	0y0	DMA2 endian types: 0: Little endian 1: Reserved
[1]	M1	R/W	0y0	DMA1 endian types: 0: Little endian 1: Reserved
[0]	E	R/W	0y0	DMA circuit control: 0 : Stop 1 : Operate

[Explanation]

a. <M2>

DMA2 endian configuration

b. <M1>

DMA2 endian configuration

c. <E>

The registers for the DMA circuit cannot be written or read unless the DMA circuit operates. When operating the DMA, always keep the DMA circuit operating.

12. DMACC0SrcAddr (DMAC Channel0 Source Address Register)

Address = (0x4000_0000) + 0x0100

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	SrcAddr	R/W	0x00000000	Sets a DMA transfer source address

[Explanation]

a. <SrcAddr>

Because enabling channels updates the data written in the registers, set DMACCxSrcAddr before enabling the channels.

When the DMA is operating, the value in the DMACCxSrcAddr register sequentially changes, so the read values are not fixed.
Do not update DMACC0SrcAddr during transfer. To change the value, be sure to set the DMACCxConfiguration register to disable the channel before change.

- DMACCxSrcAddr (DMAC Channel x Source Address Register) (x = 0~7)

Refer to the description on DMACC0SrcAddr because the structures and explanations on the above registers are the same as DMACC0SrcAddr. Also, refer to Table 3.8.4 SFR list for register names and addresses.

13. DMACC0DestAddr (DMAC Channel0 Destination Address Register)

Address = (0x4000_0000) + 0x0104

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	DestAddr	R/W	0x00000000	Sets a DMA transfer destination address

[Explanation]

a. <DestAddr>

Do not update DMACC0DestAddr during transfer. To change the value, be sure to set the DMACCxConfiguration register to disable the channel before change.

- DMACCxDestAddr (DMAC Channel x Destination Address Register) (x = 0~7)

  Refer to the description on DMACC0DestAddr because the structures and explanations on the above registers are the same as DMACC0DestAddr. Also, refer to Table 3.8.4 SFR list for register names and addresses.

14. DMACC0LLI (DMAC Channel0 Linked List Item Register)

Address = (0x4000_0000) + 0x0108

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	LLI	R/W	0x00000000	Sets the first address of the next transfer information.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	LM	R/W	0y0	Selects LLI storage destination AHB Master: 0y0 : DMA1 0y1 : DMA2

[Explanation]

a. <LLI>

Set a value smaller than 0xFFFF_FFF0 for <LLI>.

When <LLI> = 0, currently, LLI is the last chain. After DMA transfer finishes, the DMA channel is disabled.

* For detailed operation, see "Special Functions" in Section 3.8.4.

b. <LM>

Selects the storage destination AHB Master for a next LLI load.

- DMACCxLLI (DMAC Channel x Linked List Item Register) (x = 0~7)

  Refer to the description on DMACC0LLI because the structures and explanations on the above registers are the same as DMACC0LLI. Also, refer to Table 3.8.4 SFR list for register names and addresses.

15. DMACC0Control (DMAC Channel0 Control Register)

Address = (0x4000_0000) + 0x010C

Bit	Bit Symbol	Type	Reset Value	Description
[31]	I	R/W	0y0	Register for enabling a transfer end interrupt when the scatter/gather function is used 0y0 : Disable 0y1 : Enable
[30]	Prot[3]	R/W	0y0	Read undefined. Write as zero.
[29]	Prot[2]	R/W	0y0	Read undefined. Write as zero.
[28]	Prot[1]	R/W	0y0	Read undefined. Write as zero.
[27]	DI	R/W	0y0	Increment the transfer destination address 0y0: Do not increment 0y1: Increment
[26]	SI	R/W	0y0	Increment the transfer source address 0y0: Do not increment 0y1: Increment
[25]	D	R/W	0y0	Transfer destination AHB Master 0y0 : DMA1 0y1 : DMA2
[24]	S	R/W	0y0	Transfer source AHB Master 0y0 : DMA1 0y1 : DMA2
[23:21]	Dwidth[2:0]	R/W	0y000	Transfer destination bit width 0y000: Byte (8 bits) 0y001: Half-word (16 bits) 0y010: Word (32 bits) other: Reserved
[20:18]	Swidth[2:0]	R/W	0y000	Transfer source bit width 0y000: Byte (8 bits) 0y001: Half-word (16 bits) 0y010: Word (32 bits) other: Reserved
[17:15]	DBSize[2:0]	R/W	0y000	Transfer destination burst size: 0y000: 1 beat 0y001: 4 beats 0y010: 8 beats 0y011: 16 beats 0y100: 32 beats 0y101: 64 beats 0y110: 128 beats 0y111: 256 beats
[14:12]	SBSize[2:0]	R/W	0y000	Transfer source burst size: 0y000: 1 beat 0y001: 4 beats 0y010: 8 beats 0y011: 16 beats 0y100: 32 beats 0y101: 64 beats 0y110: 128 beats 0y111: 256 beats
[11:0]	TransferSize	R/W	0x000	Set the total number of transfers

[Explanation] The same explanation is applied for the other channels too.

a. <I>

Register for enabling a transfer end interrupt when the scatter/gather function is used

b. <DI>

Increments the address of a transfer destination.

Depends on the bit width of the transfer source. Increments the address, each depending on Dwidth as follows:

8-bit	: 1 byte
16-bit	: 2 bytes
32-bit	: 4 bytes

c. <SI>

Increments the address of a transfer destination.

Depends on the bit width of the transfer source. Increments the address, each depending on Swidth as follows:

8-bit	: 1 byte
16-bit	: 2 bytes
32-bit	: 4 bytes

d. <D>

Transfer destination AHB Master

e. <S>

Transfer source AHB Master

f. <Swidth[2:0]>

Set a transfer destination bit width so that the transfer size becomes an integral multiple of the transfer destination bit width.

g. <DBSize[2:0]>

Note) The burst size to be set with DBsize has nothing to do with the HBURST for the AHB bus.

h. <SBSize[2:0]>

Note) The burst size to be set with SBsize has nothing to do with the HBURST for the AHB bus.

i. <TransferSize>

Set the total number of transfers when the DMAC is used as the flow controller.

This value decrements to "0" as DMAC transfer is executed. The read operation reads the number of transfers that have not been executed yet.

The total number of transfers is used as the unit for the transfer source bit width.

ex: When Swidth=8bit, the number of transfers is expressed in the units of byte.

ex: When Swidth=16bit, the number of transfers is expressed in the units of half word.

ex: When Swidth=32bit, the number of transfers is expressed in the units of word.

Note) If the transfer source bit width is smaller than the transfer destination bit width, care must be taken when setting the total number of transfers.
Set the number so that the following expression is satisfied:

Transfer source bit width × total number of transfers = Transfer destination bit width × N
N: Integer number

- DMACCxControl (DMAC Channel x Control Register) (x = 0~7)

  Refer to the description on DMACC0Control because the structures and explanations on the above registers are the same as DMACC0Control. Also, refer to Table 3.8.4 SFR list for register names and addresses.

16. DMACC0Configuration (DMAC Channel0 Configuration Register)

Address = (0x4000_0000) + 0x0110

<table>
<tr><th>Bit</th><th>Bit Symbol</th><th>Type</th><th>Reset Value</th><th>Description</th></tr>
<tr><td>[31:19]</td><td>–</td><td>–</td><td>Undefined</td><td>Read undefined. Write as zero.</td></tr>
<tr><td>[18]</td><td>Halt</td><td>R/W</td><td>0y0</td><td>0y0: Accept a DMA request
0y1: Ignore a DMA request</td></tr>
<tr><td>[17]</td><td>Active</td><td>RO</td><td>0y0</td><td>0y0: No data exists in the FIFO
0y1: Data exists in the FIFO</td></tr>
<tr><td>[16]</td><td>Lock</td><td>R/W</td><td>0y0</td><td>0y0: Disable locked transfer
0y1: Enable locked transfer</td></tr>
<tr><td>[15]</td><td>ITC</td><td>R/W</td><td>0y0</td><td>Transfer end interrupt enable register
0y0: Disable interrupt
0y1: Enable interrupt</td></tr>
<tr><td>[14]</td><td>IE</td><td>R/W</td><td>0y0</td><td>Error interrupt enable register
0y0: Disable interrupt
0y1: Enable interrupt</td></tr>
<tr><td>[13:11]</td><td>FlowCntrl</td><td>R/W</td><td>0y000</td><td><table>
<tr><th>FlowCntrl setting value</th><th>Transfer method</th></tr>
<tr><td>0y000</td><td>Memory to Memory</td></tr>
<tr><td>0y001</td><td>Memory to Peripheral</td></tr>
<tr><td>0y010</td><td>Peripheral to Memory</td></tr>
<tr><td>0y011</td><td>Peripheral to Peripheral</td></tr>
</table>
0y100~0y111: Reserved</td></tr>
<tr><td>[10]</td><td>–</td><td>–</td><td>Undefined</td><td>Read undefined. Write as zero.</td></tr>
<tr><td>[9:6]</td><td>DestPeripheral</td><td>R/W</td><td>0y000</td><td>Transfer destination peripheral (Note 1)
0y000~0y1111</td></tr>
<tr><td>[5]</td><td>–</td><td>–</td><td>Undefined</td><td>Read undefined. Write as zero.</td></tr>
<tr><td>[4:1]</td><td>SrcPeripheral</td><td>R/W</td><td>0y000</td><td>Transfer source peripheral (Note 1)
0y000~0y1111</td></tr>
<tr><td>[0]</td><td>E</td><td>R/W</td><td>0y0</td><td>Channel enable
0y0 : Disable
0y1 : Enable</td></tr>
</table>

Note) Refer to DMA request number chart.

[Explanation]

a. <Halt>

Halts DMA.

b. <Active>

Indicates whether data is present in the channel FIFO.

c. <Lock>

Sets a locked transfer (Non-divided transfer). When locked transfer is enabled, as many burst transfers as specified are consecutively executed without releasing the bus. For detailed operation, see Section 3.8.5.7.

d. <ITC>

Transfer end interrupt enable

e. <IE>

Error interrupt enable

f. <FlowCntrl>

Sets a transfer method.

0y000: Memory to Memory

0y001: Memory to Peripheral

0y010: Peripheral to Memory

0y011: Peripheral to Peripheral

0y100~0y111: Reserved

Note) When "memory to memory" is selected, hardware start for DMA startup is not supported. Writing to <E>= 1 starts transfer.

g. <DestPeripheral>

The DMA request peripheral number is expressed by binary.

When a memory is the transfer destination, this setting is ignored.

h. <SrcPeripheral>

The DMA request peripheral number is expressed by binary.

When a memory is the transfer source, this setting is ignored.

i. <E>

This bit can be used to enable/disable the channels. Disabling channels during transfer loses the data in the FIFO. Initialize all the channels before restart.

To pause the transfer, stop the DMA request by using the <HALT> bit, and poll the data until the <Active> bit becomes "0" and then disable the channel with the <E> bit.

- DMACCxConfiguration (DMAC Channel x Configuration Register) (x = 0~7)

  Refer to the description on DMACC0Configuration because the structures and explanations on the above registers are the same as DMACC0Configuration. Also, refer to Table 3.8.4 SFR list for register names and addresses.

- Flow for setting the DMAC

Example of setting the transfer from memory to the FIFO in UART0, using DMAC ch1

Total transfer data amount = 8 words, Swidth = Word basis, total transfer count = 8 transfers

```
DMACConfiguration     ←    0x00000001    ; Set DMAC Active
DMACC1SrcAddr         ←    0xf8004000    ; Source address (DMAC ch1)
DMACC1DestAddr        ←    UART0DR       ; Destination address
DMACC1Control         ←    0x04492008    ; Swidth = 1word, Dswidth=1word
                                         ; DBSize= 4 burst, SBSize= 4 burst (Note)
                                         ; TransferSize = 8 times
```

```
                DMACC1Configuration     ←    0x000009c1     ; channel 1 Enable,
                                                            ; Memory to PERIPHERAL (UART0)
                …                                           ; Set and prepare UART0
                UARTDMACR               ←    0x00000002     ; UART DMA Ready, request a DMA
                                                            transfer
finish_DMA                                                  ; label
                DMACC1Configulation     →    r0             ; read DMACC1Configulation data to r0
                CMP    r0, #0x0                             ; check the End of Tx DMAC
                BNE     finish_DMA                          ; r0 ≠ 0x0, jump to finish_DMA
                UART0DMACR              ←    0x00000000     ; Clear a UART0 DMA request
```

Note) Set the burst size according to the FIFO size of the peripheral.

17. CG_DMASELR (DMA request Select control Register)

Address = (0x4000_0000) + 0x0110

<table>
<tr><th>Bit</th><th>Bit
Symbol</th><th>Type</th><th>Reset
Value</th><th>Description</th></tr>
<tr><td>[31:20]</td><td>–</td><td>–</td><td>Undefined</td><td>Read undefined. Write as zero.</td></tr>
<tr><td>[19:16]</td><td>DMASEL</td><td>R/W</td><td>0y0000</td><td>Select 2 IPs from 4 IPs (SSP2, SSP3, UART2, and UART3) for the DMA request peripheral numbers 11 to 14
<table>
<tr><th>Setting value</th><th>Factor 11/12</th><th>Factor 13/14</th></tr>
<tr><td>0y0000
0y0001</td><td>SSP2 transmission/reception</td><td>SSP3 transmission/reception</td></tr>
<tr><td>0y0010</td><td>SSP2 transmission/reception</td><td rowspan="2">UART2 transmission/reception</td></tr>
<tr><td>0y0011</td><td>SSP3 transmission/reception</td></tr>
<tr><td>0y0100</td><td>SSP2 transmission/reception</td><td rowspan="2">UART3 transmission/reception</td></tr>
<tr><td>0y0101</td><td>SSP3 transmission/reception</td></tr>
<tr><td>0y0110
0y0111</td><td>UART2 transmission/reception</td><td>UART3 transmission/reception</td></tr>
<tr><td>0y1000
0y1001</td><td>SSP3 transmission/reception</td><td>SSP2 transmission/reception</td></tr>
<tr><td>0y1010</td><td rowspan="2">UART2 transmission/reception</td><td>SSP2 transmission/reception</td></tr>
<tr><td>0y1011</td><td>SSP3 transmission/reception</td></tr>
<tr><td>0y1100</td><td rowspan="2">UART3 transmission/reception</td><td>SSP2 transmission/reception</td></tr>
<tr><td>0y1101</td><td>SSP3 transmission/reception</td></tr>
<tr><td>0y1110
0y1111</td><td>UART3 transmission/reception</td><td>UART2 transmission/reception</td></tr>
</table>
</td></tr>
<tr><td>[15:2]</td><td>–</td><td>–</td><td>Undefined</td><td>Read undefined. Write as zero.</td></tr>
<tr><td>[1]</td><td>SP1U1DMA</td><td>R/W</td><td>0y0</td><td>Select either SSP CH1 or UART CH1 for the DMA request peripheral numbers 9 and 10.
0y0: SSP1 transmission/reception DMA request
0y1: UART1 transmission/reception DMA request</td></tr>
<tr><td>[0]</td><td>SP0U0DMA</td><td>R/W</td><td>0y0</td><td>Select either SSP CH0 or UART CH0 for the DMA request peripheral numbers 7 and 8.
0y0: SSP0 transmission/reception DMA request
0y1: UART0 transmission/reception DMA request</td></tr>
</table>

Note) Refer to DMA request number chart.

[Explanation]

a. <SP0U0DMA>

Selects either SSP0 or UART0 transmission/reception DMA request for the DMA request peripheral numbers 7 and 8. When a memory is the transfer destination, this setting is ignored.

b. <SP1U1DMA>

Select either SSP1 or UART1 transmission/reception DMA request for the DMA request peripheral numbers 9 and 10. When a memory is the transfer destination, this setting is ignored.

c. <DMASEL>

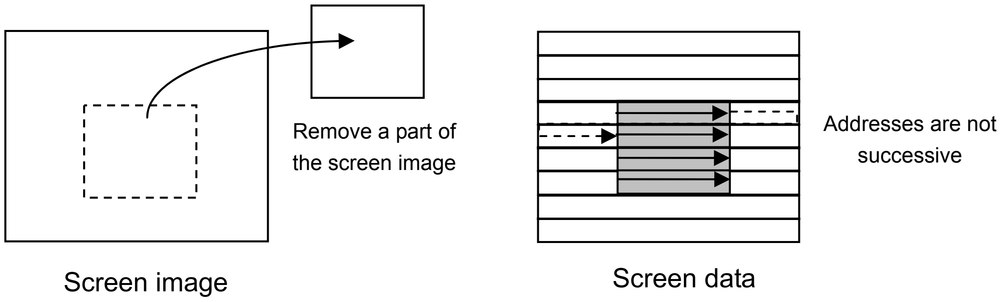
Selects 2 IPs from 4 IPs (SSP2, SSP3, UART2, and UART3) for the DMA request peripheral numbers 11/12 and 13/14, and maps the transmission/reception DMA request for each.

When a memory is the transfer destination, this setting is ignored.

3.8.4 Special Functions

1) Scatter/gather function

When removing a part of image data and transferring it, image data cannot be handled as consecutive data, and the address changes dramatically depending on the special rule. Since DMA can transfer data only by using consecutive addresses, it is necessary to make required settings at locations where addresses changes.



Screen image

Screen data

The scatter/gather function can consecutively operate DMA settings (transfer source address, destination address, number of transfers, and transfer bus width) by re-loading them each time a specified number of DMA executions have completed via a pre-set "Linked List" where the CPU does not need to control the operation.

Setting "1" in the DMACCxLLI register enables/disables the operation.

The items that can be set with Linked List are configured with the following 4 words:

1) DMACCxSrcAddr
2) DMACCxDestAddr
3) DMACCxLLI
4) DMACCxControl

They can be used with the interrupt operation.

An interrupt depends on the Terminal Count Interrupt enable bit of the DMACCxControl register, and can be generated at the end of each LLI. When this bit is used, a condition can be added even during transfer using LLI to perform branch operation, etc. To clear the interrupt, control the appropriate bit of the DMACIntTCClear register.

2) Linked list operation

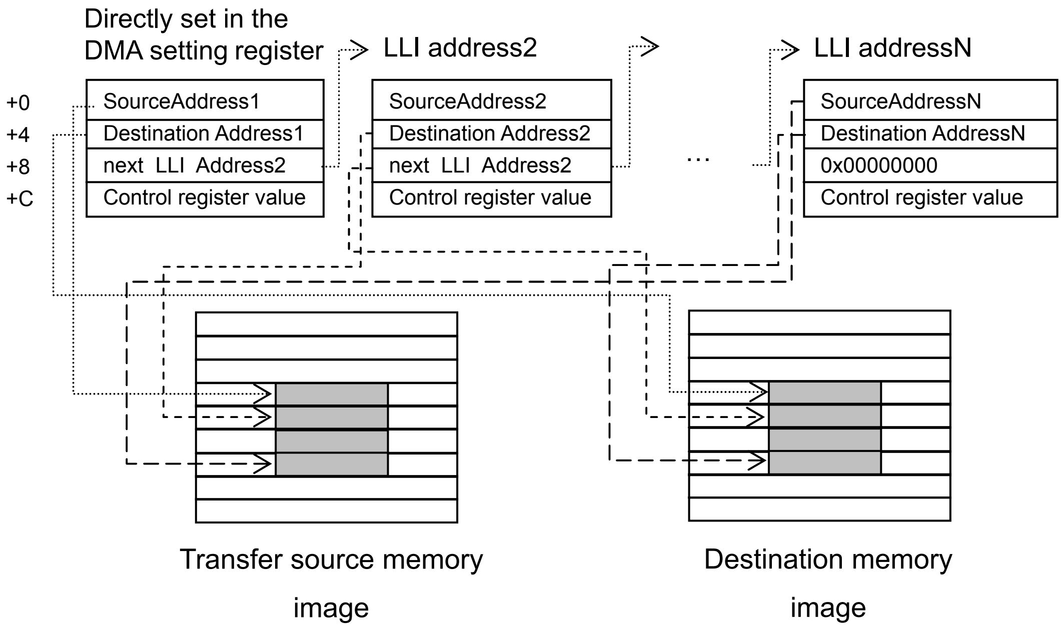
To operate the scatter/gather function, a transfer source and destination data areas need to be defined by creating a set of Linked Lists first.

Each setting is called LLI (LinkedList).

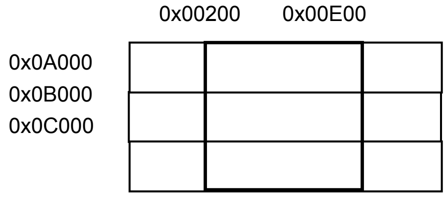
Each LLI controls the transfer of one block of data. Each LLI indicates normal DMA setting and controls transfer of successive data. Each time each DMA transfer is complete, the next LLI setting will be loaded to continue the DMA operation (Daisy Chain).

An example of the setting is shown below.

1. The first DMA transfer setting should be made directly in the DMA register.
2. The second and subsequent DMA transfer settings should be written in the addresses of the memory set in "next LLI AddressX."
3. To stop up to N'th DMA transfer, set "next LLI AddressX" to 0x00000000.

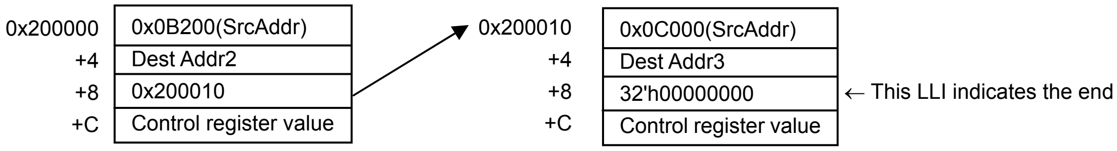


Example: Setting example to transfer the area enclosed by the square in the left figure.



DMACCxSrcAddr:	0x0A200
DMACCxDestAddr:	Destination address 1
DMACCxLLI:	0x200000
DMACCxControl:	Set the number of burst transfers and the number of transfers, etc.

Linked List



0x200000	0x0B200(SrcAddr)
+4	Dest Addr2
+8	0x200010
+C	Control register value

0x200010	0x0C000(SrcAddr)
+4	Dest Addr3
+8	32'h00000000
+C	Control register value

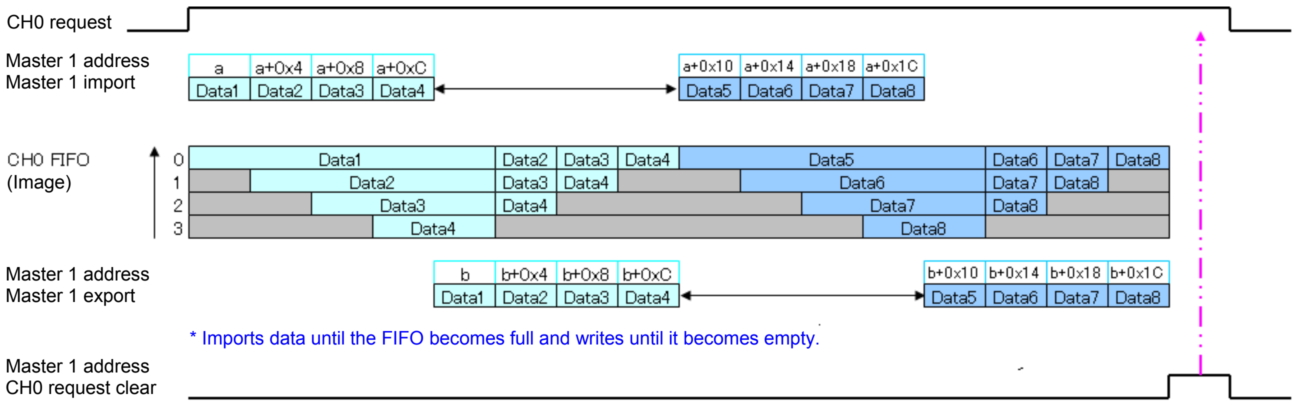
← This LLI indicates the end

3.8.5 DMA Transfer Setting Examples and Operation Overview

(Basic DMA transfer)

3.8.5.1 Example for transfer using one master in one channel

Setting item		Register name	Bit	Setting
Transfer address	Destination	DMACC0DestAddr	—	b
	Source	DMACC0SrcAddr	—	a
Address increment	Destination	DMACC0Control	DI	Increment
	Source	DMACC0Control	SI	Increment
Master	Destination	DMACC0Control	D	Master-1
	Source	DMACC0Control	S	Master-1
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	32-bit(0y010)
Burst size	Destination	DMACC0Control	DBSize	8-beat(0y010)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	8-word (0x8)



* Imports data until the FIFO becomes full and writes until it becomes empty.

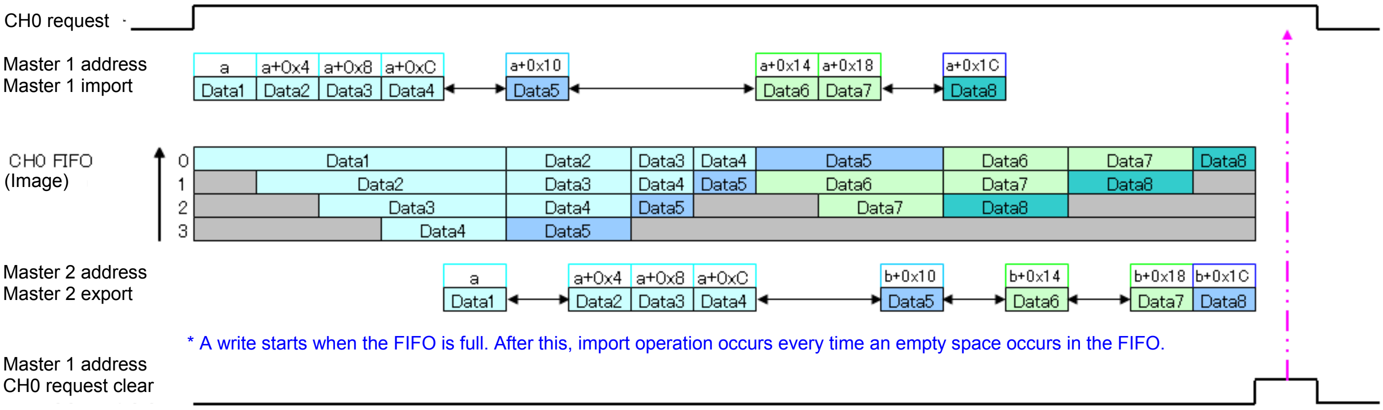
The channel imports data equivalent to the burst size from the transfer source address to the FIFO of the DMA channel using AHB Master 1. The DMA controller has one 32-bit wide and 4-stage deep FIFO per channel. Because data is transferred through this FIFO, the burst size is restricted. Therefore, in the above case where the burst size of data transferred at a request is set to 8, 4 beats of 32-bit wide data are transferred. In other words, two 4-word burst transfers are executed (the burst size of the DMA controller is not equal to the HBURST of the AHB protocol). Also, address increment is set so that the address increments on the basis of 32-bit wide data transfer and thus the address increments on the basis of 0x4.

Next, the channel writes the data imported to the FIFO of the DMA channel using AHB Master 1 to the transfer destination address. After the FIFO of the DMA channel becomes empty, the remaining 4 words to the burst size of 8 are imported and then similarly exported.

After data equivalent to the burst size has been transferred, the DMA channel asserts a DMA transfer request clear signal to the peripheral circuit. Responding to this, the peripheral circuit deasserts the DMA transfer request. At this time, if the set total transfer amount has not been met, the DMA controller will wait for the next DMA transfer request; after the total transfer amount has been reached, the DMA channel will be disabled. In the above case, because the total transfer amount is also set to 8, the DMA channel is disabled.

3.8.5.2 Example for transfer using two masters in one channel

Setting item		Register name	Bit	Setting
Transfer address	Destination	DMACC0DestAddr	—	b
	Source	DMACC0SrcAddr	—	a
Address increment	Destination	DMACC0Control	DI	Increment
	Source	DMACC0Control	SI	Increment
Master	Destination	DMACC0Control	D	Master-2
	Source	DMACC0Control	S	Master-1
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	32-bit(0y010)
Burst size	Destination	DMACC0Control	DBSize	8-beat(0y010)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	8-word (0x8)



* A write starts when the FIFO is full. After this, import operation occurs every time an empty space occurs in the FIFO.

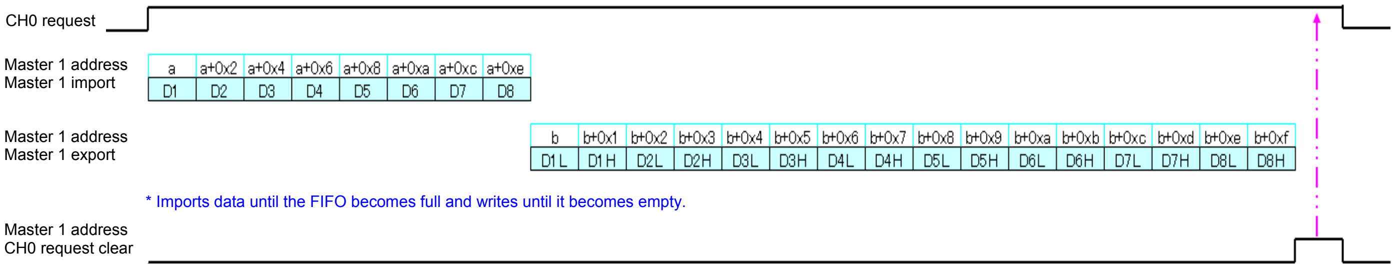
This example uses two masters, Master 1 and Master 2, for the transfer source and destination.

Importing the first 4 words is the same as the case where only AHB Master 1 is used, but exporting operation is different. When an empty space of undefined length occurs in the FIFO of the DMA channel due to the export by Master 2, Master 1 tries to import data to the FIFO.

In other words, because a burst transfer of undefined length, instead of a fixed length burst transfer, occurs and thus the number of transfers increases when two masters are used in one channel, if the amount of transfer data is bigger, the transfer efficiency may be reduced due to overhead.

3.8.5.3 Example 1 for transfer of differing data width (Data width: 16 bits for transfer source > 8 bits for transfer destination)

Setting item		Register name	Bit	Setting
Transfer address	Destination	DMACC0DestAddr	—	b
	Source	DMACC0SrcAddr	—	a
Address increment	Destination	DMACC0Control	DI	Increment
	Source	DMACC0Control	SI	Increment
Master	Destination	DMACC0Control	D	Master-2
	Source	DMACC0Control	S	Master-1
Data width	Destination	DMACC0Control	Dwidth	8-bit(0y000)
	Source	DMACC0Control	Swidth	16-bit(0y001)
Burst size	Destination	DMACC0Control	DBSize	16-beat(0y011)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	4-word (0x8)



* Imports data until the FIFO becomes full and writes until it becomes empty.

In this example, the data width for the transfer source is 16 bits, and 4 words of data is transferred.

Data width of 16 bits is set for the transfer source and thus the burst size can be up to 8 beats. The total number of transfers is 8, because it is based on the data width of the source side.

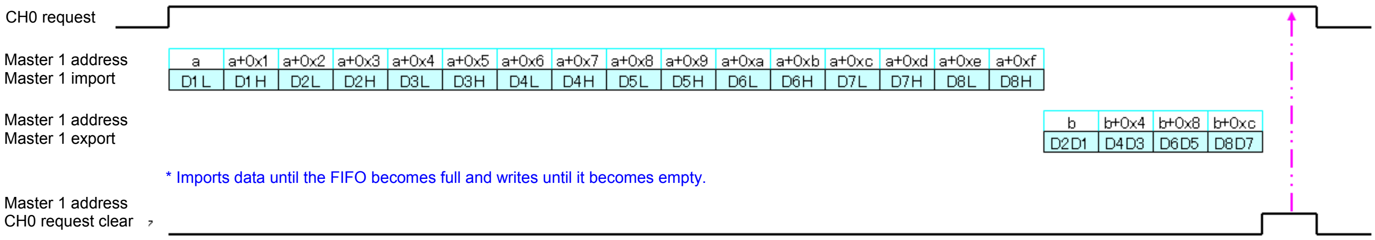
As for the transfer destination, data width of 8 bits is set and thus the burst size can be up to 16 beats.

At this time, the data width for the transfer source is 16 bits and thus the address increments by +0x2; the data width for the transfer destination is 8 bits and thus the address increments by +0x1.

* The data size for one transfer indicated by the total number of transfers is based on the data width of the source side.

3.8.5.4 Example 2 for transfer of differing data width (Data width: 8 bits for transfer source > 32 bits for transfer destination)

Setting item		Register name	Bit	Setting
Transfer address	Destination	DMACC0DestAddr	—	b
	Source	DMACC0SrcAddr	—	a
Address increment	Destination	DMACC0Control	DI	Increment
	Source	DMACC0Control	SI	Increment
Master	Destination	DMACC0Control	D	Master-2
	Source	DMACC0Control	S	Master-1
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	8-bit(0y000)
Burst size	Destination	DMACC0Control	DBSize	4-beat(0y001)
	Source	DMACC0Control	SBSize	16-beat(0y011)
Total number of transfers		DMACC0Control	TransferSize	4-word (0x10)



In this example, the data width for the transfer source is 8 bits, and 4 words of data is transferred.

Data width of 8 bits is set for the transfer source and thus the burst size can be up to 16 beats. The total number of transfers is 16, because it is based on the data width of the source side.

As for the transfer destination, data width of 32 bits is set and thus the burst size can be up to 4 beats.

At this time, the data width for the transfer source is 8 bits and thus the address increments by +0x1; the data width for the transfer destination is 32 bits and thus the address increments by +0x4.

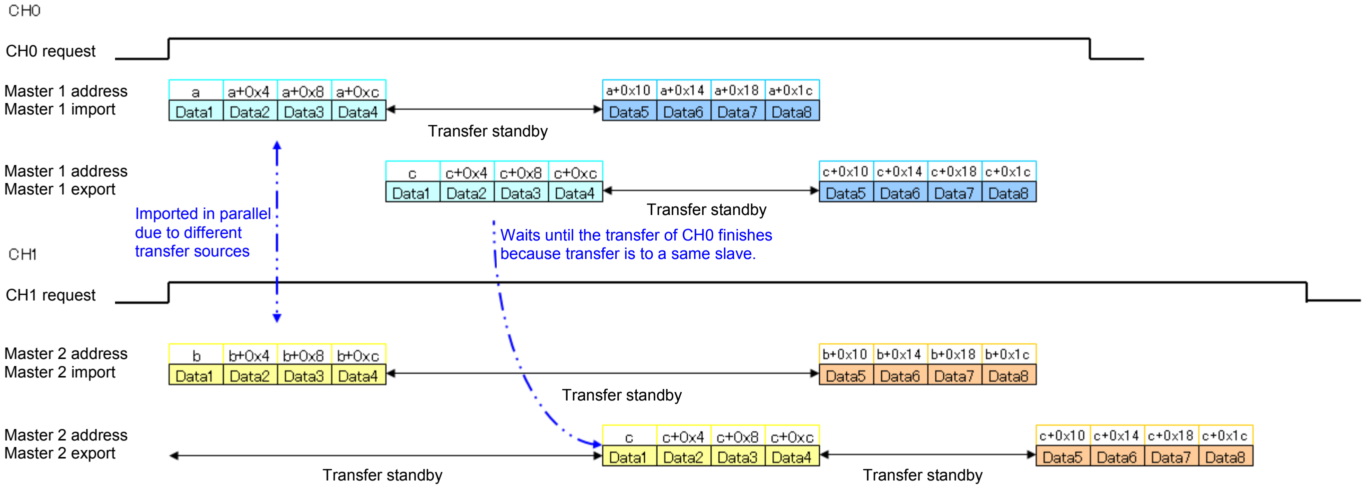
* The data size for one transfer indicated by the total number of transfers is based on the data width of the source side.

3.8.5.5 Example 1 of simultaneous requests (Transfer to a same slave (peripheral) by two channels)

When requests are output simultaneously, data is transferred according to fixed priorities. However, because the DMA FIFO size is restricted, the bus is released at each FIFO size when a burst transfer exceeding the FIFO size is executed (when non-locked transfer is used).

At this time, if a transfer with higher priority is waiting, the transfer with the higher priority will be executed first.

Setting item		Register name	Bit	Setting
CH0 (Master 1)	Destination	DMACC0DestAddr	—	c
	Source	DMACC0SrcAddr	—	a
CH1 (Master 2)	Destination	DMACC0DestAddr	—	c
	Source	DMACC0SrcAddr	—	b
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	32-bit(0y010)
Burst size	Destination	DMACC0Control	DBSize	8-beat(0y010)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	8-word (0x8)



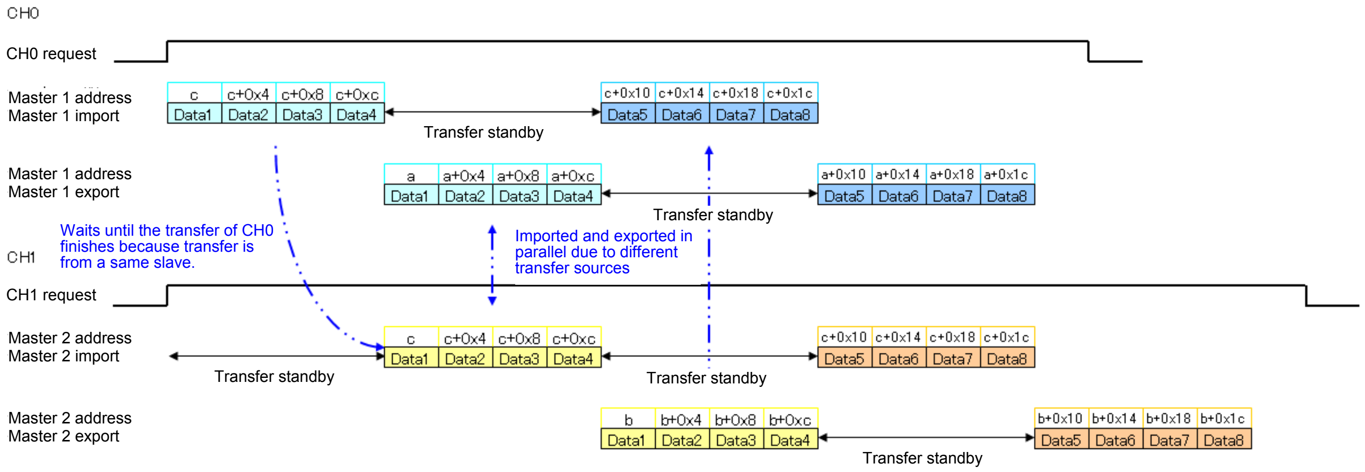
This example sets that CH0 and CH1 transfer data from different slaves (addresses a and b respectively) to a same slave (address c). When CH0 and CH1 receive DMA transfer requests at the same time, because the transfer sources are read from different slaves in the above example, data is imported in parallel irrespective of the priorities of the channels. Next, because exporting data from the FIFOs of the channels is from a same slave, the transfer of CH0 is executed first based on the priority. As with the "transfer using one master in one channel" described earlier, the bus ownership is given to CH1 after four burst transfers because the channel FIFO size is restricted irrespective of the burst size setting.

3.8.5.6 Example 2 of simultaneous requests (Transfer from a same slave by two channels)

When requests are output simultaneously, data is transferred according to fixed priorities. However, because the DMA FIFO size is restricted, the bus is released at each FIFO size when a burst transfer exceeding the FIFO size is executed (when non-locked transfer is used).

At this time, if a transfer with higher priority is waiting, the transfer with the higher priority will be executed first.

Setting item		Register name	Bit	Setting
CH0 (Master 1)	Destination	DMACC0DestAddr	—	a
	Source	DMACC0SrcAddr	—	c
CH1 (Master 2)	Destination	DMACC0DestAddr	—	b
	Source	DMACC0SrcAddr	—	c
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	32-bit(0y010)
Burst size	Destination	DMACC0Control	DBSize	8-beat(0y010)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	8-word (0x8)



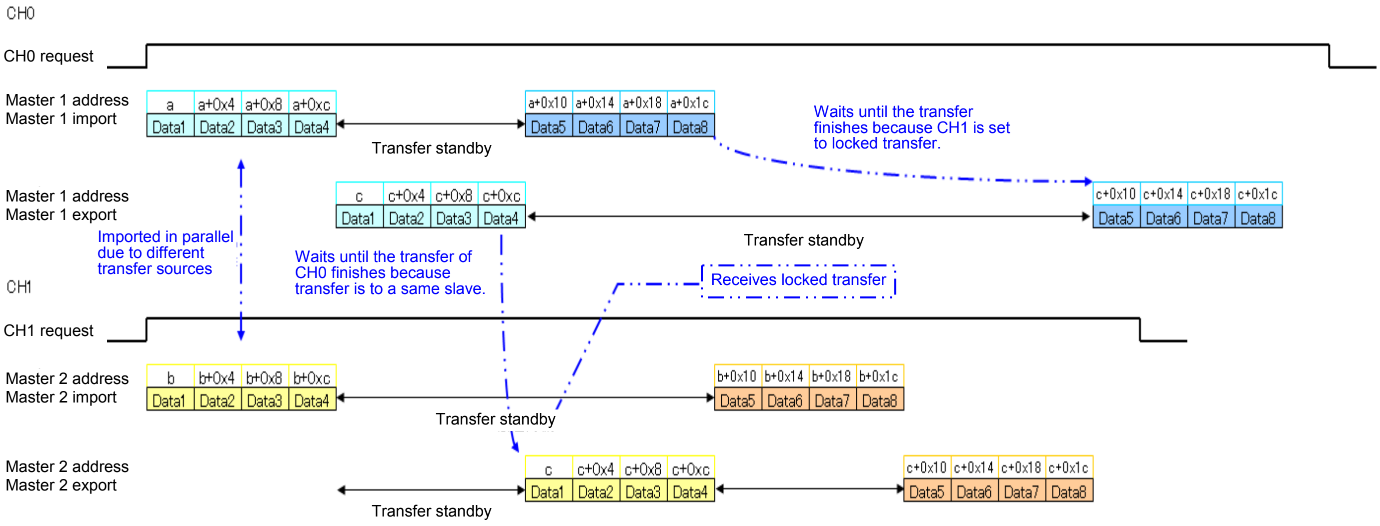
This example sets that CH0 and CH1 each transfer data from a same slave (address c) to different slaves (addresses a and b respectively). When CH0 and CH1 receive DMA transfer requests at the same time, because the transfer sources are read from a same slave in the above example, the transfer of CH0 is executed first based on the priorities of the channels. At this time, as with the "transfer using one master in one channel" described earlier, the bus ownership is given to CH1 after four burst transfers because the channel FIFO size is restricted irrespective of the burst size setting. Next, because the CH0's exporting data from the FIFO of the channel, and the CH1's importing data from the transfer source are accessed to different slaves, these processes are executed in parallel irrespective of the priorities.

3.8.5.7 Example 3 of simultaneous requests (Locked transfer. Transfer to a same slave by two channels)

This section describes an example where locked transfer is set additionally to "Example 1 of simultaneous requests."

When a burst transfer exceeding the FIFO size is executed, slaves switch in the transfer source and destination at each FIFO size. During this, the bus ownership of the transfer source slave is released when non-locked transfer is used, whereas, when locked transfer is used, the master keeps holding the bus ownership.

Setting item		Register name	Bit	Setting
CH0 (Master 1)	Destination	DMACC0DestAddr	—	c
	Source	DMACC0SrcAddr	—	a
	—	DMACC0Configuration	L	unlock (0y0)
CH1 (Master 2)	Destination	DMACC0DestAddr	—	c
	Source	DMACC0SrcAddr	—	b
	—	DMACC0Control	L	lock (0y1)
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	32-bit(0y010)
Burst size	Destination	DMACC0Control	DBSize	8-beat(0y010)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	8-word (0x8)



CH0 is set with non-locked transfer and CH1 is set with locked transfer. For the transfer source side, because data is transferred from different slaves in both CH0 and CH1, transfer is executed in parallel. At the point when first exporting data stored in the FIFOs of the channels, because the transfer destination is a same slave, the transfer of CH0 is executed first based on the priority.

After the data export for CH0 finishes, data export for CH1 is executed, where the locked transfer set between Master 2 and the address "c" slave is received.

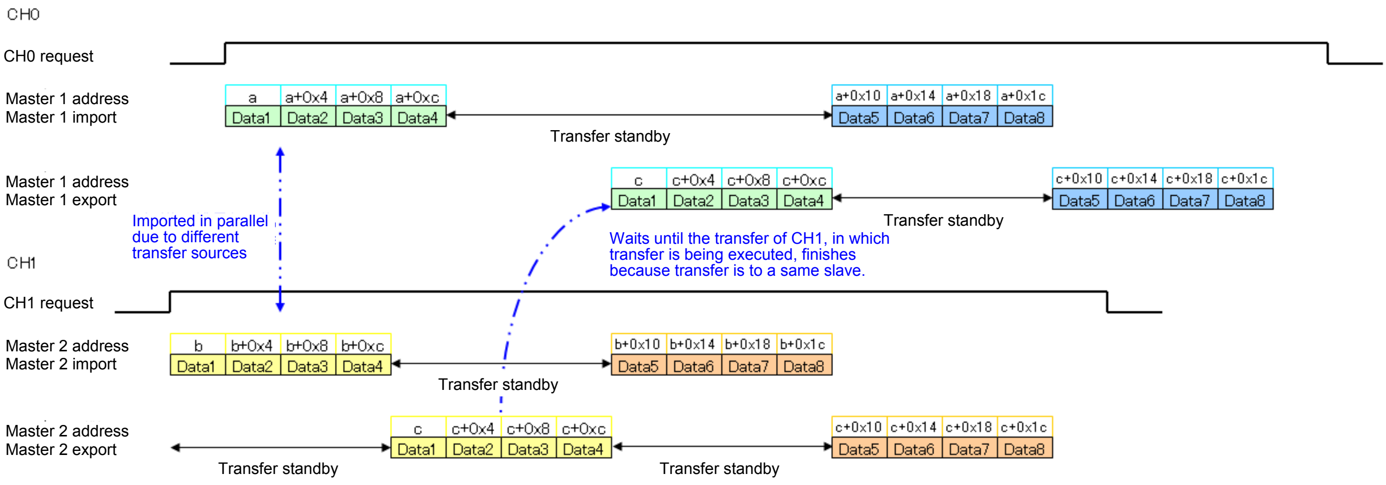
For non-locked transfer, in the moment when the data export for CH0 is executed after the data export for CH1 finishes, CH0 is forced to wait until the transfer of CH1 finishes because CH1 holds the bus ownership as locked transfer in this case. After CH1 finishes the transfer to the address "c" slave, CH1 releases the bus ownership. Then, the transfer of CH0 restarts to complete the transfer.

3.8.5.8 Example of a request made when a transfer is being executed (Transfer to a same slave by two channels)

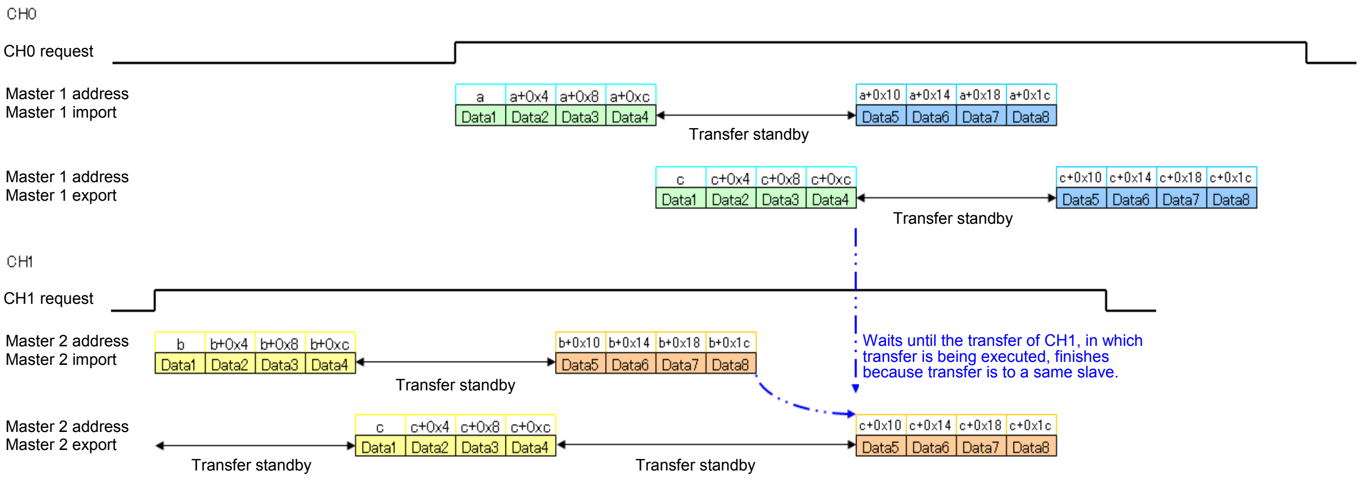
This section describes multiple requests with lags in their DMA transfer requests.

Basically, DMA transfer requests are processed for a transfer start on a first-come, first-served basis. In other words, unless requests occur at the same time, priorities are irrelevant.

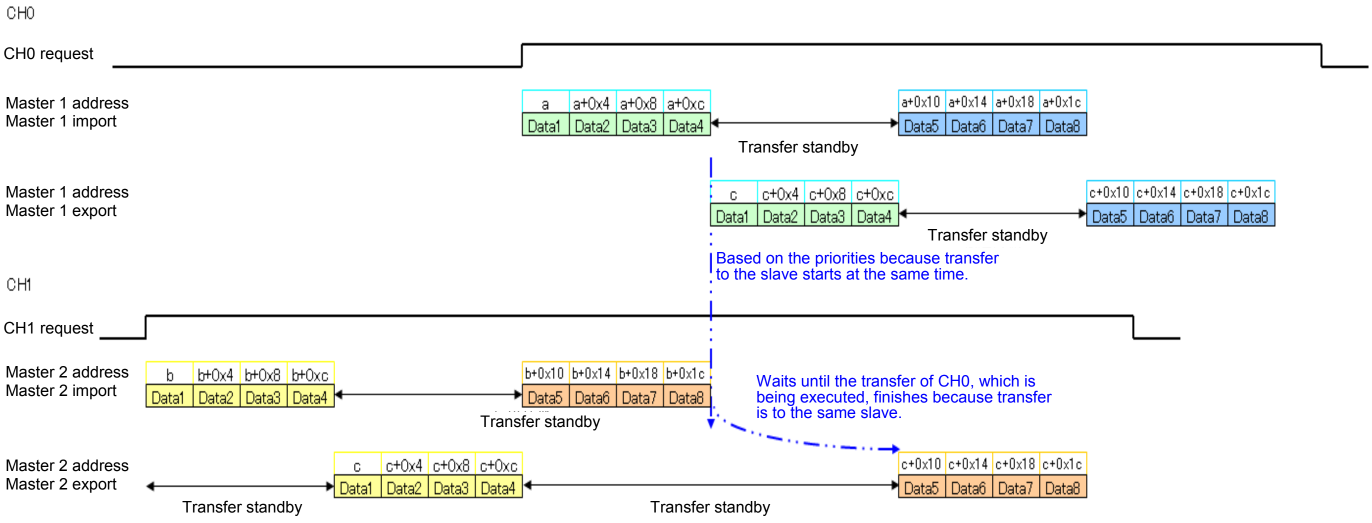
Setting item		Register name	Bit	Setting
CH0 (Master 1)	Destination	DMACC0DestAddr	—	c
	Source	DMACC0SrcAddr	—	a
CH1 (Master 2)	Destination	DMACC0DestAddr	—	c
	Source	DMACC0SrcAddr	—	b
Data width	Destination	DMACC0Control	Dwidth	32-bit(0y010)
	Source	DMACC0Control	Swidth	32-bit(0y010)
Burst size	Destination	DMACC0Control	DBSize	8-beat(0y010)
	Source	DMACC0Control	SBSize	8-beat(0y010)
Total number of transfers		DMACC0Control	TransferSize	8-word (0x8)



In this timing example, CH1 starts transferring first, whereas CH0 also starts transferring, because CH1 releases the bus ownership at every four burst transfers, which is the FIFO size restriction described earlier due to non-locked transfer. Note that when the transfer of CH1 to the slave is being executed (Master 2 is exporting data), CH0, which has the higher priority, waits until a burst transfer of CH1 finishes even though CH0 is also ready for data transfer.



In this timing example, CH1 starts transferring first, whereas CH0 also starts transferring, because CH1 releases the bus ownership at every four burst transfers, which is the FIFO size restriction described earlier due to non-locked transfer. A bus conflict occurs at the first data exporting of CH0 and the second data exporting of CH1, but, because a burst transfer of CH0 is being executed, CH1 waits until this transfer finishes before starting the transfer of CH1.



In this timing example, CH1 starts transferring first, whereas CH0 also starts transferring, because CH1 releases the bus ownership at every four burst transfers, which is the FIFO size restriction described earlier due to non-locked transfer. The first data exporting of CH0 occurs at the same timing as the second data exporting of CH1. In this case, CH0 starts transferring first because CH0 has the higher priority.

3.9 Port Function

This section shows the list of port pin functions and the list of input-output port settings showing how to set each pin.

Table 3.9.1 TMPM320C1D Port Pin Function List

Destination	Representative name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SSP0/1 UART0/1	PA	SP1CLK U1RTSn	SP1FSS U1CTSn	SP1DI U1RXD	SP1DO U1TXD	SP0CLK U0RTSn	SP0FSS U0CTSn	SP0DI U0RXD	SP0DO U0TXD
I2C0/1 Timer0/2/4/8 UART2/3	PB	I2C1SDA U3RXD	I2C1SCL U3TXD	I2C0SDA U2RXD	I2C0SCL U2TXD	PWM6OUT	PWM4OUT	PWM2OUT	PWM0OUT
SDHost	PC	SDCD	SDDAT2	SDDAT3	SDCMD	SDCLK	SDDAT0	SDDAT1	SDWP
I2S SSP2/3 CG	PD	BCK SP3CLK TMBCK	LRCK SP3FSS TMLRCK	MCLK SP3DI	DAO SP3DO	BCKi SP2CLK	LRCKi SP2FSS	DAIi SP2DI	SP2DO
USB INT	PE		− (Note 2)	USBPON	USBOCn	INT3	INT2	INT1	INT0
TPIU	PF	− (Note 2)	− (Note 2)	− (Note 2)	TRACECLK	TRACEDATA3	TRACEDATA2	TRACEDATA1	TRACEDATA0
SMC	PG	BE1n	BE0n	CS1n	CS0n	A22	A21	A20	A19

Note 1) The representative name "Px" in the table above shows the function of the general-purpose port.

Note 2) GPIO function only

Table 3.9.2 TMPM320C1D Input-Output Port Setting List (1/5)

*: don't care

Port name	Pin name	Specification	I/O register setting list					
			GPIOx DATA	GPIOx DIR	GPIOx FR1	GPIOx FR2	GPIOx OPD	GPIOx IE
Port A	PA0	Input port	*	0	0	0	-	-
		Output port	*	1	0	0		
		SP0DO output	*	*	1	0		
		U0TXD output	*	*	0	1		
	PA1	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP0DI input	*	*	1	0		
		U0RXD input	*	*	0	1		
	PA2	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP0FSS input-output	*	*	1	0		
		U0CTSn input	*	*	0	1		
	PA3	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP0CLK input-output	*	*	1	0		
		U0RTSn output	*	*	0	1		
	PA4	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP1DO output	*	*	1	0		
		U1TXD output	*	*	0	1		
	PA5	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP1DI input	*	*	1	0		
		U1RXD input	*	*	0	1		
	PA6	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP1FSS input-output	*	*	1	0		
		U1CTSn input	*	*	0	1		
	PA7	Input port	*	0	0	0		
		Output port	*	1	0	0		
		SP1CLK input-output	*	*	1	0		
		U1RTSn output	*	*	0	1		
Port B	PB0	Input port	*	0	0	0	*	-
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		PWM0OUT output (push-pull output)	*	*	1	0	0	
		PWM0OUT output (open drain output)	*	*	1	0	1	
	PB1	Input port	*	0	0	0	*	
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		PWM2OUT output (push-pull output)	*	*	1	0	0	
		PWM2OUT output (open drain output)	*	*	1	0	1	

Note) Do not use the ports with settings other than the above.

Table 3.9.3 TMPM320C1D Input-Output Port Setting List (2/5)

*: don't care

Port name	Pin name	Specification	I/O register setting list					
			GPIOx DATA	GPIOx DIR	GPIOx FR1	GPIOx FR2	GPIOx OPD	GPIOx IE
Port B	PB2	Input port	*	0	0	0	*	-
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		PWM4OUT output (push-pull output)	*	*	1	0	0	
		PWM4OUT output (open drain output)	*	*	1	0	1	
		-	-	-	-	-	-	
		-	-	-	-	-	-	
	PB3	Input port	*	0	0	0	*	
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		PWM6OUT output (push-pull output)	*	*	1	0	0	
		PWM6OUT output (open drain output)	*	*	1	0	1	
		-	-	-	-	-	-	
		-	-	-	-	-	-	
	PB4	Input port	*	0	0	0	*	
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		-	-	-	-	-	-	
		I2C0SCL input-output (open drain output)	*	0	1	0	1	
		U2TXD output (push-pull output)	*	*	0	1	0	
		U2TXD output (open drain output)	*	*	0	1	1	
	PB5	Input port	*	0	0	0	*	
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		-	-	-	-	-	-	
		I2C0SDA input-output (open drain output)	*	0	1	0	1	
		U2RXD input	*	*	0	1	0	
	PB6	Input port	*	0	0	0	*	
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		-	-	-	-	-	-	
		I2C1SCL input-output (open drain output)	*	0	1	0	1	
		U3TXD output (push-pull output)	*	*	0	1	0	
		U3TXD output (open drain output)	*	*	0	1	1	
	PB7	Input port	*	0	0	0	*	
		Output port (push-pull output)	*	1	0	0	0	
		Output port (open drain output)	*	1	0	0	1	
		-	-	-	-	-	-	
		I2C1SDA input-output (open drain output)	*	0	1	0	1	
		U3RXD input	*	*	0	1	0	
Port C	PC0	Input port	*	0	0	0	-	-
		Output port	*	1	0			
		SDWP input	*	*	1			
	PC1	Input port	*	0	0			
		Output port	*	1	0			
		SDDAT1 input-output	*	*	1			
	PC2	Input port	*	0	0			
		Output port	*	1	0			
		SDDAT0 input-output	*	*	1			

Note: Do not use the ports with settings other than the above.

Table 3.9.4 TMPM320C1D Input-Output Port Setting List (3/5)

*: don't care

Port name	Pin name	Specification	I/O register setting list					
			GPIOx DATA	GPIOx DIR	GPIOx FR1	GPIOx FR2	GPIOx OPD	GPIOx IE
Port C	PC3	Input port	*	0	0	0	-	-
		Output port	*	1	0			
		SDCLK output	*	*	1			
	PC4	Input port	*	0	0			
		Output port	*	1	0			
		SDCMD input-output	*	*	1			
	PC5	Input port	*	0	0			
		Output port	*	1	0			
		SDDAT3 input-output	*	*	1			
	PC6	Input port	*	0	0			
		Output port	*	1	0			
		SDDAT2 input-output	*	*	1			
	PC7	Input port	*	0	0			
		Output port	*	1	0			
		SDCD input	*	*	1			
Port D	PD0	Input port	*	0	0	0	-	-
		Output port	*	1	0	0		
		-	-	-	-	-		
		SP2DO output	*	*	0	1		
	PD1	Input port	*	0	0	0		
		Output port	*	1	0	0		
		DAIi input	*	*	1	0		
		SP2DI input	*	*	0	1		
	PD2	Input port	*	0	0	0		
		Output port	*	1	0	0		
		LRCKi input	*	*	1	0		
		SP2FSS input-output	*	*	0	1		
	PD3	Input port	*	0	0	0		
		Output port	*	1	0	0		
		BCKi input	*	*	1	0		
		SP2CLK input-output	*	*	0	1		
	PD4	Input port	*	0	0	0		
		Output port	*	1	0	0		
		DAO output	*	*	1	0		
		SP3DO output	*	*	0	1		
	PD5	Input port	*	0	0	0		
		Output port	*	1	0	0		
		MCLK output	*	*	1	0		
		SP3DI input	*	*	0	1		
	PD6	Input port	*	0	0	0		
		Output port	*	1	0	0		
		LRCK input-output	*	*	1	0		
		SP3FSS input-output	*	*	0	1		
		TMRCLK output (for BSIF mode only)	*	*	0	1		
	PD7	Input port	*	0	0	0		
		Output port	*	1	0	0		
		BCK input-output	*	*	1	0		
		SP3CLK input-output	*	*	0	1		
		TMBCK output (for BSIF mode only)	*	*	0	1		

Note) Do not use the ports with settings other than the above.

Table 3.9.5 TMPM320C1D Input-Output Port Setting List (4/5)

*: don't care

Port name	Pin name	Specification	I/O register setting list					
			GPIOx DATA	GPIOx DIR	GPIOx FR1	GPIOx FR2	GPIOx OPD	GPIOx IE
Port E	PE0	Input port	*	0	0	0	-	0
		-	-	-	-	-		-
		Output port	*	1	0	0		0
		-	-	-	-	-		-
		INT0 input	*	0	0	1		1
	PE1	Input port	*	0	0	0		0
		-	-	-	-	-		-
		Output port	*	1	0	0		0
		-	-	-	-	-		-
		INT1 input	*	0	0	1		1
	PE2	Input port	*	0	0	0		0
		-	-	-	-	-		-
		Output port	*	1	0	0		0
		-	-	-	-	-		-
		INT2 input	*	0	0	1		1
	PE3	Input port	*	0	0	0		0
		-	-	-	-	-		-
		Output port	*	1	0	0		0
		-	-	-	-	-		-
		INT3 input	*	0	0	1		1
	PE4	Input port	*	0	0	0		0
		-	-	-	-	-		-
		Output port	*	1	0	0		0
		-	-	-	-	-		-
		USBOCn input	*	*	1	1		0
	PE5	Input port	*	0	0	0		0
		Output port	*	1	0	0		0
		USBPON output	*	*	1	0		0
	PE6	Input port	*	0	0	0		0
		Output port	*	1	0	0		0
Port F	PF0	Input port	*	0	0	0	-	-
		Output port	*	1	0			
		TRACEDATA0 output	*	*	1			
	PF1	Input port	*	0	0			
		Output port	*	1	0			
		TRACEDATA1 output	*	*	1			
	PF2	Input port	*	0	0			
		Output port	*	1	0			
		TRACEDATA2 output	*	*	1			
	PF3	Input port	*	0	0			
		Output port	*	1	0			
		TRACEDATA3 output	*	*	1			
	PF4	Input port	*	0	0			
		Output port	*	1	0			
		TRACECLK output	*	*	1			
	PF5	Input port	*	0	0			
		Output port	*	1	0			
	PF6	Input port	*	0	0			
		Output port	*	1	0			

Note) Do not use the ports with settings other than the above.

Table 3.9.6 TMPM320C1D Input-Output Port Setting List (5/5)

*: don't care

Port name	Pin name	Specification	I/O register setting list					
			GPIOx DATA	GPIOx DIR	GPIOx FR1	GPIOx FR2	GPIOx OPD	GPIOx IE
Port F	PF7	Input port	*	0	0	0	-	-
		Output port	*	1	0			
Port G	PG0	Input port	*	0	0	0	-	-
		Output port	*	1	0			
		A19 output	*	*	1			
	PG1	Input port	*	0	0			
		Output port	*	1	0			
		A20 output	*	*	1			
	PG2	Input port	*	0	0			
		Output port	*	1	0			
		A21 output	*	*	1			
	PG3	Input port	*	0	0			
		Output port	*	1	0			
		A22 output	*	*	1			
	PG4	Input port	*	0	0			
		Output port	*	1	0			
		CS0n output	*	*	1			
	PG5	Input port	*	0	0			
		Output port	*	1	0			
		CS1n output	*	*	1			
	PG6	Input port	*	0	0			
		Output port	*	1	0			
		BE0n output	*	*	1			
	PG7	Input port	*	0	0			
		Output port	*	1	0			
		BE1n output	*	*	1			

Note) Do not use the ports with settings other than the above.

Table 3.9.7 TMPM320C1 Address and Initial Value List

Register name	R/W	Address	Description of register	Meaning		Register initial value						
				0	1	PortA	PortB	PortC	PortD	PortE	PortF	PortG
GPIOxDATA	R/W	0x000 -0x3FC	Data register	-	-	0x00	0x00	0x00	0x00	0x00	0x00	0x00
GPIOxDIR	R/W	0x400	Data direction register	Input port	Output port	0x00	0x00	0x00	0x00	0x00	0x00	0x00
GPIOxFR1	R/W	0x424	Function register 1	GPIO	Function 1 input or output enable	0x00	0x00	0x00	0x00	0x00	0x00	0x00
GPIOxFR2	R/W	0x428	Function register 2	GPIO	Function 2 input or output enable	0x00	0x00	0x00	0x00	0x00	0x00	0x00
GPIOxIS	R/W	0x804	Interrupt detection register	Edge	Level	-	-	-	-	0x00	-	-
GPIOxIBE	R/W	0x808	Interrupt both-edge register	Both edge disable	Both edge enable	-	-	-	-	0x00	-	-
GPIOxIEV	R/W	0x80C	Interrupt event register	Falling or L level	Rising or H level	-	-	-	-	0x00	-	-
GPIOxIE	R/W	0x810	Interrupt enable register	Disable	Enable	-	-	-	-	0x00	-	-
GPIOxRIS	RO	0x814	Pre-interrupt enable status register	Not requested	Requested	-	-	-	-	0x00	-	-
GPIOxMIS	RO	0x818	Post-interrupt enable status register	Not requested	Requested	-	-	-	-	0x00	-	-
GPIOxIC	WO	0x81C	Interrupt clear register	-	Request clear	-	-	-	-	0x00	-	--
GPIOxODE	R/W	0xC00	Open drain output enable register	Push-pull output	Open drain output	-	0x00	-	--	-	-	--

- No register present

Table 3.9.8 TMPM320C1D Register Base Address List

Port	base address
PortA	0x4000_8000
PortB	0x4000_9000
PortC	0x4000_A000
PortD	0x4000_B000
PortE	0x4000_C000
PortF	0x4000_D000
PortG	0x4000_E000

3.9.1 Data Registers

[Notes on data registers]

All data registers not only read/write 8-bit data simultaneously but also read/write them while masking certain bits (bit mask function).

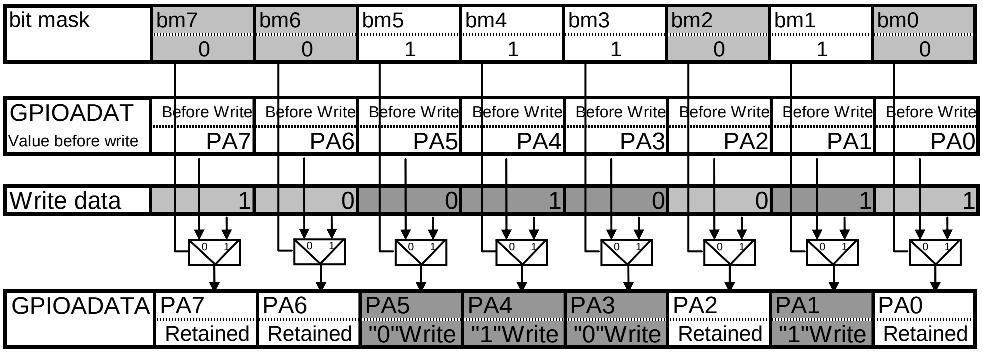
Addresses to access a data register consist of 256 address spaces (0x00-0xFF), from 0x0000 to 0x03FC. (Imagine that the address is shifted to the upper position by 2 bits. The lowest 2 bits do not have any significance. Therefore, one in every four addresses is valid such as 0x0000, 0x0004, and so on.)

Access to this 256-address area results in access to a single data register, but valid bits differ depending on the accessed address.

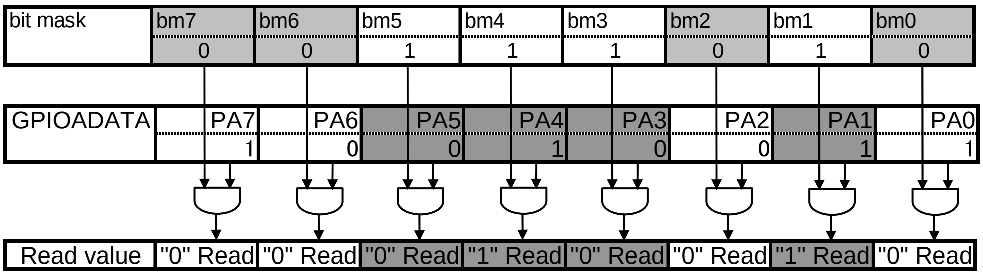
Bits [9:2] of the accessed address correspond to Bits [7:0] of the data register, and the data will be masked. Bits of Address 1 access the data register, and bits of Address 0 will be masked.

Address[9:2]	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2
Bit mask	bm7	bm6	bm5	bm4	bm3	bm2	bm1	bm0

- Bit mask WRITE example (example: writing 0x93 to Address 0x00E8 of Port A)



- Bit mask READ example (example: reading 0x12 from Address 0x00E8 of Port A)



Note) All the bits are valid in the access to 0x03FC, and no bits are valid in the access to 0x0000.

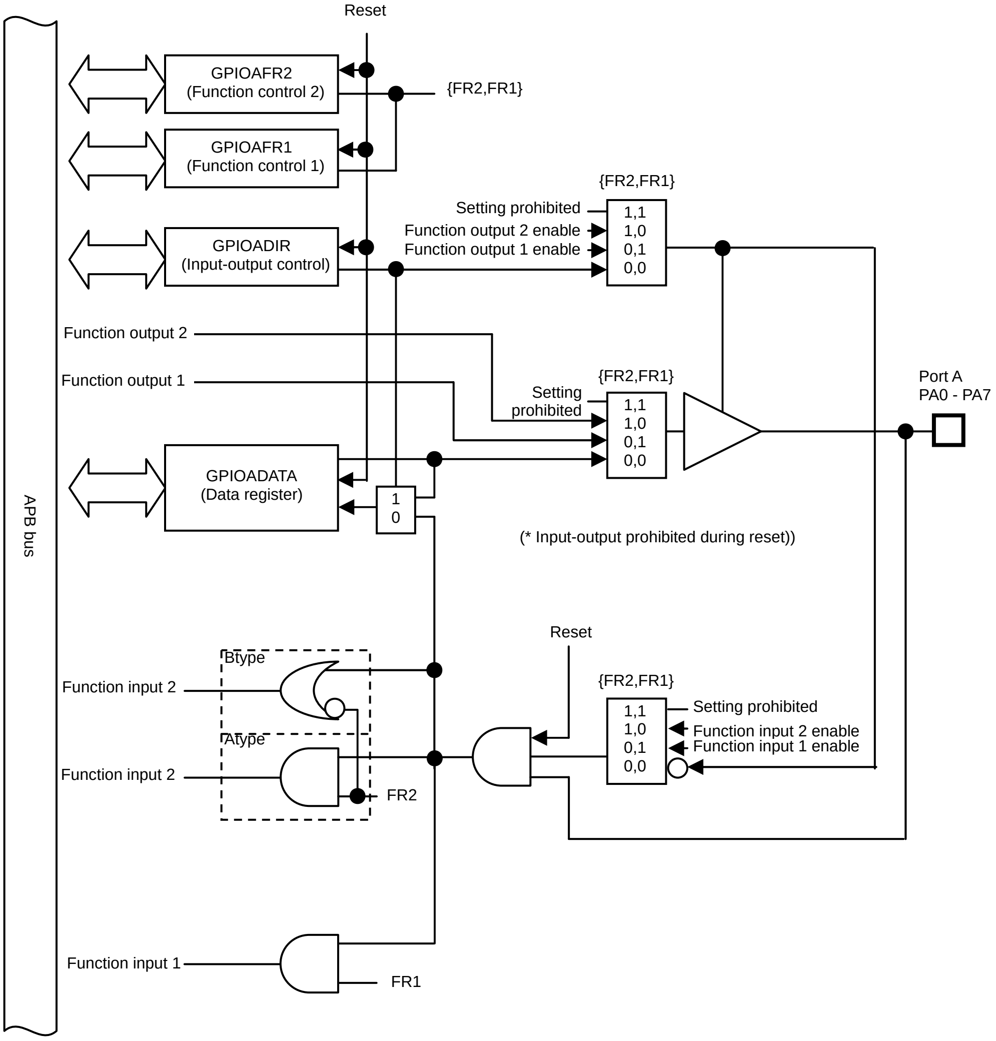
3.9.2 Port Function Setting

This chapter describes, on a port basis, the setting of Port A through Port G, which can also function as general-purpose ports. Note that access to each SFR must be WORD (32-bit) access.

3.9.2.1 PORTA

Port A can be used as an 8-bit general-purpose input-output pin.

It can also be used for the SSP function (SP1CLK, SP1FSS, SP1DI, SP1DO, SP0CLK, SP0FSS, SP0DI, SP0DO) and the UART function (U1RTSn, U1CTSn, U1RXD, U1TXD, U0RTSn, U0CTSn, U0RXD, U0TXD). They all can be set on a bit basis.



Port name	Function output 2	Function output 1	Function output 2 enable	Function output 1 enable	Function input 2 (A or B type)	Function input 1	Function input 2 enable	Function input 1 enable
PA0	U0TXD	SP0DO	"1"	*SSP	"-"	"-"	"0"	"0"
PA1	"0"	"0"	"0"	"0"	U0RXD (Btype)	SP0DI	"1"	"1"
PA2	"0"	SP0FSS	"0"	*SSP	U0CTSn(Atype)	SP0FSS	"1"	"1"
PA3	U0RTSn	SP0CLK	"1"	*SSP	"-"	SP0CLK	"0"	"1"
PA4	U1TXD	SP1DO	"1"	*SSP	"-"	"-"	"0"	"0"
PA5	"0"	"0"	"0"	"0"	U1RXD (Btype)	SP1DI	"1"	"1"
PA6	"0"	SP1FSS	"0"	*SSP	U1CTSn(Atype)	SP1FSS	"1"	"1"
PA7	U1RTSn	SP1CLK	"1"	*SSP	"-"	SP1CLK	"0"	"1"

(* Depends on the control from the SSP main unit. PA0, PA2, PA3: Ch0; PA4, PA6, PA7: Ch1)

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose input	GPIOADATA	GPIOADIR	GPIOAFR1	GPIOAFR2
	*	0	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose output	GPIOADATA	GPIOADIR	GPIOAFR1	GPIOAFR2
	*	1	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Output	Output	Output	Output	Output	Output	Output	Output

SSP settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
SSP	GPIOADATA	GPIOADIR	GPIOAFR1	GPIOAFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SP1CLK	SP1FSS	SP1DI	SP1DO	SP0CLK	SP0FSS	SP0DI	SP0DO

UART settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
UART	GPIOADATA	GPIOADIR	GPIOAFR1	GPIOAFR2
	*	*	0	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
U1RTSn	U1CTSn	U1RXD	U1TXD	U0RTSn	U0CTSn	U0RXD	U0TXD

base address = 0x4000_8000

Register Name	Address (base+)	Description
GPIOADATA	0x0000 to 0x03FC	PortA Data Register
GPIOADIR	0x0400	PortA Data Direction Register
GPIOAFR1	0x0424	PortA Function Register1
GPIOAFR2	0x0428	PortA Function Register2

1. GPIOADATA (Port Data Register)

Address = (0x4000_8000)+(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7:0]	PA7:0	RW	0y00000000	Bm7:0	Port A data register

[Explanation]

a. <PA7:0>

Data register: A register that retains data

Refer to notes on the data register for the bit mask function.

2. GPIOADIR (Port Data Direction Register)

Address ((0x4000_8000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PA7C to PA0C	RW	0y00000000	Port A data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PA7C to PA0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port

0y0: Input

0y1: Output

3. GPIOAFR1 (Port Function Register1)

Address ((0x4000_8000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	-	-	Undefined	Read undefined. Write as zero.
[7:0]	PA7F1 to PA0F1	RW	0y00000000	Port A function register 1

[Explanation]

a. <PA7F1 to PA0F1>

Function register 1: Control register 1 for switching the function

4. GPIOAFR2 (Port Function Register2)

Address ((0x4000_8000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	-	-	Undefined	Read undefined. Write as zero.
[7:0]	PA7F2 to PA0F2	RW	0y00000000	Port A function register 2

[Explanation]

a. <PA7F2 to PA0F2>

Function register 2: Control register 2 for switching the function

(Note) The function registers 1 and 2 must be set mutually exclusively. Even momentarily, never write "1" in both registers at the same time.

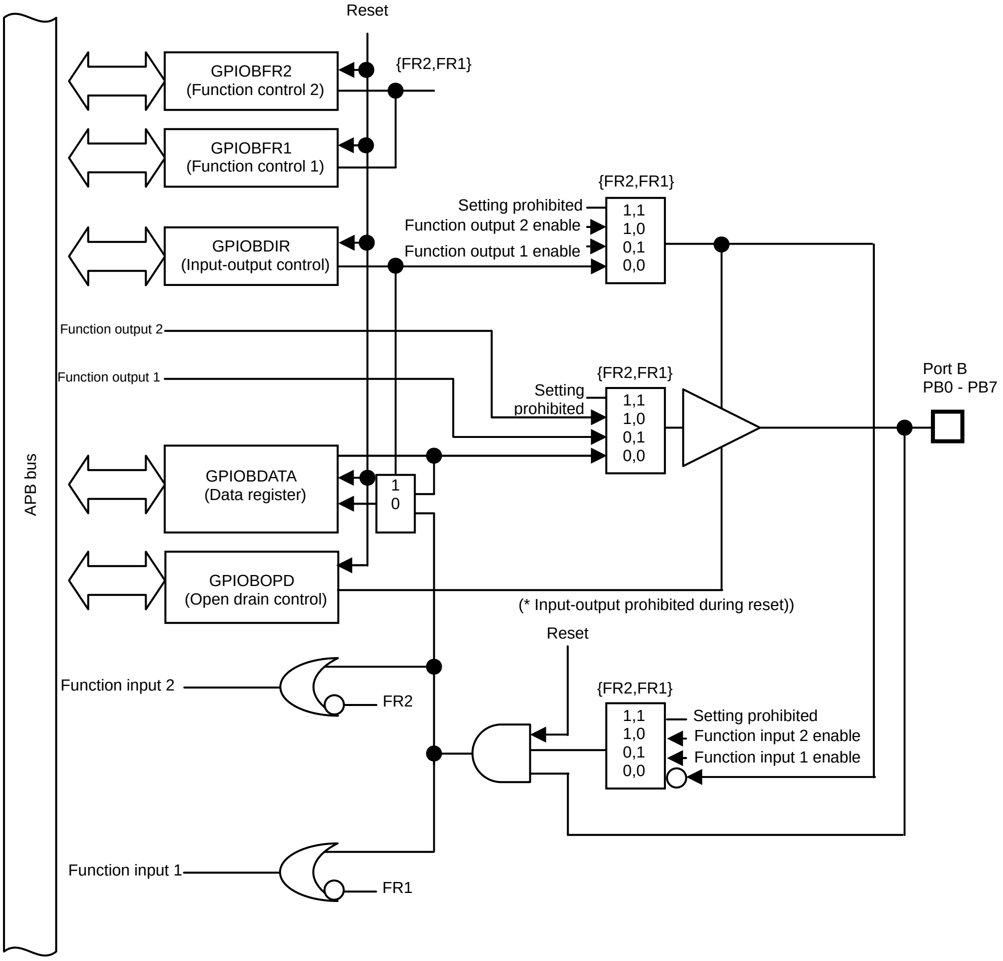
Table 3.9.9 Function register function list

Mode	GPIOAFR1	GPIOAFR2
General-purpose	0	0
Function 1 (SSP)	1	0
Function 2 (UART)	0	1
Disable	1	1

3.9.2.2 PORTB

Port B can be used as an 8-bit general-purpose input-output pin.

It can also be used for the I2C function (I2C1SDA, I2C1SCL, I2C0SDA, I2C0SCL), the timer function (PWM6OUT, PWM4OUT, PWM2OUT, PWM0OUT), and the UART function (U3RXD, U3TXD, U2RXD, U2TXD). They all can be set on a bit basis.



Port name	Function output 2	Function output 1	Function output 2 enable	Function output 1 enable	Function input 2	Function input 1	Function input 2 enable	Function input 1 enable
PB0	Setting prohibited	PWM0OUT	Setting prohibited	"1"	"-"	"-"	Setting prohibited	"0"
PB1	Setting prohibited	PWM2OUT	Setting prohibited	"1"	"-"	"-"	Setting prohibited	"0"
PB2	Setting prohibited	PWM4OUT	Setting prohibited	"1"	"-"	"-"	Setting prohibited	"0"
PB3	Setting prohibited	PWM6OUT	Setting prohibited	"1"	"-"	"-"	Setting prohibited	"0"
PB4	U2TXD	I2C0SCL	"1"	"1"	"-"	I2C0SCL	"0"	"1"
PB5	"0"	I2C0SDA	"0"	"1"	U2RXD	I2C0SDA	"1"	"1"
PB6	U3TXD	I2C1SCL	"1"	"1"	"-"	I2C1SCL	"0"	"1"
PB7	"0"	I2C1SDA	"0"	"1"	U3RXD	I2C1SDA	"1"	"1"

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Open drain enable
General-purpose input	GPIOBDATA	GPIOBDIR	GPIOBFR1	GPIOBFR2	GPIOBODE
	*	0	0	0	*

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Open drain enable
General-purpose output	GPIOBDATA	GPIOBDIR	GPIOBFR1	GPIOBFR2	GPIOBODE
	*	1	0	0	0/1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Output	Output	Output	Output	Output	Output	Output	Output

Note) Open drain works for all bits.

I2C/ timer settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Open drain enable
I2C/ timer	GPIOBDATA	GPIOBDIR	GPIOBFR1	GPIOBFR2	GPIOBODE
	*	0/* (Note 1)	1	0	0/1 (Note 2)

Note 1) Set to "0" for I2C, and set to "don't care" for timer.

Note 2) To use I2C, set the open drain setting to "1."

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2C1SDA	I2C1SCL	I2C0SDA	I2C0SCL	PWM6OUT	PWM4OUT	PWM2OUT	PWM0OUT

Notel) Open drain works for all bits.

UART settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Open drain enable
UART	GPIOBDATA	GPIOBDIR	GPIOBFR1	GPIOBFR2	GPIOBODE
	*	*	0	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
U3RXD	U3TXD	U2RXD	U2TXD	–	–	–	–

Note) Open drain works for all bits.

base address = 0x4000_9000

Register Name	Address (base+)	Description
GPIOBDATA	0x0000 to 0x03FC	PortB Data Register
GPIOBDIR	0x0400	PortB Data Direction Register
GPIOBFR1	0x0424	PortB Function Register1
GPIOBFR2	0x0428	PortB Function Register2
GPIOBODE	0x0C00	PortB Open-drain Output Enable Register

1. GPIOBDATA (Port Data Register)

Address ((0x4000_9000)+(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7:0]	PB7:0	R/W	0y00000000	Bm7:0	Port B data register

[Explanation]

a. <PB7:0>

Data register: A register that retains data.

Refer to notes on the data register for the bit mask function.

2. GPIOBDIR (Port Data Direction Register)

Address = (0x4000_9000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PB7C to PB0C	R/W	0y00000000	Port B data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PB7C:PB0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port

0y0: Input

0y1: Output

3. GPIOBFR1(Port Function Register1)

Address = (0x4000_9000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PB7F1 to PB0F1	R/W	0y00000000	Port B function register 1

[Explanation]

a. <PB7F1 to PB0F1>

Function register 1: Control register 1 for switching the function

4. GPIOBFR2(Port Function Register2)

Address = (0x4000_9000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:4]	PB7F2 to PB4F2	R/W	0y0000	Port B function register 2
[3:2]	Reserved	R/W	0y00	Read zero. Write as zero.
[1:0]	Reserved	R/W	0y00	Read zero. Write as zero.

[Explanation]

a. <PB7F2 to PB4F2>

Function register 2: Control register 2 for switching the function

Note) The function registers 1 and 2 must be set mutually exclusively. Even momentarily, never write "1" in both registers at the same time.

Table 3.9.10 Function register function list

Mode	GPIOBFR1	GPIOBFR2
General-purpose	0	0
Function 1 (timer /I2C)	1	0
Function 2 (UART)	0	1
Disable	1	1

5. GPIOBODE (Port Open-drain Output Enable Register)

Address = (4000_9000) +(0x0C00)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PB7ODE to PB0ODE	R/W	0y00000000	Port B OPD output enable register 0y0: Push-pull output 0y1: Open drain (Pch disable) Output

[Explanation]

a. <PB7ODE to PB0ODE>

OPD output enable register: A register for selecting either push-pull output or OPD output

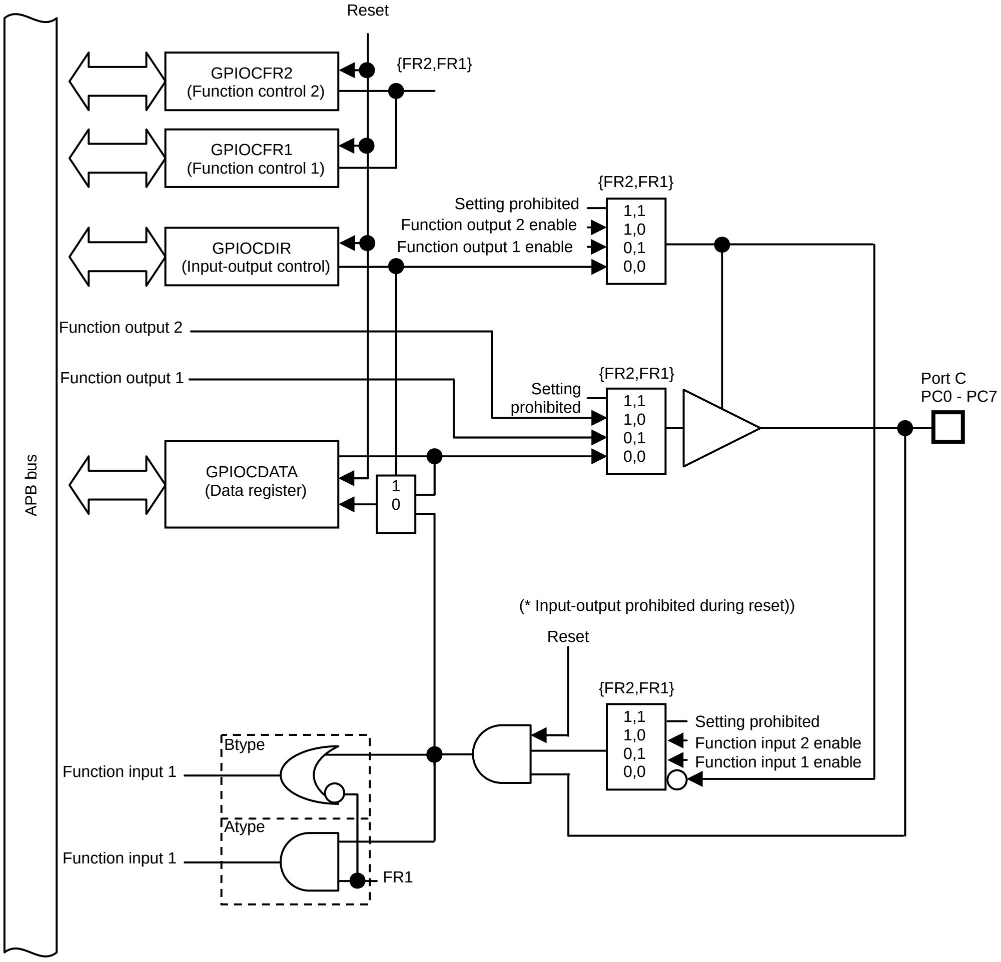
0y0: Push-pull output

0y1: Open drain (Pch disable) output

3.9.2.3 PORTC

Port C can be used as an 8-bit general-purpose input-output pin.

It can also be used for the SD host controller function (SDCD, SDDAT2, SDDAT3, SDCMD, SDCLK, SDDAT0, SDDAT1, SDWP). They all can be set on a bit basis.



Port name	Function output 2	Function output 1	Function output 2 enable	Function output 1 enable	Function input 1 (A or B type)	Function input 2 enable	Function input 1 enable
PC0	Setting prohibited	"0"	Setting prohibited	"0"	SDWP(B type)	Setting prohibited	"1"
PC1	Setting prohibited	SDDAT1	Setting prohibited	*SDHost	SDDAT1(B type)	Setting prohibited	"1"
PC2	Setting prohibited	SDDAT0	Setting prohibited	*SDHost	SDDAT0(B type)	Setting prohibited	"1"
PC3	Setting prohibited	SDCLK	Setting prohibited	"1"	"-"	Setting prohibited	"0"
PC4	Setting prohibited	SDCMD	Setting prohibited	*SDHost	SDCMD(B type)	Setting prohibited	"1"
PC5	Setting prohibited	SDDAT3	Setting prohibited	*SDHost	SDDAT3(A type)	Setting prohibited	"1"
PC6	Setting prohibited	SDDAT2	Setting prohibited	*SDHost	SDDAT2(B type)	Setting prohibited	"1"
PC7	Setting prohibited	"0"	Setting prohibited	"0"	SDCD(B type)	Setting prohibited	"1"

(* Output control from the SDHost main unit. PC1, PC2, PC4, PC5, PC6)

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose input	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2
	*	0	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose output	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2
	*	1	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Output	Output	Output	Output	Output	Output	Output	Output

SDHost settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
SDHost	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SDCD	SDDAT2	SDDAT3	SDCMD	SDCLK	SDDAT0	SDDAT1	SDWP

base address = 0x4000_A000

Register Name	Address (base+)	Description
GPIOCDATA	0x0000 to 0x03FC	PortC Data Register
GPIOCDIR	0x0400	PortC Data Direction Register
GPIOCFR1	0x0424	PortC Function Register1
GPIOCFR2	0x0428	PortC Function Register2

1. GPIOCDATA (Port Data Register)

Address = (0x4000_A000) +(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7:0]	PC7:0	R/W	0y00000000	Bm7:0	Port C data register

[Explanation]

a. <PC7:0>

Data register: A register that retains data.

Refer to notes on the data register for the bit mask function.

2. GPIOCDIR (Port Data Direction Register)

Address = (0x4000_A000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PC7C to PC0C	R/W	0y00000000	Port C data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PD7C to PD0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOCFR1 (Port Function Register1)

Address = (0x4000_A000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PC7F1 to PC0F1	R/W	0y00000000	Port C function register 1

[Explanation]

a. <PD7F1 to PD0F1>

Function register 1: Control register 1 for switching the function

4. GPIOCFR2 (Port Function Register2)

Address = (0x4000_A000) +(0x0428)

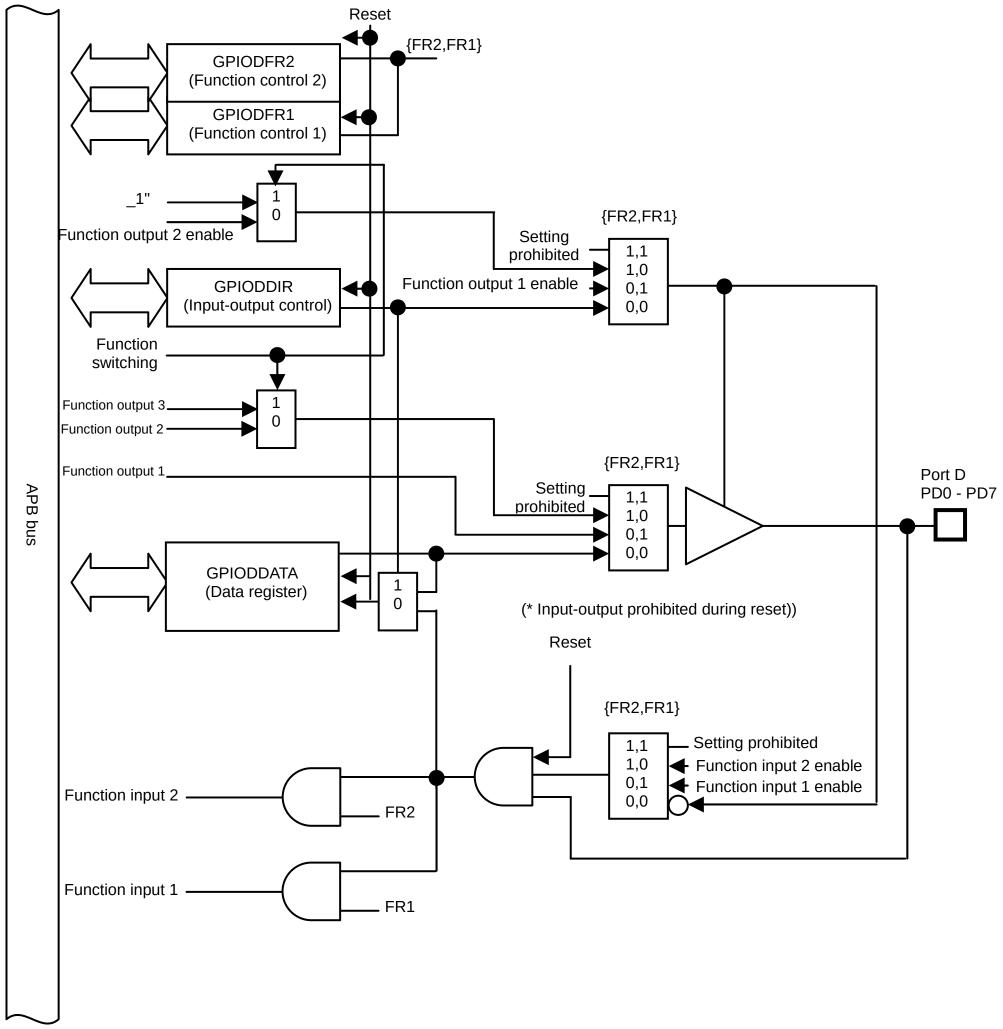
Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	Reserved	R/W	0y00000000	Read zero. Write as zero.

[Explanation]

3.9.2.4 PORTD

Port D can be used as an 8-bit general-purpose input-output pin.

It can also be used as the I2S function (BCK, LRCK, MCLK, DAO, BCKi, LRCKi, DAIi), the SSP function (SP3CLK, SP3FSS, SP3DI, SP3DO, SP2CLK, SP2FSS, SP2DI, SP2DO), and the BSIF function (TMBCK, TMLRCK). They all can be set on a bit basis.



Port name	Function output 3	Function output 2	Function output 1	Function switching	Function output 2 enable	Function output 1 enable	Function input 2	Function input 1	Function input 2 enable	Function input 1 enable
PD0	"0"	SP2DO	Setting prohibited	"0"	*SSP	Setting prohibited	"-"	"-"	"0"	Setting prohibited
PD1	"0"	"0"	"0"	"0"	"0"	"0"	SP2DI	DAIi	"1"	"1"
PD2	"0"	SP2FSS	"0"	"0"	*SSP	"0"	SP2FSS	LRCKi	"1"	"1"
PD3	"0"	SP2CLK	"0"	"0"	*SSP	"0"	SP2CLK	BCKi	"1"	"1"
PD4	"0"	SP3DO	DAO	"0"	*SSP	"1"	"-"	"-"	"0"	"0"
PD5	"0"	"0"	MCLK	"0"	"0"	"1"	SP3DI	"-"	"1"	"0"
PD6	TMLRCK	SP3FSS	LRCK	BSIF function	*SSP	*/LRCLKOSEL	SP3FSS	LRCK	"1"	*LRCLKOSEL
PD7	TMBCK	SP3CLK	BCK	BSIF function	*SSP	*/BCKOSEL	SP3CLK	BCK	"1"	*BCKOSEL

(* Depends on the control from the SSP main unit. PD0, PD2, PD3: Ch2; PD4, PD6, PD7:Ch3)

(* Depends on the control from the I2S main unit. PD6, PD7)

Note: Be sure that setting the SSP must be performed when SSP operation is disabled.

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose input	GPIODDATA	GPIODDIR	GPIODFR1	GPIODFR2
	*	0	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose output	GPIODDATA	GPIODDIR	GPIODFR1	GPIODFR2
	*	1	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Output	Output	Output	Output	Output	Output	Output	Output

SSP settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
I2S	GPIODDATA	GPIODDIR	GPIODFR1	GPIODFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BCK	LRCK	MCLK	DAO	BCKi	LRCKi	DAIi	-

SSP/BSIF settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
SSP/BSIF	GPIODDATA	GPIODDIR	GPIODFR1	GPIODFR2
	*	*	0	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SP3CLK TMBCK	SP3FSS TMLRCK	SP3DI	SP3DO	SP2CLK	SP2FSS	SP2DI	SP2DO

Note) The SP3CLK/TMBCK of Bit 7 and the SP3FSS/TMLRCK of Bit 6 are selected in the CG circuit base address = 0x4000_B000

Register Name	Address (base+)	Description
GPIODDATA	0x0000 to 0x03FC	PortD Data Register
GPIODDIR	0x0400	PortD Data Direction Register
GPIODFR1	0x0424	PortD Function Register1
GPIODFR2	0x0428	PortD Function Register2

1. GPIODDATA (Port Data Register)

Address = (0x4000_B000)+(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7:0]	PD7:0	R/W	0y00000000	Bm7:0	Port D data register

[Explanation]

a. <PD7:0>

Data register: A register that retains data.

Refer to notes on the data register for the bit mask function.

2. GPIODDIR (Port Data Direction Register)

Address = (0x4000_B000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PD7C to PD0C	R/W	0y00000000	Port D data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PD7C to PD0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIODFR1 (Port Function Register1)

Address = (0x4000_B000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PD7F1 to PD0F1	R/W	0y00000000	Port D function register 1

[Explanation]

a. <PD7F1 to PD0F1>

Function register 1: Control register 1 for switching the function

4. GPIODFR2 (Port Function Register2)

Address = (0x4000_B000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PD7F2 to PD0F2	R/W	0y00000000	Port D function register 2

[Explanation]

a. <PD7F2 to PD0F2>

Function register 2: Control register 2 for switching the function

(Note) The function registers 1 and 2 must be set mutually exclusively. Even momentarily, never write "1" in both registers at the same time.

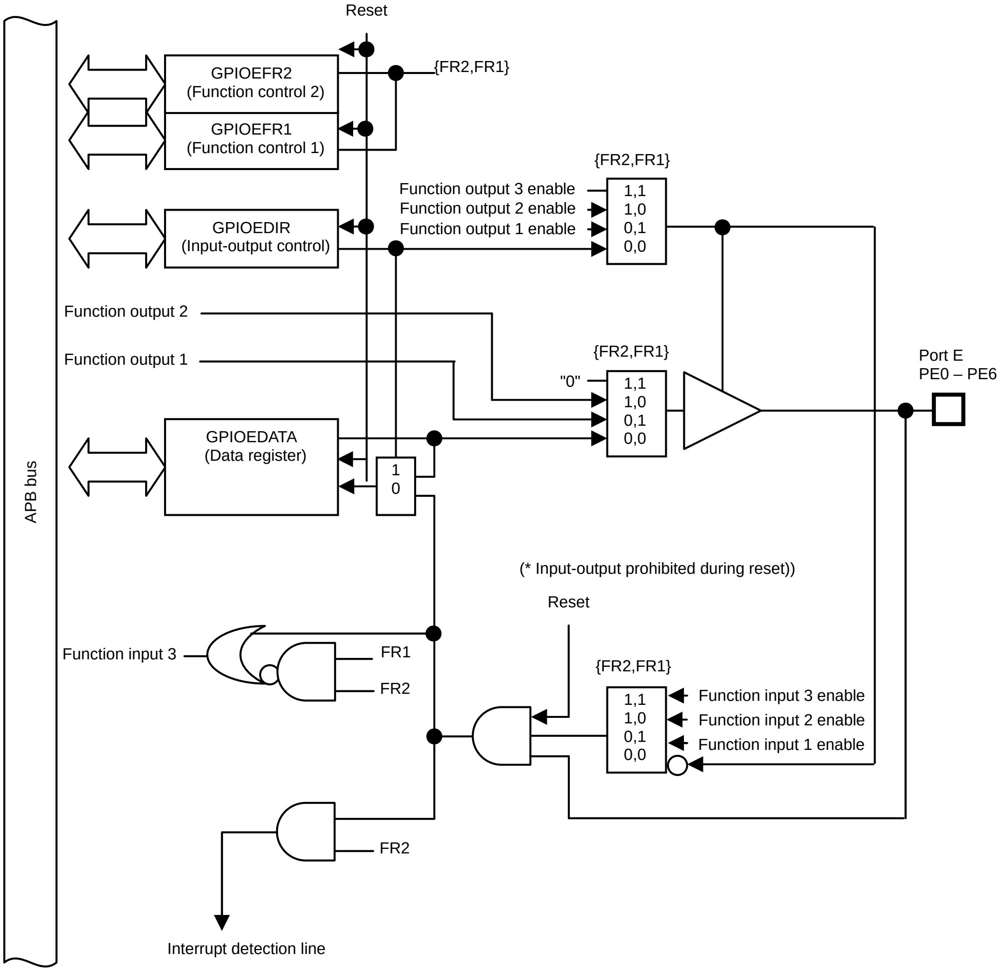
Table 3.9.11 Function Register Function List

Mode	GPIODFR1	GPIODFR2
General-purpose	0	0
Function 1 (I2S)	1	0
Function 2 (SSP/BSIF)	0	1
Disable	1	1

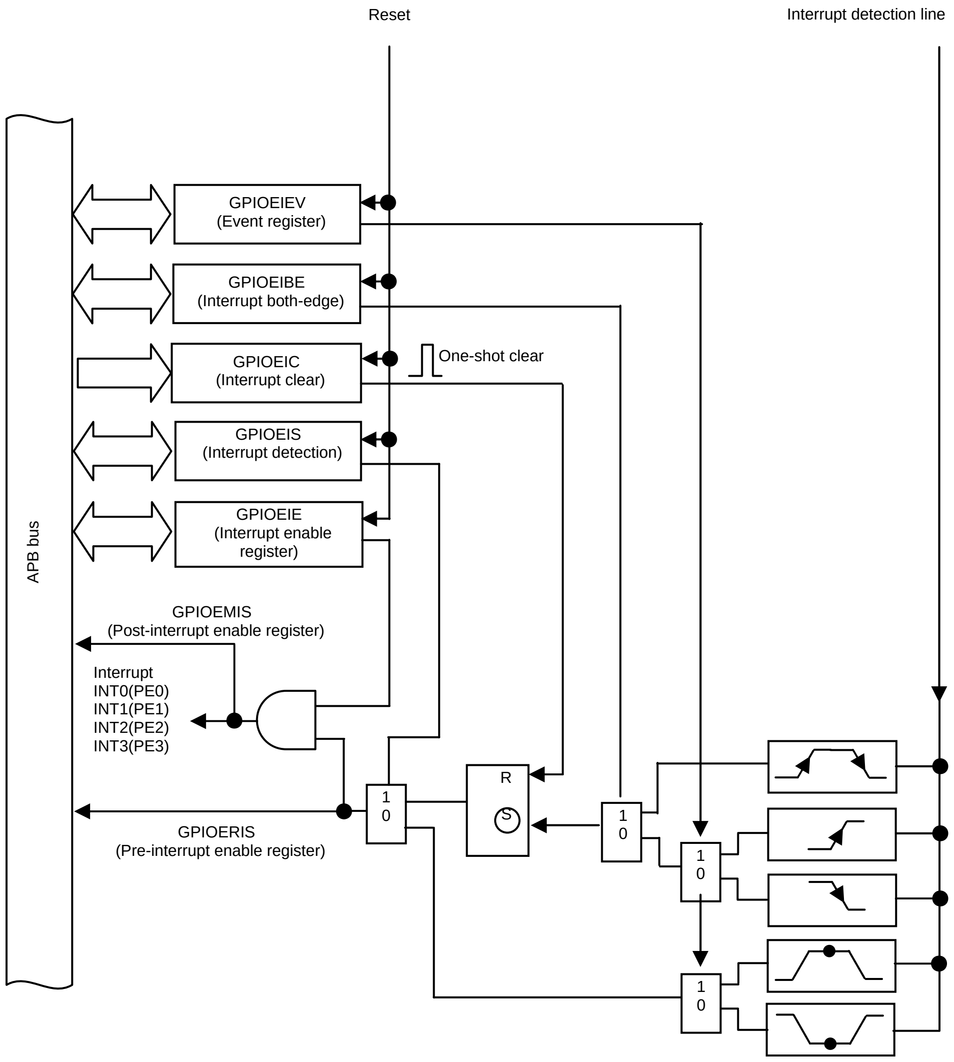
3.9.2.5 PORTE

Port E can be used as a 7-bit general-purpose input-output pin (Bit 7 is not used).

It can also be used for the USB function (USBPON, USBCOn) and the external interrupt pins (INT3 to INT0). They all can be set on a bit basis.



Port name	Function output 2	Function output 1	Function output 3 enable	Function output 2 enable	Function output 1 enable	Function input 3	Function input 3 enable	Function input 2 enable	Function input 1 enable
PE0	"0"	Setting prohibited	Setting prohibited	"0"	Setting prohibited	Setting prohibited	Setting prohibited	"1"	Setting prohibited
PE1	"0"	Setting prohibited	Setting prohibited	"0"	Setting prohibited	Setting prohibited	Setting prohibited	"1"	Setting prohibited
PE2	"0"	Setting prohibited	Setting prohibited	"0"	Setting prohibited	Setting prohibited	Setting prohibited	"1"	Setting prohibited
PE3	"0"	Setting prohibited	Setting prohibited	"0"	Setting prohibited	Setting prohibited	Setting prohibited	"1"	Setting prohibited
PE4	Setting prohibited	Setting prohibited	"0"	Setting prohibited	Setting prohibited	USBCOn	"1"	Setting prohibited	Setting prohibited
PE5	Setting prohibited	USBOPN	Setting prohibited	Setting prohibited	"1"	Setting prohibited	Setting prohibited	Setting prohibited	"0"
PE6	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited


Reset
Interrupt detection line
APB bus
GPIOEIEV
(Event register)
GPIOEIBE
(Interrupt both-edge)
GPIOEIC
(Interrupt clear)
One-shot clear
GPIOEIS
(Interrupt detection)
GPIOEIE
(Interrupt enable
register)
GPIOEMIS
(Post-interrupt enable register)
Interrupt
INT0(PE0)
INT1(PE1)
INT2(PE2)
INT3(PE3)
GPIOERIS
(Pre-interrupt enable register)
R
S
1
0

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Interrupt enable
General-purpose input	GPIOEDATA	GPIOEDIR	GPIOEFR1	GPIOEFR2	GPIOEIE
	*	0	0	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Interrupt enable
General-purpose output	GPIOEDATA	GPIOEDDIR	GPIOEFR1	GPIOEFR2	GPIOEIE
	*	1	0	*	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	Output	Output	Output	Output	Output	Output	Output

External interrupt settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Interrupt enable
External interrupt	GPIOEDATA	GPIOEDIR	GPIOEFR1	GPIOEFR2	GPIOEIE
	*	0	0	1	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	–	–	–	INT3	INT2	INT1	INT0

Note) Only Bits 3 through 0 support external interrupts.

USB settings

Purpose	Data value	Input-output change	Function change 1	Function change 2	Interrupt enable
Interrupt	GPIOEDATA	GPIOEDIR	GPIOEFR1	GPIOEFR2	GPIOEIE
	*	*	1	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	–	USBPON	USBCOn	–	–	–	–

base address = 0x4000_C000

Register Name	Address (base+)	Description
GPIOEDATA	0x0000 to 0x03FC	PortE Data Register
GPIOEDIR	0x0400	PortE Data Direction Register
GPIOEFR1	0x0424	PortE Function Register1
GPIOEFR2	0x0428	PortE Function Register2
GPIOEIS	0x0804	PortE Interrupt Selection Register (Level and Edge)
GPIOEIBE	0x0808	PortE Interrupt Selection Register (Fellow edge and Both edge)
GPIOEIEV	0x080C	PortE Interrupt Selection Register (Fall down edge/Low level and Rising up edge/High level)
GPIOEIE	0x0810	PortE Interrupt Enable Register
GPIOERIS	0x0814	PortE Interrupt Status Register (Raw)
GPIOEMIS	0x0818	PortE Interrupt Status Register (Masked)
GPIOEIC	0x081C	PortE Interrupt Clear Register

1. GPIOEDATA (Port Data Register)

Address = (0x4000_C000)+(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7]	Reserved	R/W	0y0	Bm7	Read zero. Write as zero.
[6:0]	PE6:0	R/W	0y0000000	Bm6:0	Port E data register

[Explanation]

a. <PE7:0>

Data register: A register that retains data.

Refer to notes on the data register for the bit mask function.

2. GPIOEDIR (Port Data Direction Register)

Address = (0x4000_C000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	Reserved	R/W	0y0	Read zero. Write as zero.
[6:0]	PE6C to PE0C	R/W	0y0000000	Port E data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PE7C to PE0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOEFR1 (Port Function Register1)

Address = (0x4000_C000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:6]	Reserved	R/W	0y00	Read zero. Write as zero.
[5:4]	PE5F1 to PE4F1	R/W	0y00	Port E function register 1
[3:0]	Reserved	R/W	0y0000	Read zero. Write as zero.

[Explanation]

a. <PE5F1 to PE4F1>

Function register 1: Control register 1 for switching the function

4. GPIOEFR2 (Port Function Register2)

Address = (0x4000_C000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:5]	Reserved	R/W	0y000	Read zero. Write as zero.
[4:0]	PE4F2 to PE0F2	R/W	0y00000	Port E function register 2

[Explanation]

a. <PE4F2 to PE0F2>

Function register 2: Control register 2 for switching the function

5. GPIOEIS (Port Interrupt Selection Register (Level and Edge))

Address ((0x4000_ 000) +(0x0804)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:4]	Reserved	R/W	0y0000	Read zero. Write as zero.
[3:0]	PE3IS to PE0IS	R/W	0y0000	Port E interrupt direction register (each bit) 0y0: Edge detection 0y1: Level detection

[Explanation]

a. <PE3IS to PE0IS>

Interrupt detection register: A register for selecting either edge detection or level detection

0y0: Edge detection

0y1: Level detection

6. GPIOEIBE (Port Interrupt Selection Register (Fellow edge and Both edge))

Address = (0x4000_C000) +(0x0808)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:4]	Reserved	R/W	0y0000	Read zero. Write as zero.
[3:0]	PE3IBE to PE0IBE	R/W	0y0000	Port E interrupt both-edge register (Each bit) 0y0: Single edge 0y1: Double edge

[Explanation]

a. <PE3IBE to PE0IBE>

Interrupt both-edge register: A register for selecting either single edge or double edge

0y0: Single edge

0y1: Double edge

7. GPIOEIEV (Port Interrupt Selection Register (Fall down edge/Low level and Rising up edge/High level))

Address = (0x4000_C000) +(0x080C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:4]	Reserved	R/W	0y0000	Read zero. Write as zero.
[3:0]	PE3IEV to PE0IEV	R/W	0y0000	Port E interrupt event register (each bit) 0y0: Falling edge/Low level 0y1: Rising edge/High level

[Explanation]

a. <PE3IEV to PE0IEV>

Interrupt event register: A register for controlling falling or rising in edge detection, and for selecting either Low or High level in level detection

0y0: Falling edge/Low level

0y1: Rising edge/High level

8. GPIOEIE (Port Interrupt Enable Register)

Address = (0x4000_C000) +(0x0810)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:4]	Reserved	R/W	0y0000	Read zero. Write as zero.
[3:0]	PE3IE to PE0IE	R/W	0y0000	Port E interrupt enable register (each bit) 0y0: Disable 0y1: Enable

[Explanation]

a. <PE3IE to PE0IE>

Interrupt enable register: A register for enabling or disabling interrupts

0y0: Disable

0y1: Enable

9. GPIOERIS (Port Interrupt Status Register (Raw))

Address = (0x4000_C000) +(0x0814)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined.
[7:4]	Reserved	RO	0y0000	Read zero.
[3:0]	PE3RIS to PE0RIS	RO	0y0000	Port E pre-interrupt enable status register (Each bit) 0y0: Not requested 0y1: Requested

[Explanation]

a. <PE3RIS to PE0RIS>

Pre-interrupt enable status register: A register for monitoring the pre-masking interrupt status of the enabled register

0y0: Not requested

0y1: Requested

10. GPIOEMIS (Port Interrupt Status Register (Masked))

Address ((0x4000_C000) +(0x0818)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined.
[7:4]	Reserved	RO	0y0000	Read zero.
[3:0]	PE3MIS to PE0MIS	RO	0y0000	Port E post-interrupt enable status register (Each bit) 0y0: Not requested 0y1: Requested

[Explanation]

a. <PE3MIS to PE0MIS>

Post-interrupt enable status register: A register for monitoring the post-masking status of the enabled register

0y0: Not requested

0y1: Requested

11. GPIOEIC (Port Interrupt Clear Register)

Address = (0x4000_C000) +(0x081C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Write as zero.
[7:4]	Reserved	WO	0y0000	Write as zero.
[3:0]	PE3IC to PE0IC	WO	0y0000	Port E interrupt clear register (each bit) 0y0: Disabled 0y1: Clear request

[Explanation]

a. <PE3IC to PE0IC>

Interrupt clear register: A register for clearing edge interrupts

0y0: Disabled

0y1: Clear request

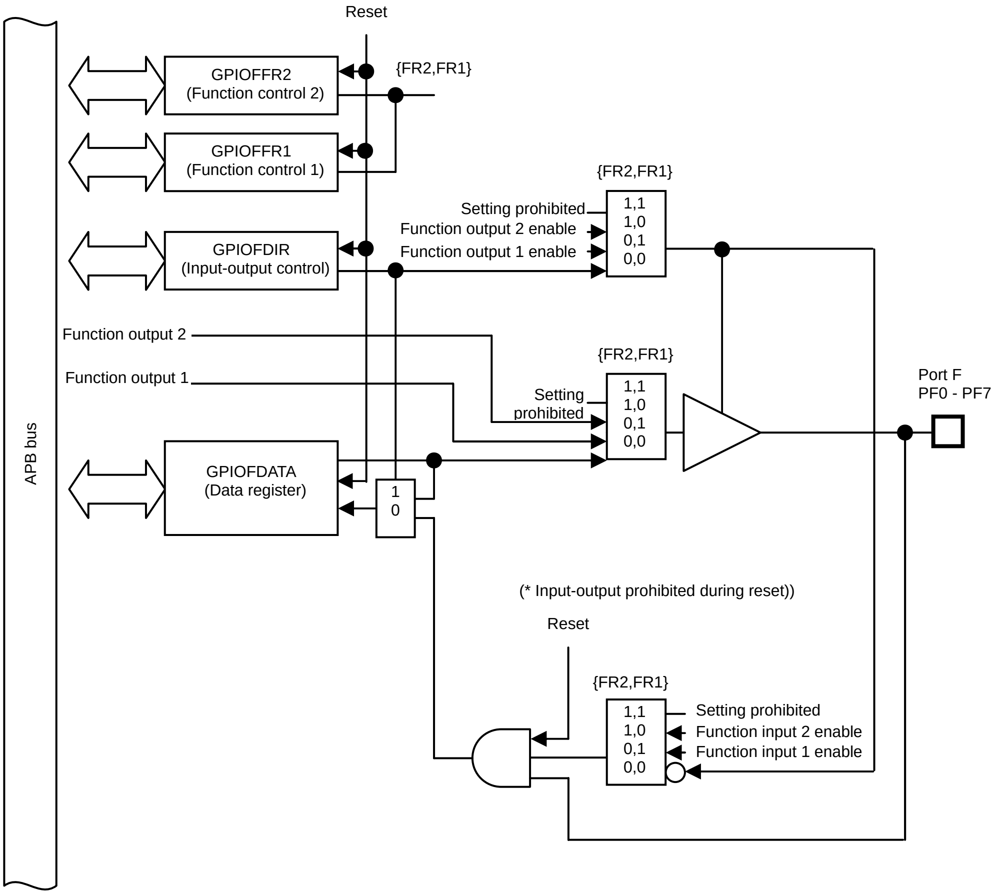
The following is the list of external interrupt register settings, where combinations of the bit settings of each register are listed:

Register setting				Detection method	Output		
GPIOEIS (Interrupt direction register)	GPIOEIBE (Interrupt both-edge register)	GPIOEIEV (Interrupt event register)	GPIOEIE (Interrupt enable register)		GPIOERIS (Pre-interrupt enable status register)	GPIOEMIS (Post-interrupt enable status register)	INTx (INTx interrupt)
0	0	0	0	Falling edge detection	Detection enable	Detection disable (0x00)	Detection disable
		1		Rising edge detection			
	1	0		Both-edge detection			
		1					
	0	0	1	Falling edge detection	Detection enable	Detection enable	Detection enable
		1		Rising edge detection			
	1	0		Both-edge detection			
		1					
1	0	0	0	Low level detection	Detection enable	Detection disable (0x00)	Detection disable
		1		High level detection			
	1	0		Low level detection			
		1		High level detection			
	0	0	1	Low level detection	Detection enable	Detection enable	Detection enable
		1		High level detection			
	1	0		Low level detection			
		1		High level detection			

3.9.2.6 PORTF

Port F can be used as an 8-bit general-purpose input-output pin.

It can also be used as the TPIU function (TRACECLK, TRACEDATA3, TRACEDATA2, TRACEDATA1, TRACEDATA0). They all can be set on a bit basis.



Port name	Function output 2	Function output 1	Function output 2 enable	Function output 1 enable	Function input 2 enable	Function input 1 enable
PF0	Setting prohibited	TRACEDATA0	Setting prohibited	"1"	Setting prohibited	"0"
PF1	Setting prohibited	TRACEDATA1	Setting prohibited	"1"	Setting prohibited	"0"
PF2	Setting prohibited	TRACEDATA2	Setting prohibited	"1"	Setting prohibited	"0"
PF3	Setting prohibited	TRACEDATA3	Setting prohibited	"1"	Setting prohibited	"0"
PF4	Setting prohibited	TRACECLK	Setting prohibited	"1"	Setting prohibited	"0"
PF5	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
PF6	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
PF7	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose input	GPIOFDATA	GPIOFDIR	GPIOFFR1	GPIOFFR2
	*	0	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose output	GPIOFDATA	GPIOFDIR	GPIOFFR1	GPIOFFR2
	*	1	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Output	Output	Output	Output	Output	Output	Output	Output

TPIU settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
TPIU	GPIOFDATA	GPIOFDIR	GPIOFFR1	GPIOFFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
–	–	–	TRACECLK	TRACEDATA3	TRACEDATA2	TRACEDATA1	TRACEDATA0

base address = 0x4000_D000

Register Name	Address (base+)	Description
GPIOFDATA	0x0000 to 0x03FC	PortF Data Register
GPIOFDIR	0x0400	PortF Data Direction Register
GPIOFFR1	0x0424	PortF Function Register1
GPIOFFR2	0x0428	PortF Function Register2

1. GPIOFDATA (Port Data Register)

Address = (0x4000_D000)+(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7:0]	PF7:0	R/W	0y00000000	Bm7:0	Port F data register

[Explanation]

a. <PF7:0>

Data register: A register that retains data.

Refer to notes on the data register for the bit mask function.

2. GPIOFDIR (Port Data Direction Register)

Address = (0x4000_D000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PF7C to PF0C	R/W	0y00000000	Port F data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PF7C to PF0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOFFR1 (Port Function Register1)

Address = (0x4000_D000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:5]	Reserved	R/W	0y000	Read zero. Write as zero.
[4:0]	PF4F1 to PF0F1	R/W	0y00000	Port F function register 1

[Explanation]

a. <PF4F1 to PF0F1>

Function register 1: Control register 1 for switching the function

4. GPIOFFR2 (Port Function Register2)

Address = (0x4000_D000) +(0x0428)

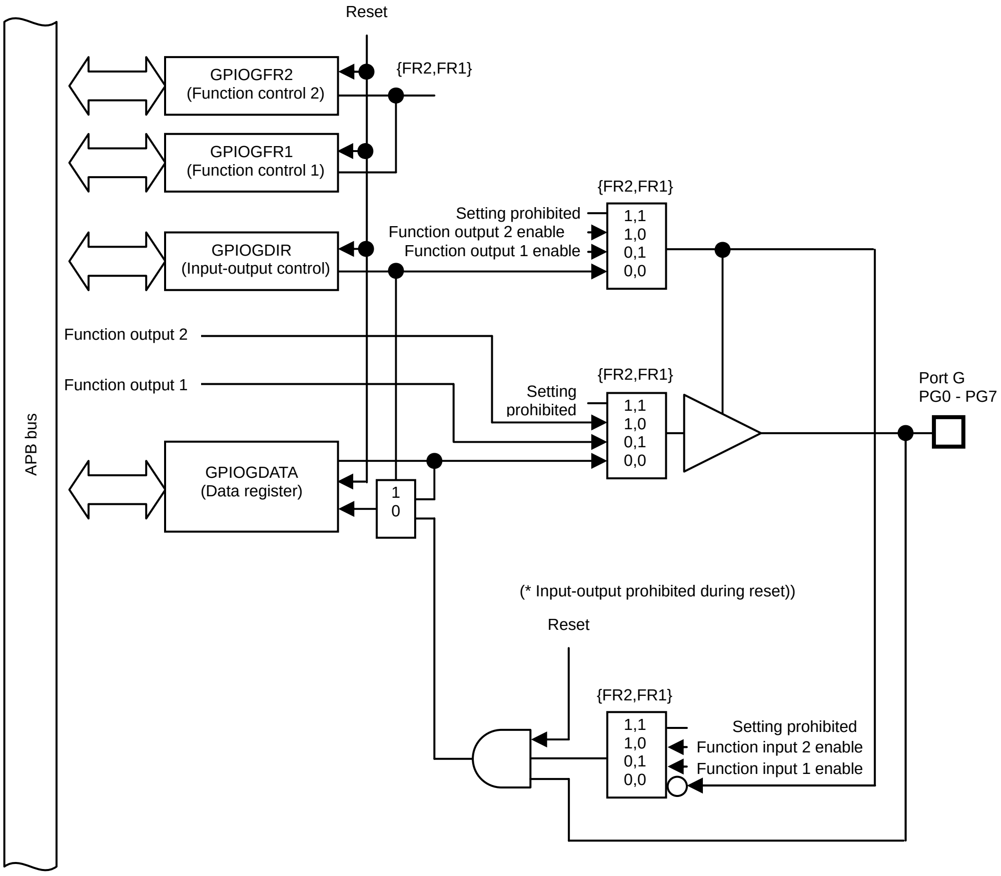
Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	Reserved	R/W	0y00000000	Read zero. Write as zero.

[Explanation]

3.9.2.7 PORTG

Port G can be used as an 8-bit general-purpose input-output pin.

It can also be used as the SMC function (BE1n, BE0n, CS1n, CS0n, A22, A21, A20, A19). They all can be set on a bit basis.



Port name	Function output 2	Function output 1	Function output 2 enable	Function output 1 enable	Function input 2 enable	Function input 1 enable
PG0	Setting prohibited	A19	Setting prohibited	"1"	Setting prohibited	"0"
PG1	Setting prohibited	A20	Setting prohibited	"1"	Setting prohibited	"0"
PG2	Setting prohibited	A21	Setting prohibited	"1"	Setting prohibited	"0"
PG3	Setting prohibited	A22	Setting prohibited	"1"	Setting prohibited	"0"
PG4	Setting prohibited	CS0n	Setting prohibited	"1"	Setting prohibited	"0"
PG5	Setting prohibited	CS1n	Setting prohibited	"1"	Setting prohibited	"0"
PG6	Setting prohibited	BE0n	Setting prohibited	"1"	Setting prohibited	"0"
PG7	Setting prohibited	BE1n	Setting prohibited	"1"	Setting prohibited	"0"

* : don't care

General-purpose input settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose input	GPIOGDATA	GPIOGDIR	GPIOGFR1	GPIOGFR2
	*	0	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
General-purpose output	GPIOGDATA	GPIOGDIR	GPIOGFR1	GPIOGFR2
	*	1	0	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Output	Output	Output	Output	Output	Output	Output	Output

SMC settings

Purpose	Data value	Input-output change	Function change 1	Function change 2
SMC	GPIOGDATA	GPIOGDIR	GPIOGFR1	GPIOGFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BE1n	BE0n	CS1n	CS0n	A22	A21	A20	A19

base address = 0x4000_E000

Register Name	Address (base+)	Description
GPIOGDATA	0x0000 to 0x03FC	PortG Data Register
GPIOGDIR	0x0400	PortG Data Direction Register
GPIOGFR1	0x0424	PortG Function Register1
GPIOGFR2	0x0428	PortG Function Register2

1. GPIOGDATA (Port Data Register)

Address = (0x4000_E000) +(0x0000~0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read undefined. Write as zero.
[7:0]	PG7:0	R/W	0y00000000	Bm7:0	Port G data register

[Explanation]

a. <PG7:0>

Data register: A register that retains data.

Refer to notes on the data register for the bit mask function.

2. GPIOGDIR (Port Data Direction Register)

Address = (0x4000_E000) +(0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PG7C to PG0C	R/W	0y00000000	Port G data direction register (each bit) 0y0: Input 0y1: Output

[Explanation]

a. <PG7C to PG0C>

Data direction register: A register for controlling input and output of each pin when the port is used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOGFR1 (Port Function Register1)

Address = (0x4000_E000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	PG7F1 to PG0F1	R/W	0y00000000	Port G function register 1

[Explanation]

a. <PG7F1 to PG0F1>

Function register 1: Control register 1 for switching the function

4. GPIOGFR2 (Port Function Register2)

Address = (0x4000_E000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	Reserved	R/W	0y00000000	Read zero. Write as zero.

[Explanation]

3.9.3 Notes

<Procedure for using the interrupt function>

Interrupts can be detected in various modes for edge and level. When using the interrupt detection function (setting the concerned bits of GPIOEIE to "1") and switching modes (changing the concerned bits of GPIOEIS, GPIOEIBE, and GPIOEIEV), follow the procedure below:

0. Disable the NVIC INTx interrupts.
1. Disable the interrupts of concerned bits. GPIOEIE=0
2. Change the mode of concerned bits. Set GPIOEIS, GPIOEIBE, and GPIOEIEV.
3. Clear the interrupts of concerned bits. GPIOEIC=1
4. Enable the interrupts of concerned bits. GPIOnIE=1
5. Enable NVIC.

3.10 Memory Controller (Static Memory Controller)

This LSI contains an SMC (Static Memory Controller) for controlling asynchronous external memory (NOR flash memory, SRAM, etc.).

3.10.1 Function Overview

Table 3.10.1 shows the features of the SMC.

Table 3.10.1 Features of SMC

Feature	Chip select 0/1
Support memory	External asynchronous memory (NOR flash memory, SRAM, etc.) Only separate buses are supported.
Data bus width	Only 16-bit data bus width is supported.
Access space	Up to 16-MB of access space is supported. Two spaces are supported by chip select.
Timing adjustment	AC timing can be adjusted by registers.
Clock	The clock for generating external control pins can be set with the clock controller CG_PLLCTRL6<FSMCDIV>. 1/2 × SMCCLK and 1/4 × SMCCLK can be selected.
External control pin	OEn, WEn, CS0n, CS1n, BE0n, BE1n D15-D0, A0-A18, A19-A22 (Used together with PG0-PG3)

3.10.2 Explanation of the Register

(1) SMC registers

Only 32-bit accesses are supported for register read and write.

Table 3.10.1 shows a list of registers.

Table 3.10.2 SFR list

base address = 0x4000_4000

Register Name	Address (base+)	Type	Reset value	Description
–	0x0000	RO	Undefined	Reserved
	0x0004	RO	Undefined	Reserved
–	0x0008	WO	Undefined	Reserved
–	0x000C	WO	Undefined	Reserved
smc_direct_cmd	0x0010	WO	–	SMC Direct Command Register
smc_set_cycles	0x0014	WO	–	SMC Set Cycles Register
smc_set_opmode	0x0018	WO	–	SMC Set Opmode Register
–	0x0020	R/W	Undefined	eserved
smc_sram_cycles0_0 smc_sram_cycles0_1	0x0100 0x0120	RO	0x0002B3CC	SMC SRAM Cycles Registers <0-1>
–	0x0140	RO	Undefined	Reserved
–	0x0160	RO	Undefined	Reserved
smc_opmode0_0 smc_opmode0_1	0x0104 0x0124	RO	0x00000802	SMC Opmode Registers <0-3>
–	0x0144	RO	Undefined	Reserved
–	0x0164	RO	Undefined	Reserved
–	0x0200	RO	Undefined	Reserved
–	0x0204	WO	Undefined	Reserved
–	0x0E00	R/W	Undefined	Reserved
–	0x0E04	RO	Undefined	Reserved
–	0x0E08	WO	Undefined	Reserved
–	0x0FE0-0x0FEC	RO	Undefined	Reserved
–	0x0FF0-0x0FFC	RO	Undefined	Reserved

1. smc_direct_cmd (SMC Direct Command Register)

Address = (0x4000_4000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:26]	–	–	Undefined	Write as zero.
[25:23]	chip_select	WO	–	CS selection: 0y000 = CS0 0y001 = CS1 0y010 = Reserved 0y011 = Reserved 0y100-0y111 = Reserved
[22:21]	cmd_type	WO	–	Current command: 0y00 = Reserved 0y01 = Reserved 0y10 = UpdateRegs 0y11 = Reserved
[20]	Reservred	WO	Undefined	Write as zero.
[19:0]	Reserved	WO	Undefined	Write as zero.

Executing <UpdateRegs> in the smc_direct_cmd register enables the values set in the set_opmode and set_cycles registers.

[Explanation]

a. <chip_select>
Select a CS to update its setting.

b. <cmd_type>
A transfer command can be selected (register update command, etc.).

2. smc_set_cycles (SMC Set Cycles Register)

Address = (0x4000_4000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:23]	–	–	Undefined	Write as zero.
[22:20]	Reserved	WO	–	Write as zero.
[19:17]	Set_t5	WO	–	Set value of tTR: 0y000 – 0y111
[16:14]	Set_t4	WO	–	Set value of tPC: 0y000 – 0y111
[13:11]	Set_t3	WO	–	Set value of tWP: 0y000 – 0y111
[10:8]	Set_t2	WO	–	Set value of tCEOE: 0y000 – 0y111
[7:4]	Set_t1	WO	–	Set value of tWC: 0y0000 – 0y1111
[3:0]	Set_t0	WO	–	Set value of tRC: 0y0000 – 0y1111

This register is the register for adjusting the access cycle of StaticMemory.

Set it according to the A.C. required by memory.

To enable the setting, you need to execute <UpdateRegs> in the smc_direct_cmd register.

[Explanation]

a. <Set_t5>

Can set the value of t_{TR}.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

b. <Set_t4>

Can set the value of t_{PC}.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

c. <Set_t3>

Can set the value of t_{WP}.

0y000: SMCCLK× 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

d. <Set_t2>

Can set the value of $t_{CEOE.}$

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

e. <Set_t1>

Can set the value of $t_{WC.}$

0y000: SMCCLK × 15 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 15 clocks

f. <Set_t0>

Can set the value of t_{RC}.

0y000: SMCCLK × 15 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 15 clocks

3. smc_set_opmode (SMC Set Opmode Register)

Address = (0x4000_4000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Write as zero.
[15:13]	set_burst_align	WO	–	Memory burst boundary division setting: 0y000 = Boundary of an arbitrary address can be crossed. 0y001 = Division at the 32-beat burst boundary 0y010 = Division at the 64-beat burst boundary 0y011 = Division at the 128-beat burst boundary 0y100 = Division at the 256-beat burst boundary other = Reserved
[12]	set_bls	WO	–	bls timing setting for write: 0y0 = chip select timing 0y1 = smc_we_n_0 timing
[11]	Reserved	WO	Undefined	Write as zero.
[10]	Reserved	WO	Undefined	Write as zero.
[9:7]	set_wr_bl	WO	–	Write burst length: 0y000 = 1 beat 0y001 = 4 beats 0y010~0y111 = Reserved
[6]	Reserved	WO	Undefined	Write as zero.
[5:3]	set_rd_bl	WO	–	Read burst length: 0y000 = 1 beat 0y001 = 4 beats 0y010–0y111 = Reserved
[2]	Reserved	WO	Undefined	Write as zero.
[1:0]	set_mw	WO	–	Set value of memory data bus width: 0y00 = Reserved 0y01 = 16 bits 0y10 = Reserved 0y11 = Reserved

To enable the settings of the smc_set_opmode register, you need to execute <UpdateRegs> in the smc_direct_cmd register.

[Explanation]

a. <set_burst_align>

For asynchronous transfers, when set_rd_sync = 0, PL241 always aligns read bursts to the memory burst boundary.

When set_wr_sync = 0, PL241 always aligns write bursts to the memory burst boundary.

b. <set_bls>

Controls the timing of bls (byte-lane strobe) output.

Since the set_bls=1 setting is for 8-bit wide memory, connect bls[3:0] to /WE.

c. <set_wr_bl>

Can set the burst length for memory write.

d. <set_rd_bl>

Can set the burst length for memory read.

e. <set_mw>

Can set the data bus width of memory.

4. smc_sram_cycles0_0 (SMC SRAM Cycles Registers 0 <0>)

Address = (0x4000_4000) + (0x0100)

Bit	Bit Symbol	Type	Reset Value	Description
[31:20]	–	–	Undefined	Read undefined. Write as zero.
[19:17]	t_tr	RO	0y001	Turn-around time for the SRAM chip setting 0y000 – 0y111
[16:14]	t_pc	RO	0y010	Page cycle time: 0y000 – 0y111
[13:11]	t_wp	RO	0y110	Delay time of smc_we_n_0: 0y000 – 0y111
[10:8]	t_ceoe	RO	0y011	Delay time of smc_oe_n_0: 0y000 – 0y111
[7:4]	t_wc	RO	0y1100	Write cycle time: 0y0000 – 0y1111
[3:0]	t_rc	RO	0y1100	Read cycle time: 0y0000 – 0y1111

[Explanation]

a. <t_tr>

Shows the turn-around time for the SRAM chip setting.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

b. <t_ pc>

Shows the page cycle time.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

c. <t_wp>

Shows the delay time of smc_we_n_0.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

d. <t_ceoe>

Shows the delay time of smc_oe_n_0.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

e. <t_wc>

Shows the write cycle time.

0y000: SMCCLK × 15 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 15 clocks

f. <t_rc>

Shows the read cycle time.

0y000: SMCCLK × 15 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 15 clocks

5. smc_sram_cycles0_1 (SMC SRAM Cycles Registers 0 <1>)

Address = (0x4000_4000) + (0x0120)

Bit	Bit Symbol	Type	Reset Value	Description
[31:20]	–	–	Undefined	Read undefined. Write as zero.
[19:17]	t_tr	RO	0y001	Turn-around time for the SRAM chip setting 0y000 – 0y111
[16:14]	t_pc	RO	0y010	Page cycle time: 0y000 – 0y111
[13:11]	t_wp	RO	0y110	Delay time of smc_we_n_0: 0y000 – 0y111
[10:8]	t_ceoe	RO	0y011	Delay time of smc_oe_n_0: 0y000 – 0y111
[7:4]	t_wc	RO	0y1100	Write cycle time: 0y0000 – 0y1111
[3:0]	t_rc	RO	0y1100	Read cycle time: 0y0000 – 0y1111

[Explanation]

a. <t_tr>

Shows the turn-around time for the SRAM chip setting.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

b. <t_ pc>

Shows the page cycle time.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

c. <t_wp>

Shows the delay time of smc_we_n_0.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

d. <t_ceoe>

Shows the delay time of smc_oe_n_0.

0y000: SMCCLK × 7 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 7 clocks

e. <t_wc>

Shows the write cycle time.

0y000: SMCCLK × 15 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 15 clocks

f. <t_rc>

Shows the read cycle time.

0y000: SMCCLK × 15 clocks

0y001 - 0y111: SMCCLK × 1 clock - SMCCLK × 15 clocks

- Example of tRC tCEOE register setting

tRC=3, tCEOE=1

smc_set_cycles = 0x0002B1C3

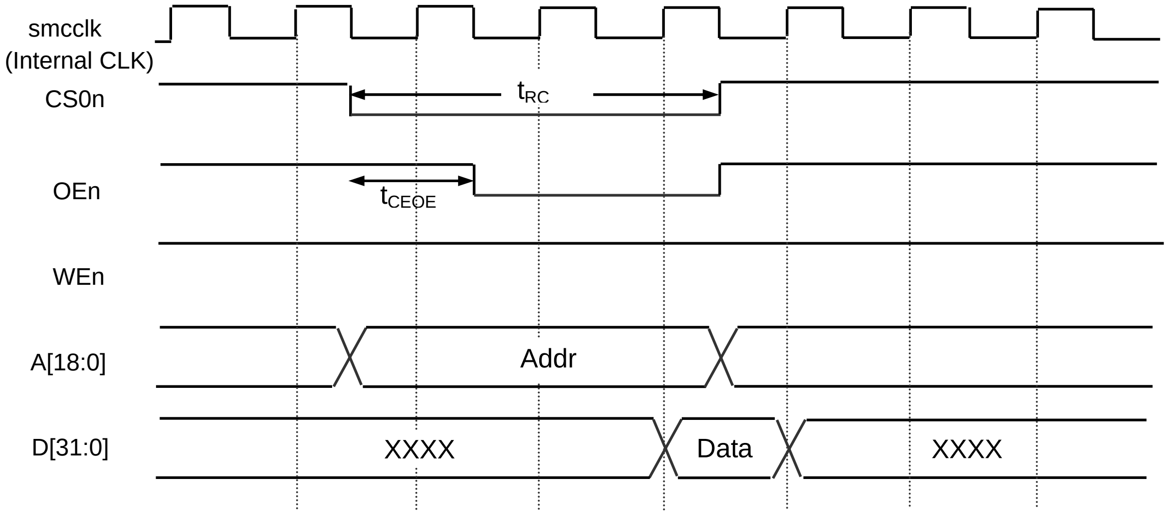


Figure 3.10.1 Asynchronous read

- Example of tWC tWP register setting

tWC=4, tWP=2

smc_set_cycles = 0x0002934C

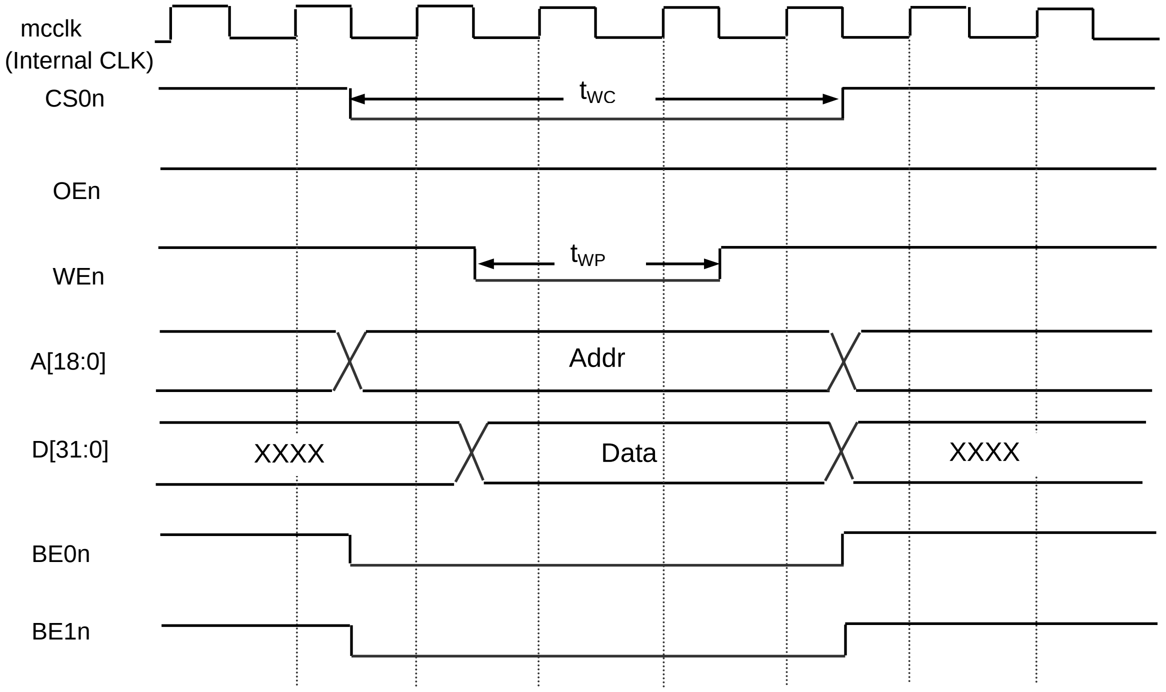


Figure 3.10.2 Asynchronous write

■ Example of tRC, tCEOE, tPC register setting

tRC=3, tCEOE=2, tPC=1

smc_set_cycles = 0x000272C3

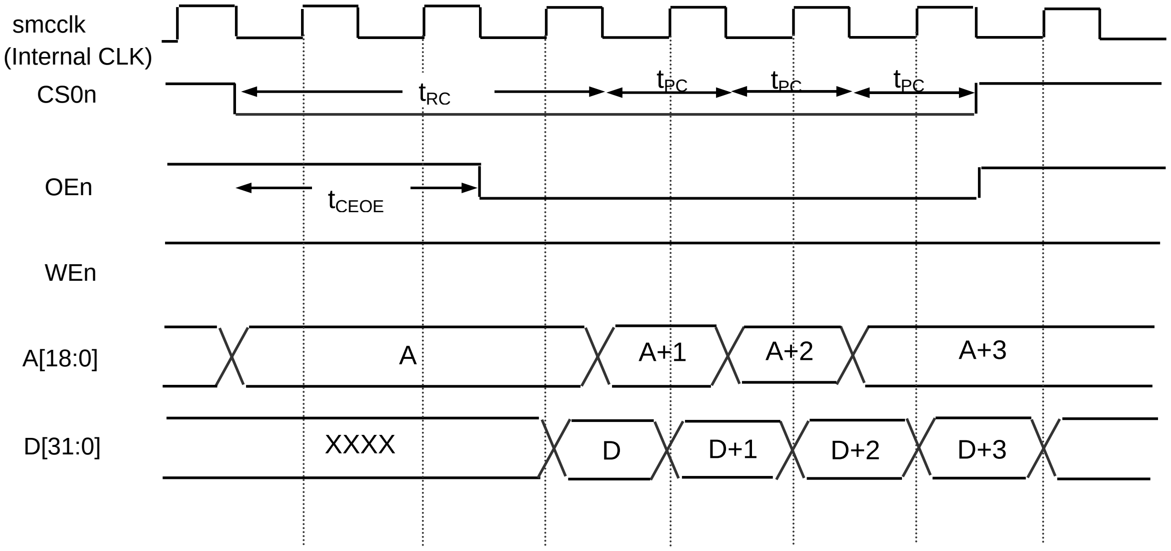


Figure 3.10.3 Asynchronous page read

■ Example of tTR register setting

tTR=1

smc_set_cycles = 0x00029143

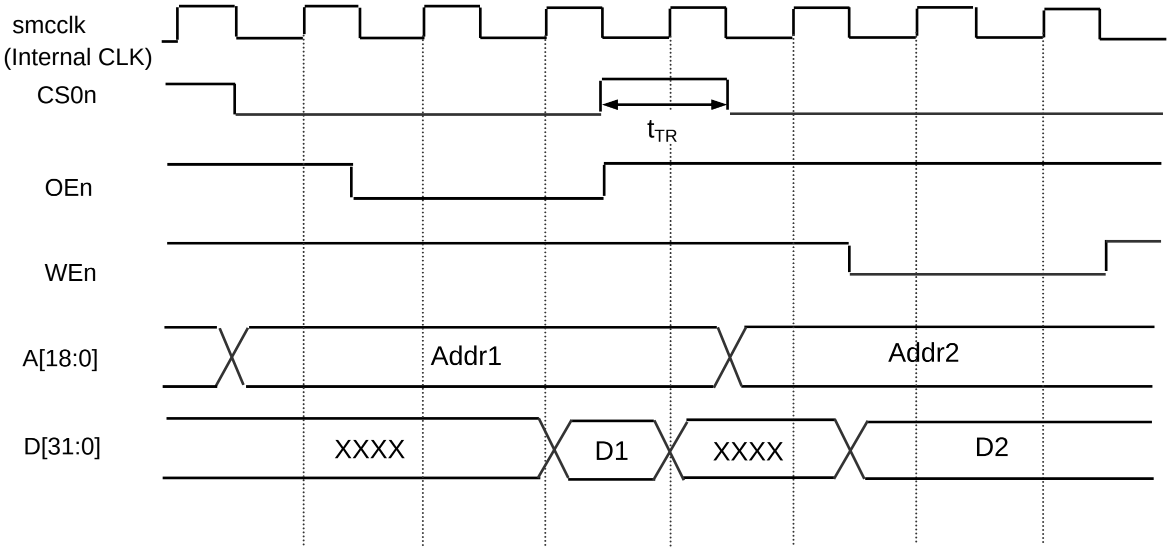


Figure 3.10.4 Asynchronous write after asynchronous read

6. smc_opmode0_0 (SMC Opmode Registers 0<0>)

Address = (0x4000_4000) + (0x0104)

Bit	Bit Symbol	Type	Reset Value	Description

[31:24]	address_match	RO	0x00(Normal)
			0x60(Boot)
				Set value of start address [31:24]
				0x00: Normal mode
				0x60: Boot mode

[23:16]	Reserved	RO	0xFF	Read as 0xFF
[15:13]	burst_align	RO	0y000	Memory burst boundary division set value 0y000 = Boundary of an arbitrary address can be crossed. 0y001 = Division at the 32-beat burst boundary 0y010 = Division at the 64-beat burst boundary 0y011 = Division at the 128-beat burst boundary 0y100 = Division at the 256-beat burst boundary other = Reserved
[12]	bls	RO	0y0	bls timing setting: 0y0 = chip select timing 0y1 = Reserved
[11]	Reserved	RO	Undefined	Read undefined.
[10]	Reserved	RO	Undefined	Read undefined.
[9:7]	wr_bl	RO	0y000	Write memory burst length: 0y000 = 1 beat 0y001 = 4 beats 0y010–0y111 = Reserved
[6]	Reserved	RO	Undefined	Read Undefined
[5:3]	rd_bl	RO	0y000	Read memory burst length: 0y000 = 1 beat 0y001 = 4 beats 0y010–0y111 = Reserved
[2]	Reserved	RO	Undefined	Read Undefined
[1:0]	mw	RO	0y10	Set value of memory data bus width: 0y00 = Reserved 0y01 = 16bits 0y10 = Reserved 0y11 = Reserved

Note) Do not access SMC areas (not used) other than set CS areas.

[Explanation]

a. <address_match>

Shows the set value of start address [31:24].

For normal mode: 0×00

For Boot mode: 0×60

b. <burst_align>

Shows the set value of memory burst boundary division.

c. <bls>

Shows the setting of the bls (byte lane strobe) timing.

d. <wr_bl>

Shows the burst length for memory write.

e. <rd_bl>

Shows the burst length for memory read.

f. <mw>

Shows the setting of the data bus width of the memory assigned to CS0.

7. smc_opmode0_1 (SMC Opmode Registers 0<1>)

Address = (0x4000_4000) + (0x0124)

Bit	Bit Symbol	Type	Reset Value	Description

[31:24]	address_match	RO	0x00(Normal)
			0x61(Boot)
				Set value of start address [31:24]
				0x00: Normal mode
				0x61: Boot mode

[23:16]	Reserved	RO	0xFF	Read as 0xFF
[15:13]	burst_align	RO	0y000	Memory burst boundary division set value: 0y000 = Boundary of an arbitrary address can be crossed. 0y001 = Division at the 32-beat burst boundary 0y010 = Division at the 64-beat burst boundary 0y011 = Division at the 128-beat burst boundary 0y100 = Division at the 256-beat burst boundary other = Reserved
[12]	bls	RO	0y0	bls timing setting: 0y0 = chip select timing 0y1 = Reserved
[11]	Reserved	RO	Undefined	Read undefined.
[10]	Reserved	RO	Undefined	Read undefined.
[9:7]	wr_bl	RO	0y000	Write memory burst length: 0y000 = 1 beat 0y001 = 4 beats 0y010–0y111 = Reserved
[6]	Reserved	RO	Undefined	Read undefined.
[5:3]	rd_bl	RO	0y000	Read memory burst length: 0y000 = 1 beat 0y001 = 4 beats 0y010–0y111 = Reserved
[2]	Reserved	RO	Undefined	Read undefined.
[1:0]	mw	RO	0y10	Set value of memory data bus width: 0y00 = Reserved 0y01 = 16 bits 0y10 = Reserved 0y11 = Reserved

Note) Do not access other SMC areas (not used) than set CS areas.

[Explanation]

a. <address_match>

Shows the set value of start address [31:24].

For normal mode: 0×00

For Boot mode: 0×61

b. <burst_align>

Shows the set value of memory burst boundary division.

c. <bls>

Shows the setting of the bls (byte lane strobe) timing.

d. <wr_bl>

Shows the burst length for memory write.

e. <rd_bl>

Shows the burst length for memory read.

f. <mw>

Shows the setting of the data bus width of the memory assigned to CS1.

3.10.3 Example of External Memory Connection

Figure 3.10.5 shows an example of connection with an external 16-bit SRAM and a 16-bit NOR-flash memory.

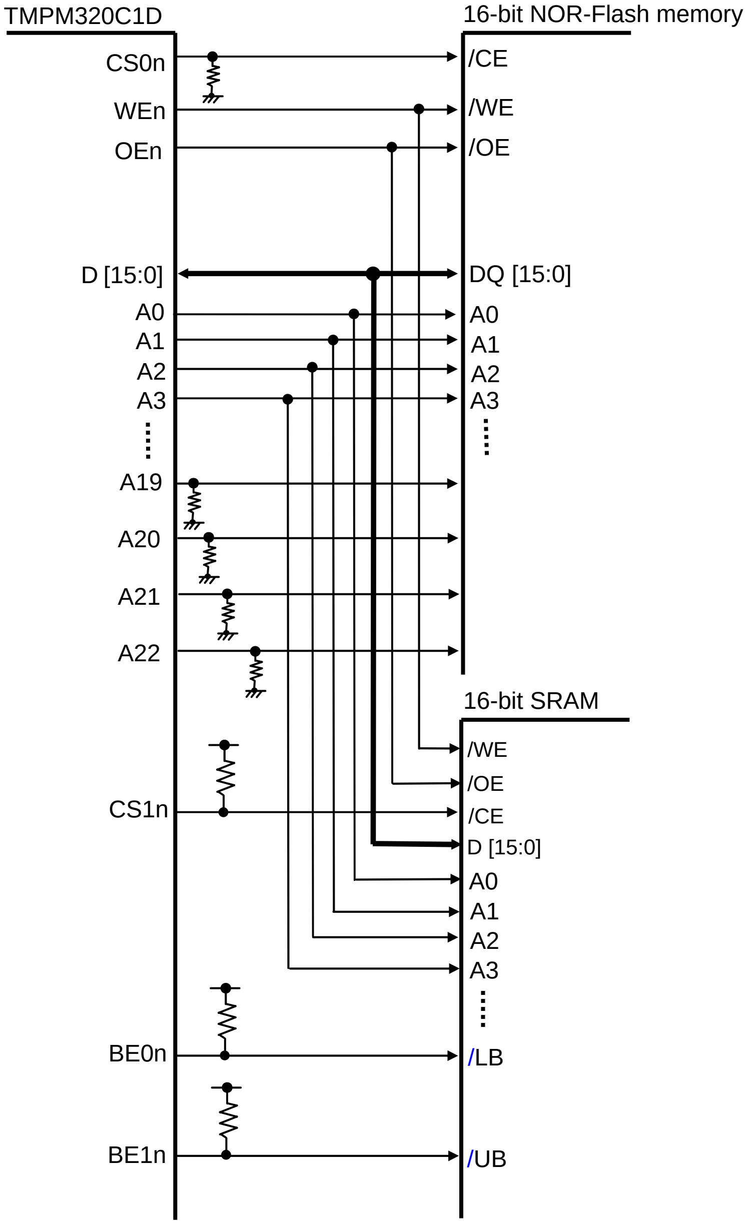


Figure 3.10.5 Example of connection with external 16-bit SRAM and NOR-flash memory

3.11 eDRAM Controller

This LSI has a built-in circuit that controls an embedded DRAM (hereinafter, refer to as eDRAM controller). The following explains the features of the eDRAM controller.

(a) Built-in eDRAM has the memory size of 1 Mbytes.

The eDRAM size built into this LSI is shown below.

Number of banks:	1 bank
Row address:	2048 rows
Column address:	16 columns
Data bus width:	256 bits

(b) The AC parameter of the eDRAM that changes depending on the clock frequency can be changed by using the register settings. (The initial value is set with f_{HCLK} = 144 MHz.)

(c) Changing the CKE control register can reduce consumption current while data is held.

(d) Two types of software resets (controller + eDRAM or eDRAM only) are available.

(e) Distributed refresh or burst refresh can be selected by using the register.

3.11.1 Function Overview

This controller has 2 types of I/F modules: the I/F modules for the AHB bus (hereinafter, referred to as AHB I/F) and the eDRAM direct I/F module (hereinafter, referred to as eDRAM I/F).

The AHB I/F performs handshaking with the AMBA_AHB bus in order to control the write/read operations to/from the eDRAM via the eDRAM I/F.

The eDRAM I/F controls the addresses to the eDRAM macros and the commands, and inputs/outputs data.

Figure 3.11.1 shows a simple block diagram of the eDRAM controller.

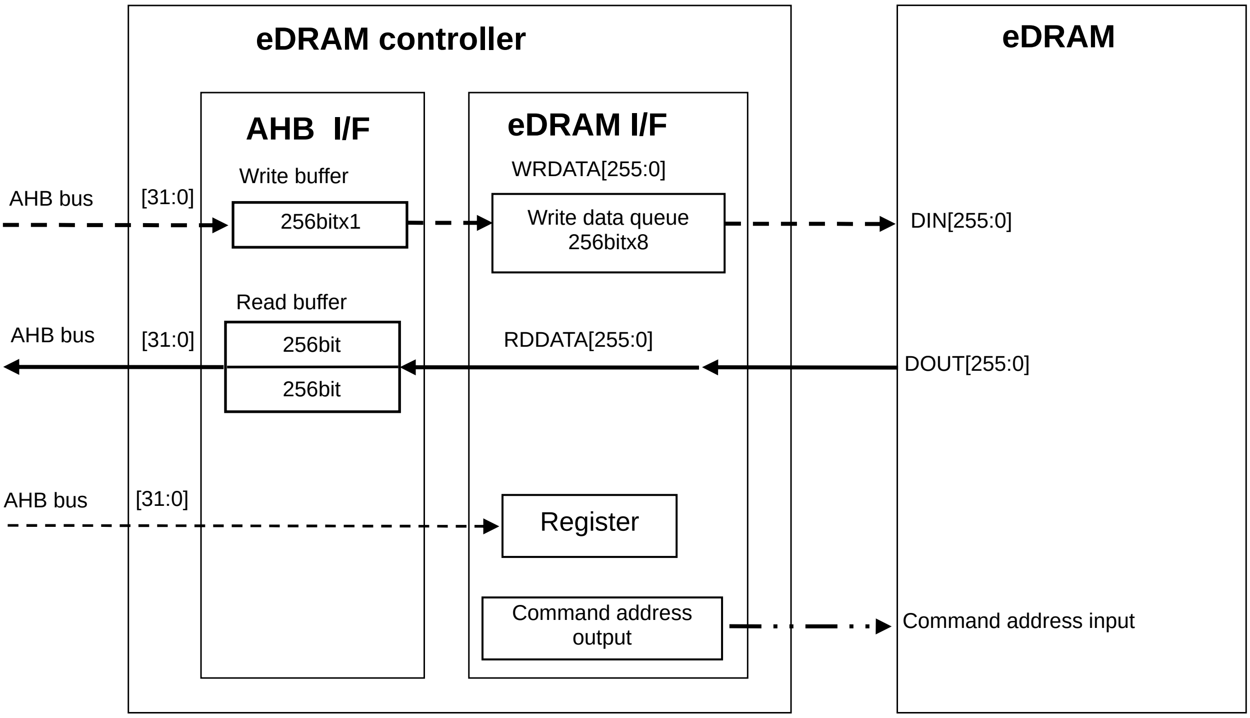


Figure 3.11.1 eDRAM controller block diagram

3.11.2 Operation Description

(1) AHB I/F

The AHB I/F performs handshaking with the AHB bus and the eDRAM I/F.

Writing:

- The data transferred from the AHB bus is stored in the write buffer (256 bits) (32 bits → 256 bits).
- The write data mask signal to the eDRAM is generated (controlled in units of bytes).

Reading:

- The data transferred from the eDRAM I/F is stored in the read buffer (256 bits).
- Data is transferred from the read buffer to the AHB bus (256 bits → 32 bits).

(1-1) Write buffer

The write buffer stores the data from the AHB bus (32 bits).

All of a single set of transaction data is stored and then transferred to the eDRAM I/F.

When the transaction data at the address that is not aligned to the write buffer is stored, the buffer may overflow depending on the transfer start address. When the buffer overflows, the subsequent data is not stored until the data that was stored in the buffer first becomes empty.

For effective data transfer, access data so that 1 transaction is fit into the write buffer.

Because the controller has a data queue, data of up to 8 transactions can be stored.

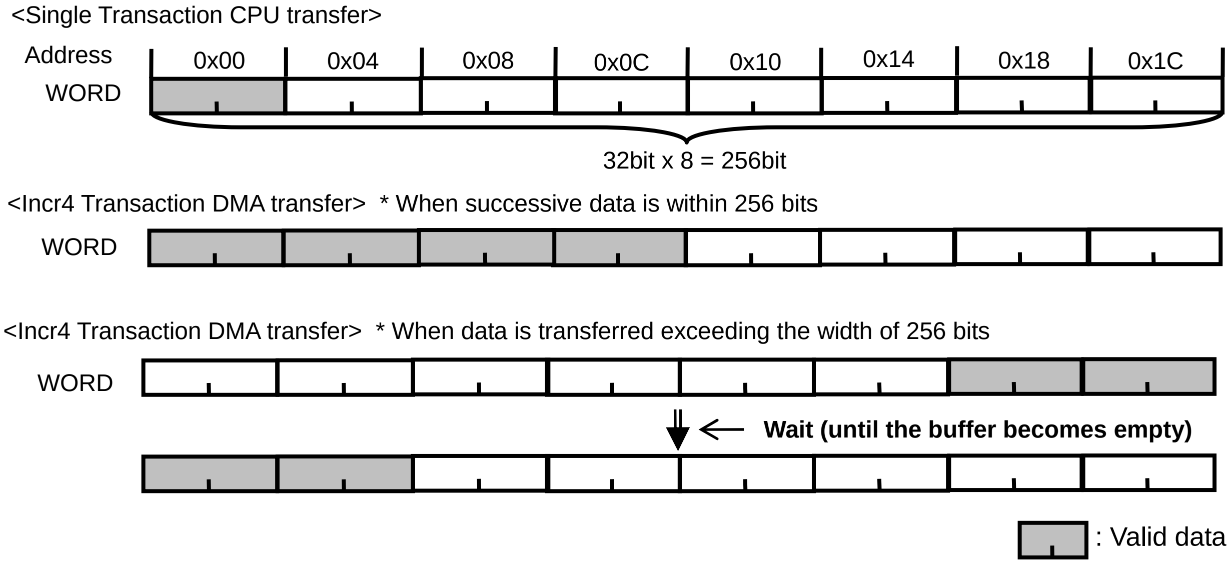

<Single Transaction CPU transfer>
Address
WORD
0x00
0x04
0x08
0x0C
0x10
0x14
0x18
0x1C
32bit x 8 = 256bit
<Incr4 Transaction DMA transfer> * When successive data is within 256 bits
WORD
<Incr4 Transaction DMA transfer> * When data is transferred exceeding the width of 256 bits
WORD
Wait (until the buffer becomes empty)
: Valid data

Figure 3.11.2 Image of the write buffer during data write operation

(1-2) Read buffer

The read buffer stores the data (256 bits) from the eDRAM I/F.

All data for a single burst transfer is stored and then transferred to the AHB bus (32 bits).

In addition, the read buffer has a double buffer structure which can store transfer data of up to 512 bits.

This enables data transfer without generating a wait even when data is transferred with an unaligned address.

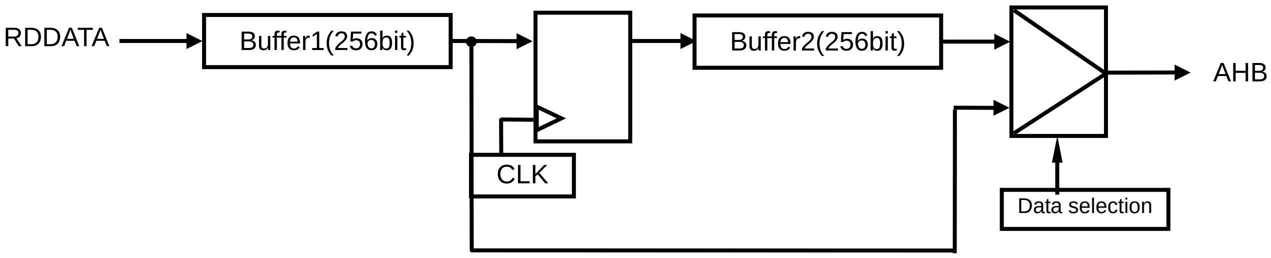


Figure 3.11.3 Image of the read buffer during data read operation

(2) eDRAM I/F

The eDRAM I/F inputs/outputs data to/from the eDRAM and controls the commands.

(2-1) Reset control

Inputting an external pin reset signal initializes the eDRAM I/F and also instructs initialization of the eDRAM.

In addition, the soft reset register 1 (R_MDSRST1) can be used to initialize the controller + eDRAM, and the soft reset register 2 (R_MDSRST2) can be used to instruct initialization of the eDRAM.

(2-2) Reset sequence

When a reset sequence is requested, it is forcedly executed regardless of the existence of the command. When the reset sequence is complete, the eDRAM starts normal operation.

* Conditions in which a reset sequence is requested:

1. When a software reset (SRST of the eDRAM I/F and the CPU) is executed
2. When a hardware reset (WDT, external) is executed

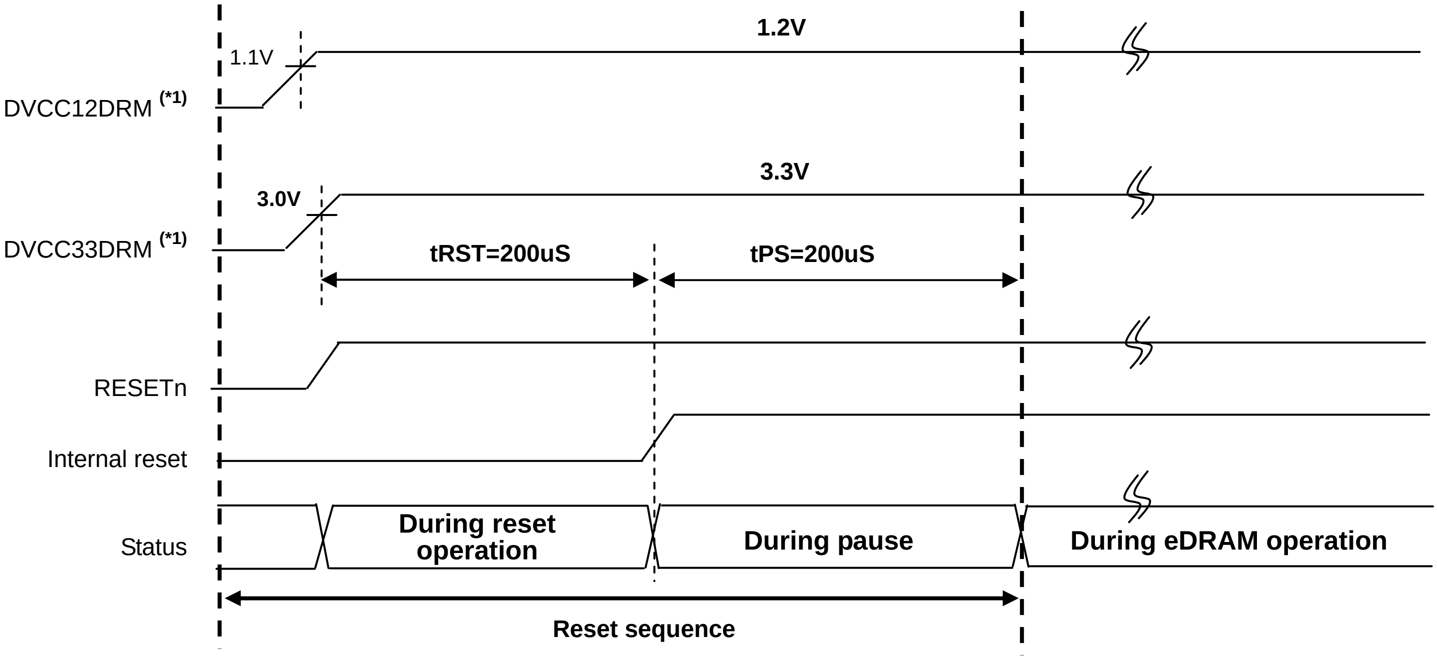


Figure 3.11.4 Waveforms during reset sequence

(*1): Apply the **DVCC33DRM** and **DVCC12DRM** power at the same time, or apply the **DVCC12DRM** power first.

(2-3) Auto refresh

Auto refresh starts automatically when the power-up sequence is complete. Also, burst refresh or distributed refresh can be selected by using the register (R_MDREFMODE).

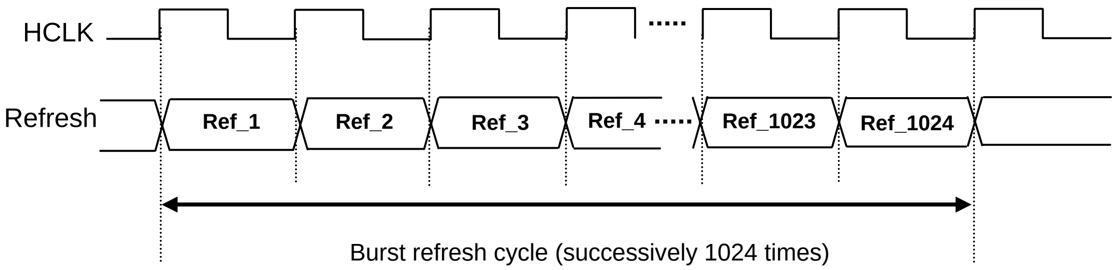
Execute refresh after all the commands that are currently being executed are complete.

Once the burst refresh starts, it occupies the eDRAM until the specified number of refreshes (1024 times) has completed. Therefore, any remaining command will not be executed until the burst refresh is complete.

* Notes on switching the refresh modes
Do not switch between the burst refresh and the distributed refresh dynamically.
Data may be lost because the refresh time restrictions are not satisfied.
To switch the modes, reset the eDRAM macro first.

- Burst refresh

When the burst refresh is selected, the refresh will be executed successively for 1024 times at intervals of 2 ms.



- Distributed refresh

When the distributed refresh is selected, the refresh will be distributed and executed 1024 times within 2 ms.

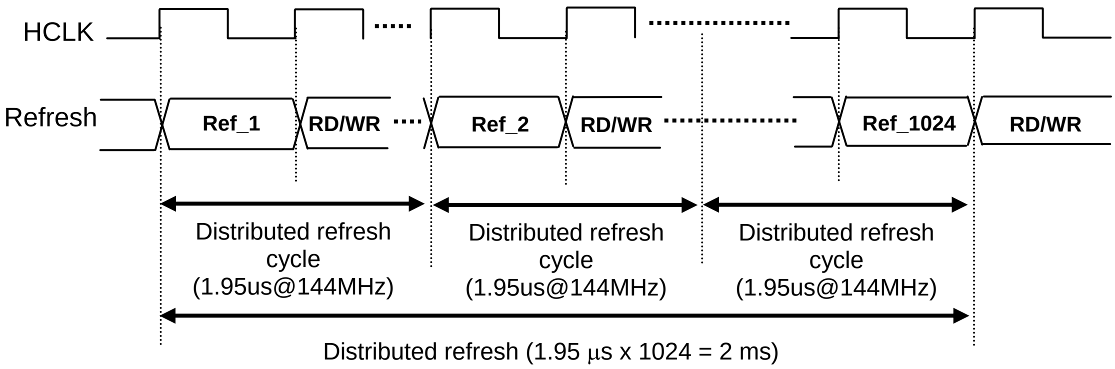

HCLK
Refresh
Ref_1
RD/WR
Ref_2
RD/WR
Ref_1024
RD/WR
Distributed refresh cycle (1.95us@144MHz)
Distributed refresh cycle (1.95us@144MHz)
Distributed refresh cycle (1.95us@144MHz)
Distributed refresh (1.95 μs x 1024 = 2 ms)

Figure 3.11.5 Waveforms during burst refresh/distributed refresh operation

(2-4) Clock enable (CKE) control

The CKE control register (R_MDCKE) can be used to control the clocks in the eDRAM to reduce power consumption.

- Processing sequence when Clock Disable is set
  When Clock Disable is requested (R_MDCKE = 0), it is not executed immediately. The processing sequence is shown below.

  1) When a command or the auto refresh is being executed, wait until the operation is complete.
     - When a command is being executed:
       Wait for only the command that is currently performing processing to access to the eDRAM to complete. (The commands already stored in the command queue will not be lost.)
     - When the auto refresh is being executed:
       When a request is received during the burst refresh operation, wait until the burst refresh is complete (1024 times).
       When the distributed refresh is being executed, wait until the current refresh is complete.

  2) Auto refresh timing in clock disable mode

     Because the auto refresh operation is required even in the clock disable mode, the clock is temporarily enabled at the timing of the auto refresh to perform the following auto refresh operation.

In distribute refresh mode:	Execute the refresh once.
In burst refresh mode:	Execute the refresh successively 1024 times.

  3) Command processing in clock disable mode

     Command requests are stored in the command queue even in the clock disable mode.

     The received commands start processing after clock disable is cancelled (R_MDCKE = 1).

(2-5) Register control

A register used to change the AC timing parameter for the eDRAM is provided.

Refer to the description in 3.11.3 “Details of the registers” for details.

3.11.3 Details of the registers

The register list is shown below.

Table 3.11.1 Register list

base address = 0x4000_1000

Address (base+)	Name	R/W	Width	Reset Value	Description
0x000	R_MDRAC	R/W	24bit	0x853331(*4)	Timing adjustment register 1
0x004	R_MDWAC	R/W	28bit	0x8532331(*4)	Timing adjustment register 2
0x008	R_MDREFAC	R/W	25bit	0x0011C9(*4)	Timing adjustment register 3
0x00C	R_MDRSTAC	R/W	16bit	0x7140(*4)	Timing adjustment register 4
0x010	R_MDTRASMAX	R/W	11bit	0x5DC(*4)	Timing adjustment register 5
0x014 (*2)	R_MDREFMODE	R/W	2bit	0x0	Auto refresh mode setting
0x018 (*2)	R_MDRL	R/W	2bit	0x3	Timing adjustment register 6
0x01C	R_MDSTART	R/W	1bit	0x0	Controller START instruction
0x020 (*3)	R_MDSRST1	R/W	1bit	0x1	Controller soft reset
0x024	R_MDSRST2	R/W	1bit	0x1	Macro soft reset
0x028	R_MDSTATUS1	R	2bit	0x3	Controller status
0x02C	R_MDSTATUS2	R	2bit	0x3	Macro status
0x030	R_MDCKE	R/W	1bit	0x1	Macro CKE control
0x034	–	–	1bit	0x1	reserved

(*2) Cannot be changed during operation. To change the mode, reset the eDRAM macro.

(*3) When the controller soft reset (R_MDSRST1) is executed, the contents in all the registers are initialized.

(*4) The initial register values are the AC set values during a frequency of 144 MHz.

Table 3.11.2 AC specifications

Symbol	Item		Min	Max	Equation	Unit
tCYC	f_{HCLK} Period (= T)		6.9	83.3	–	ns
tRC	Active-Active/Refresh command interval		48	–	R_MDRAC<tRC> × T R_MDWAC<tRC> × T	ns
tRC(REF)	Refresh – Active/Refresh		60	–	R_MDREFAC<tRC(REF)> × T	
tRAS	Active – Precharge command interval		30	10000	R_MDRAC<tRAS> × T R_MDWAC<tRAS> × T R_MDTRASMAX<tRAS(max)> × T (The <tRAS(max)> value is at the max. spec.)	
tRCD	Active – Read / Write command interval		18	–	R_MDRAC<tRCD> × T R_MDWAC<tRCD> × T	
tCCD	Read – Read/Write – Write command interval		1	–	–	Cycle
tRP	Precharge – Active/Refresh command interval		18	–	R_MDRAC<tRP> × T R_MDWAC<tRP> × T	ns
tRRD	Active(a) – Active(b) command interval		18	–	R_MDRAC<tRRD> × T R_MDWAC<tRRD> × T	
tWR	Write – Precharge command interval		12	–	R_MDWAC<tWR> × T	
tCK	CLK cycle time	RL=1	18	–	R_MDRL<tRL> × T	
		RL=2	9	–		
		RL=3	6	–		
tCH	CLK pulse width (H)		1.5	–	–	
tCL	CLK pulse width (L)		1.5	–	–	
tREF (*4)	Refresh cycle		–	2	–	2 ms/ 1024 cycles

Table 3.11.3 shows the required number of cycles to satisfy the AC restrictions in Table 3.11.2.
The following table also shows the calculated values of “cycle x period (ns)” at 144 MHz.

Table 3.11.3 Example of register settings at 144 MHz and 12 MHz

Item		Read (= R_MDRAC setting value)						Write (= R_MDWAC setting value)							Auto Refresh
A.C SPEC		tRC	tRAS	tRCD	tRP	tRRD	tCCD	tRC	tRAS	tRCD	tWR	tRP	tRRD	tCCD	tRC(REF)
		48ns	30ns	18ns	18ns	18ns	1cyc	48ns	30ns	18ns	12ns	18ns	18ns	1cyc	60ns
f(MHz)	tCK(ns)														
144	6.9	8	5	3	3	3	1	8	5	3	2	3	3	1	9
Calculation value (ns)		55.2	34.5	20.7	20.7	20.7	6.9	55.2	34.5	20.7	13.8	20.7	20.7	6.9	62.1
12	83.3	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Calculation value (ns)		83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3

1. Timing adjustment register 1 (R_MDRAC)

Address = (0x4000_1000) + (0x000)

Unit: Cycle (hexadecimal)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tMDRAC	RW	0x00853331	f_{HCLK} = Setting value at 144 MHz: 0x00853331 f_{HCLK} = Setting value at 12 MHz: 0x00111111

This register is used to set AC parameters during eDRAM Read.

Set the number of cycles for the expected AC parameter here.

2. Timing adjustment register 2 (R_MDWAC)

Address = (0x4000_1000) + (0x004)

Unit: Cycle (hexadecimal)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tMDWAC	RW	0x08532331	f_{HCLK} = Setting value at 144 MHz: 0x08532331 f_{HCLK} = Setting value at 12 MHz: 0x01111111

This register is used to set AC parameters during eDRAM Write.

Set the number of cycles for the expected AC parameter here.

3. Timing adjustment register 3 (R_MDREFAC)

Address = (0x4000_1000) + (0x008)

Unit: Cycle (hexadecimal)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tMDREFAC	RW	0x000011C9	f_{HCLK} = Setting value at 144 MHz: 0x000011C9 f_{HCLK} = Setting value at 12 MHz: 0x00000171

Note 1) Do not set a value smaller than tRC(REF) for tREF.

Note 2) Do not set 10 or fewer cycles for tREF.

This register is used to set AC parameters during eDRAM Refresh.

Set the number of cycles for the expected AC parameter here.

4. Timing adjustment register 4 (R_MDRSTAC)

Address = (0x4000_1000) + (0x00C)

Unit: Cycle (hexadecimal)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tMDRSTAC	RW	0x00007140	f_{HCLK} = Setting value at 144 MHz: 0x00007140 f_{HCLK} = Setting value at 12 MHz: 0x00000961

This register is used to set AC parameters during eDRAM Reset.

Because a counter is shared between the reset hold period (tRST) and the stabilization period after reset cancellation (tPS), the restrictions on both periods are applied for the setting values for this register.

5. Timing adjustment register 5 (R_MDTRASMAX)

Address = (0x4000_1000) + (0x010)

Unit: Cycle (hexadecimal)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tMDRSTAC	RW	0x000005DC	f_{HCLK} = Setting value at 144 MHz: 0x000005DC f_{HCLK} = Setting value at 12 MHz: 0x00000078

This register is used to set the maximum time from the active command to the precharge command.

This is used commonly between Read and Write.

If precharge operation is not executed until the time set in this register is reached, the precharge operation is executed automatically.

6. Auto refresh mode setting register (R_MDREFMODE)

Address = (0x4000_1000) + (0x014)

Bit	Bit Symbol	Type	Reset Value	Description
[1:0]	tREFMODE	RW	0y00	0y00: Auto refresh mode 0 0y01 : Reserved 0y10: Auto refresh mode 2 0y11 : Reserved

Note) Cannot be changed during operation. To change the mode, reset the eDRAM.

This register is used to set the operation mode for auto refresh.

The refresh modes can only be changed during eDRAM Reset. To set the mode, reset the eDRAM.

[Explanation]

a. <tREFMODE>

Sets the operation mode for auto refresh.

Auto refresh mode 0

Distributed refresh is performed only with the settings for the tREF counter (refresh period counter).

Auto refresh mode 2

Burst refresh is performed only with the settings for the tREF counter (refresh period counter).

7. Timing adjustment register 6 (R_MDRL)

Address = (0x4000_1000) + (0x018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tMDRL	RW	0x00000003	f_{HCLK} = Setting value at 144 MHz: 0x00000003 f_{HCLK} = Setting value at 12 MHz: 0x00000001

Note 1) Cannot be changed during operation. To change the mode, reset the eDRAM.

Note 2) The value must be confirmed before power-on sequence.

This register is used to set the read latency value during eDRAM Read.

8. Controller start register (R_MDSTART)

Address = (0x4000_1000) + (0x01C)

Bit	Bit Symbol	Type	Reset Value	Description
[0]	tSTART	RW	0y0	0y00: Stop the controller. 0y01: Start the controller.

This register is used to control the start of the eDRAM controller.

Unless “0y1” is written in this register, it is impossible to access to eDRAM.

[Explanation]

a. <tSTART>

Sets start or stop of the eDRAM controller.

9. Controller soft reset register (R_MDSRST1)

Address = (0x4000_1000) + (0x020)

Bit	Bit Symbol	Type	Reset Value	Description
[0]	tSRST1	RW	0y0	0y0: Start controller reset. 0y1: Cancel controller reset.

Note) The values in the registers and queues are initialized. Also, “0y1” must be written in R_MDSTART to restart.

Note) The contents of the eDRAM immediately after reset is cleared are undefined.

This register is used to control reset of the controller internal section and the eDRAM.
When the reset is complete, the internal reset signal is negated.
At the same time, the eDRAM is also reset, so a reset stabilization period is required before restarting.

[Explanation]

a. <tSRST1>
Starts/stops reset of the eDRAM controller internal section and the eDRAM.

10. eDRAM soft reset register (R_MDSRST2)

Address = (0x4000_1000) + (0x024)

Bit	Bit Symbol	Type	Reset Value	Description
[0]	tSRST2	RW	0y0	0y0: Start controller reset. 0y1: Cancel controller reset.

Note) “0y1” must be written in R_MDSTART to restart.

Note) The contents of the eDRAM immediately after reset is cleared are undefined.

This register is used to control reset of the eDRAM only.
(This does not reset the controller internal section.)
When the reset is complete, the eDRAM reset signal is negated.
Also, a reset stabilization period is required before restarting.

[Explanation]

a. <tSRST2>
Starts or cancels reset of the eDRAM controller.

11. Controller status register (R_MDSTATUS1)

Address = (0x4000_1000) + (0x028)

Bit	Bit Symbol	Type	Reset Value	Description
[1:0]	tSTATUS1	R	0y11	0y00: During reset operation 0y01: Wait for start instruction (Wait for "0y1" to be written in R_MDSTART) 0y10: During operation 0y11: Disabled

This register is used to monitor the current status of the eDRAM controller.

It is possible to determine if the controller is being reset or in operation by reading this register.

[Explanation]

a. <tSATUS1>

Indicates the current status of the eDRAM controller.

12. eDRAM status register (R_MDSTATUS2)

Address = (0x4000_1000) + (0x02C)

Bit	Bit Symbol	Type	Reset Value	Description
[1:0]	tSTATUS2	R	0y11	0y00: During reset operation (during BPOR assert) 0y01: In pause after start (wait for the power-up sequence to complete) 0y10: During operation 0y11: Disabled

This register is used to monitor the current status of the eDRAM.

It is possible to determine if the eDRAM is being reset or in operation by reading this register.

[Explanation]

a. <tSATUS2>

Indicates the current status of the eDRAM.

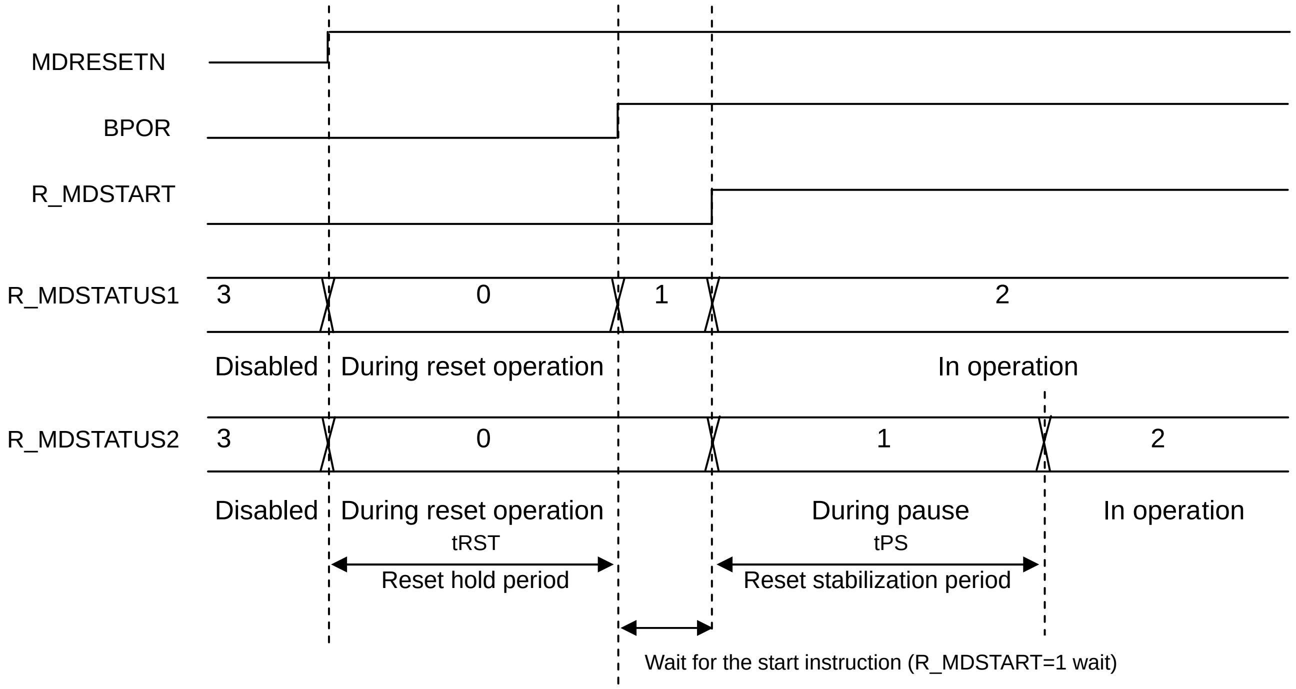


Figure 3.11.7 State timings with the status registers 1 and 2

13. eDRAM clock enable register (R_MDCKE)

Address = (0x4000_1000) + (0x030)

Bit	Bit Symbol	Type	Reset Value	Description
[0]	tCKE	RW	0y1	0y0: Clock disable 0y1: Clock enable

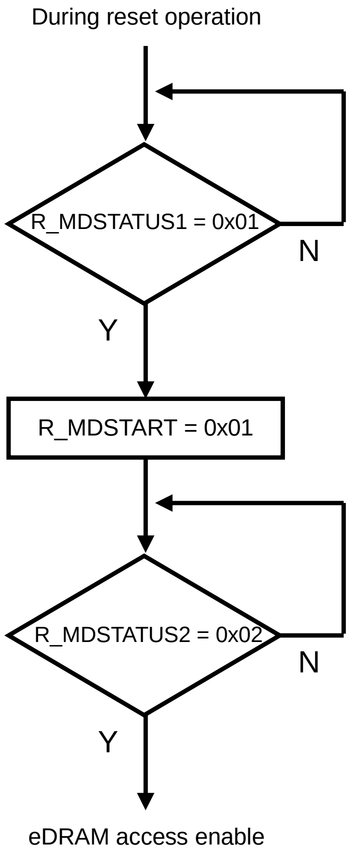
This register is used to enable and disable the eDRAM clock.

[Explanation]

a. <tCKE>
Enables or disables the clock for the eDRAM.

3.11.4 eDRAM Startup Sequence

The following shows the eDRAM startup sequence after reset is cleared.



3.12 16-bit Timer/PWM

3.12.1 Function Overview

Multifunction 16-bit timers for 8 channels are contained. They operate in the following two modes:

1) Free-running mode
2) Periodic timer mode
Supports PWM function

The circuit consists of 4 blocks, each associated with 2 channels. Of the 4 blocks, all the blocks 1 to 4 support PWM (Pulse Width Modulation) output.

	Block1		Block2	
	Timer0	Timer1	Timer2	Timer3
Free-Run	○	○	○	○
Peridiodic	○	○	○	○
One-shot	○	○	○	○
PWM	○	N/A	○	N/A
	PWM0OUT(PB0)	×	PWM2OUT(PB1)	×
Interrupt source signal	INTS[7]	INTS[8]	INTS[9]	INTS[10]
	Block3		Block4	
	Timer4	Timer5	Timer6	Timer7
Free-Run	○	○	○	○
Peridiodic	○	○	○	○
One-shot	○	○	○	○
PWM	○	N/A	○	N/A
	PWM0OUT(PB2)	×	PWM2OUT(PB3)	×
Interrupt source signal	INTS[11]	INTS[12]	INTS[13]	INTS[14]

Since all blocks are of the same specifications except for the PWM function and interrupt sources, the circuit of block 1 only is described in the following sections.

3.12.2 Block Diagram

The timer block, which has a built-in timer circuit with 2 channels, is composed of a programmable 16-bit free-run decrement counter. The TIMCLK input is used for counter operation. This clock is an internal system clock (f_{PCLK}).

Figure 3.12.1 shows the block diagram of the timer.

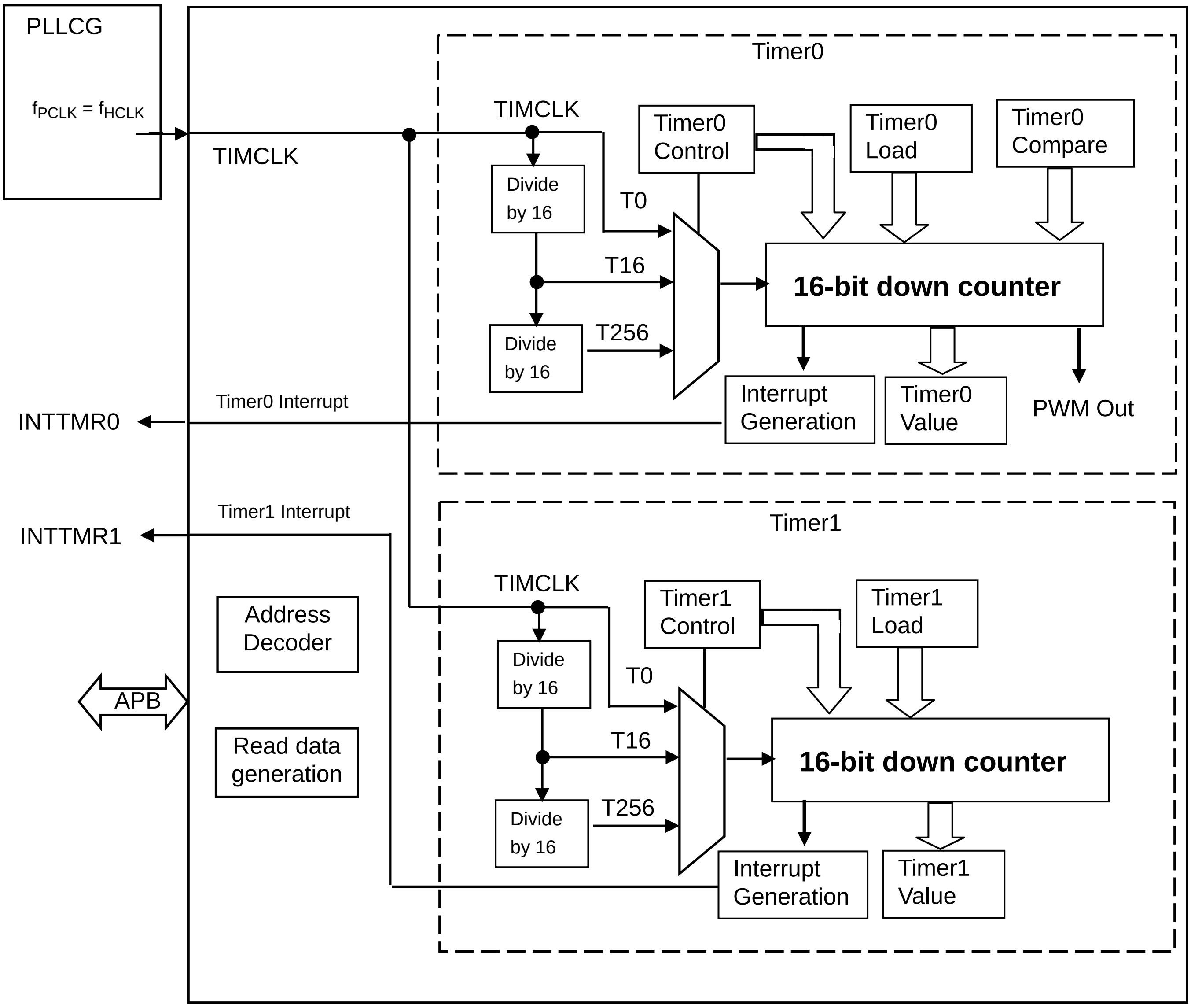


Figure 3.12.1 Timer block diagram (Timer0 and Timer1)

The timer clock (TIMCLK) is generated by the prescale unit.

T0: f_{PCLK} (=f_{HCLK})

T16: f_{PCLK} divided by 16. It is generated by a 4-bit prescale unit.

T256: f_{PCLK} divided by 256. It is generated by an 8-bit prescale unit.

3.12.3 Operation Description

The following description is about an example of the setting of Timer0. The timers of other channels operate similarly to Timer0.

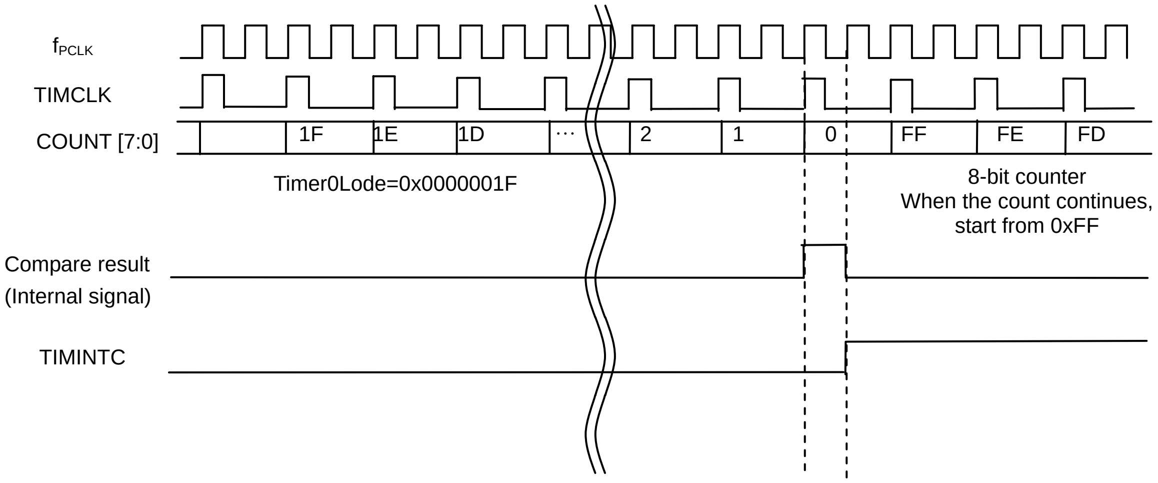
1) Free-running mode

When the timer is started, the counter decrements from the counter setting value. When the counter value becomes "0," an interrupt is generated.

In one-shot operation (Timer0Control<TIM0OSCTL>=“1”), an interrupt is generated only once.

In wrapping operation (Timer0Control<TIM0OSCTL>=“0”), the maximum counter value is reloaded to continue decrement. The maximum value is 0x000000FF for the 8-bit counter and 0x0000FFFF for the 16-bit counter.

The following figure shows an example when the timer value is set to 0x0000001F.



Example when the free-running mode is set (Wrapping-Operation)

Bits Register	MSB								LSB	Function
	[31:8]	7	6	5	4	3	2	1	0	
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stop the Timer0.
Timer0Load	0x000000	0	0	0	1	1	1	1	1	[15:0]: Set the Timer0 period to 0x0000001F.
Timer0Control	0x000000	1	0	1	0	0	0	0	0	[7]: Enable the Timer0. (Start the count) [6]: Select the free-run timer mode. [5]: Enable the timer interrupt. [3:2]: Select the input clock T0. [1]: Select the 8-bit counter. [0]: Select the Wrapping-Operation.

×: Don't care -: no change

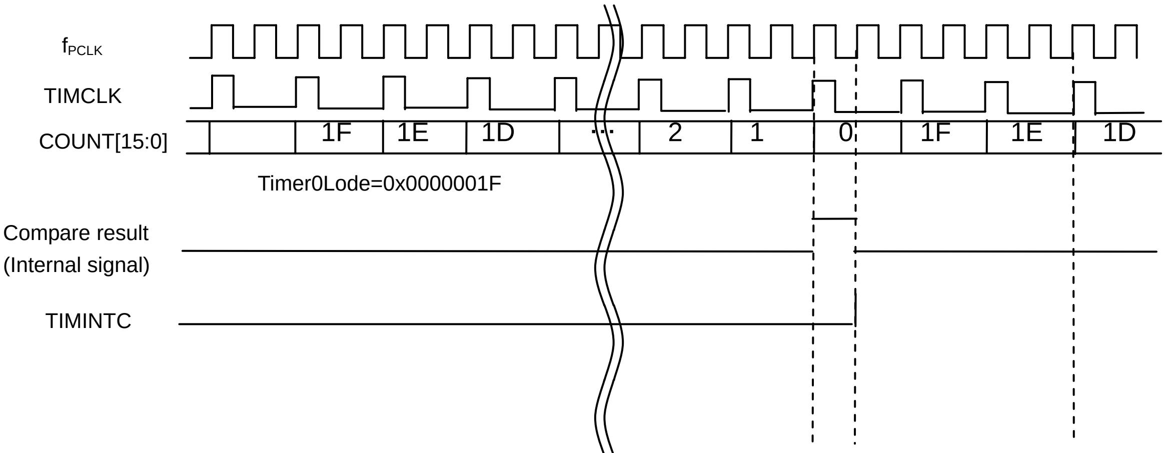
2) Periodic timer mode

When the timer is started, the counter decrements from the counter setting value. When the counter value becomes "0," an interrupt is generated.

In one-shot operation (Timer0Control<TIM0OSCTL>=“1”), an interrupt is generated only once.

In wrapping operation (Timer0Control<TIM0OSCTL>=“0”), the counter setting value is reloaded to continue decrement. Therefore, the interrupt occurs at fixed periods.

The following figure shows an example when the timer value is set to 0x0000001F.



Example when the periodic timer mode is set (Wrapping-Operation)

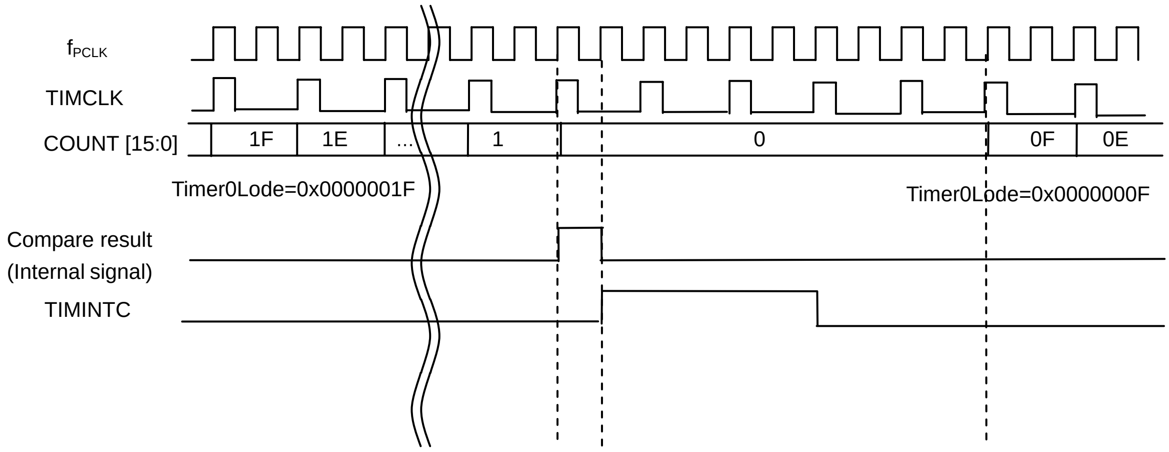
Bits / Register	MSB [31:8]	7	6	5	4	3	2	1	LSB 0	Function
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stop the Timer0.
Timer0Load	0x000000	0	0	0	1	1	1	1	1	[15:0]: Set the Timer0 period to 0x0000001F.
Timer0Control	0x000000	1	1	1	0	0	0	1	0	[7]: Enable the Timer0. (Start the count) [6]: Select the periodic timer mode.[5]: [5]: Enable the timer interrupt. [3:2]: Select the input clock T0. [1]: Select the 16-bit counter. [0]: Select the Wrapping-Operation.

×: Don't care -: no change

- Notes on the one-shot operation

To restart the timer in one-shot operation, set the Timer0Load register again.

If the Timer0Load register value is not set again and "1" is written in Timer0Control<TIM0EN>, the timer cannot restart.



Example when the free-running mode is set (One-shot operation)

Bits / Register	MSB							LSB	Function
	[31:8]	7	6	5	4	3	2	1	0	
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stop the Timer0.
Timer0Load	0x000000	0	0	0	1	1	1	1	1	[15:0]: Set the Timer0 period to 0x0000001F.
Timer0Control	0x000000	1	0	1	0	0	0	0	1	[7]: Enable the Timer0. (Start the count) [6]: Select the free-run timer mode. [5]: Enable the timer interrupt. [3:2]: Set the input clock to T0. [1]: Select the 8-bit counter. [0]: Select the one-shot operation.
Timer0IntClr	×	×	×	×	×	×	×	×	×	[32:0]: Writing any value can clear the interrupt.

×: Don't care -: no change

- Support for the PWM function

The 16-bit PWM function for 4 channels is provided in blocks 1 to 4. The PWM outputs for 4 channels are output to the PWM0OUT pin (PB0), PWM2OUT pin (PB1), PWM4OUT pin (PB2), and PWM6OUT pin (PB3).

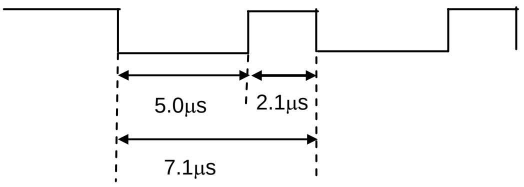
The PWM0OUT inverts the output when the decrement counter matches the setting value of the compare register Timer0Compare1 or when the counter specified in Timer0Mode<PWM Period> is decremented to "0."

It is possible to set Duty 0% to 100% for Timer0Compare1. When the decrement counter value becomes "0," it starts countdown from 2^n-2.

Since 2 channels are of the same specifications, this is also applied for Timer2.

Note) To use the PWM function, always set "periodic timer mode," "16-bit counter," and "Wrapping-Operation."

Example: When the Timer0 is used with f_{PCLK}=144MHz = TIMCLK=144MHz to output the following PWM waveform to the PWM0OUT pin (clock condition: high-speed clock gear 1/1)



(1) When T0 = 6.94ns, the PWM period of 7.1µs can be obtained by calculating the equation 7.1µs / 0.00694µs = 1023 = 2^n-1. Therefore, set n = 10.

(2) Because the "L" level period is 5.0µs, set (7.1µs - 5.0µs) / 0.00694µs = 303 = 0x12F for Timer0Compare1 when T0 = 6.94ns.

Bits Register	MSB							LSB	Function
	[15:8]	7	6	5	4	3	2	1	0	
Timer0Control	0x00	0	×	×	0	×	×	×	×	[7]: Stop the Timer0.
Timer0Mode	0x01	0	1	0	0	0	0	0	0	[6],[5:4]: Enable the PWM mode and set the PWM period to 2^{10}-1.
Timer0Compare1	0x00	0	0	1	1	0	1	1	1	[7:0]: Set the compare value 0x12F.
Timer0CmpEn	0x00	0	0	0	0	0	0	0	1	[0]: Enable the compare.
Timer0Control	0x00	1	1	1	0	0	0	1	0	[7]: Enable the Timer0. (Start the count) [6]: Select the periodic timer mode. [5]: Enable the timer interrupt. [3:2]: Set the input clock to T0. [1]: Select the 16-bit counter. (Always set to 16 bits) [0]: Select the Wrapping-Operation.

×: Don't care -: no change

The following explains the minimum PWM resolutions and duties.

Table 3.12.1 Minimum PWM resolutions (TIMCLK=144MHz)

PWM period / Prescaler	2^8 - 1	2^9 - 1	2^{10} - 1	2^{11} - 1
T0	1.77μs(564.7kHz)	3.55μs(281.8kHz)	7.10μs(140.8kHz)	1.42ms(70.35kHz)
T16	28.3μs(35.29kHz)	56.8μs(17.61kHz)	114μs(8.8kHzkHz)	227μs (4.4kHz)
T256	453μs(2.206kHz)	908μs(1.101kHz)	1.82ms(0.55kHz)	3.64ms(0.275kHz)

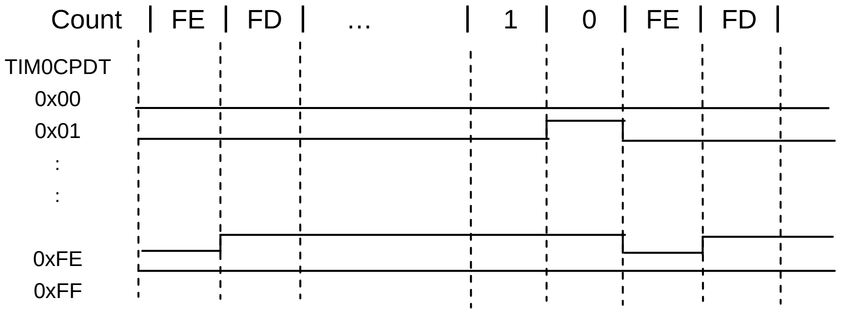


Figure 3.12.2 Example of PWM output waveform

Example: Duty when the period is 2^{10}-1 (1023 counts)

The initial value of the PWM output is always "L" output. Duty 0% and Duty 100% always indicate the "L" output and "H" output, respectively.

When "0x00" is set for Timer0Compare1, the duty is 0/1023 x 100=0%.

When "0x01" is set for Timer0Compare1, the duty is 1/1023 x 100=0.098%.

:

:

When "0x3FE" is set for Timer0Compare1, the duty is 1022/1023 x 100=99.9%.

When "0x3FF" is set for Timer0Compare1, the duty is 1023/1023 x 100=100%.

- In 2^n-1 mode, when 2^n-1 is set for Timer0Compare1, the F/F of the PWM output is set to "H." Therefore, to change the PWM period only after that and then start the operation, disable the PWM mode once before making settings again.
- To use the PWM mode, satisfy the following condition:
  0 ≤ (setting value of TimerxCompare1) ≤ 2^n-1

3.12.4 Explanation of the Register

The following lists the SFRs and their functions:

Table 3.12.2 SFR list (1/3) base address = 0x4000_F000

Register Name	Address (base+)	Description
Timer0Load	0x0000	Timer0 Load value
Timer0Value	0x0004	The current value for Timer0
Timer0Control	0x0008	Timer0 control register
Timer0IntClr	0x000C	Timer0 interrupt clear
Timer0RIS	0x0010	Timer0 raw interrupt status
Timer0MIS	0x0014	Timer0 masked interrupt status
Timer0BGLoad	0x0018	Background load value for Timer0
Timer0Mode	0x001C	Timer0 mode register
–	0x0020	Reserved
–	0x0040	Reserved
–	0x0060	Reserved
–	0x0064	Reserved
–	0x0068	Reserved
Timer0Compare1	0x00A0	Timer0 Compare value
Timer0CmpIntClr1	0x00C0	Timer0 Compare Interrupt clear
Timer0CmpEn	0x00E0	Timer0 Compare Enable
Timer0CmpRIS	0x00E4	Timer0 Compare raw interrupt status
Timer0CmpMIS	0x00E8	Timer0 Compare masked int status
Timer0BGCmp	0x00EC	Background compare value for Timer0
–	0x00F0	Reserved
Timer1Load	0x0100	Timer1 Load value
Timer1Value	0x0104	The current value for Timer1
Timer1Control	0x0108	Timer1 control register
Timer1IntClr	0x010C	Timer1 interrupt clear
Timer1RIS	0x0110	Timer1 raw interrupt status
Timer1MIS	0x0114	Timer1 masked interrupt status
Timer1BGLoad	0x0118	Background load value for Timer1
–	0x0120	Reserved
–	0x0140	Reserved
–	0x0160	Reserved
–	0x0164	Reserved
–	0x0168	Reserved
–	0x01A0	Reserved
–	0x01C0	Reserved
–	0x01E0	Reserved
–	0x01E4	Reserved
–	0x01E8	Reserved

Table 3.12.2 SFR list (2/3)

base address = 0x4001_0000

Register Name	Address (base+)	Description
Timer2Load	0x0000	Timer2 Load value
Timer2Value	0x0004	The current value for Timer2
Timer2Control	0x0008	Timer2 control register
Timer2IntClr	0x000C	Timer2 interrupt clear
Timer2RIS	0x0010	Timer2 raw interrupt status
Timer2MIS	0x0014	Timer2 masked interrupt status
Timer2BGLoad	0x0018	Background load value for Timer2
Timer2Mode	0x001C	Timer2 mode register
–	0x0020	Reserved
–	0x0040	Reserved
–	0x0060	Reserved
–	0x0064	Reserved
–	0x0068	Reserved
Timer2Compare1	0x00A0	Timer2 Compare value
Timer2CmpIntClr1	0x00C0	Timer2 Compare Interrupt clear
Timer2CmpEn	0x00E0	Timer2 Compare Enable
Timer2CmpRIS	0x00E4	Timer2 Compare raw interrupt status
Timer2CmpMIS	0x00E8	Timer2 Compare masked int status
Timer2BGCmp	0x00EC	Background compare value for Timer2
:	:	:
Timer3Load	0x0100	Timer3 Load value
Timer3Value	0x0104	The current value for Timer3
Timer3Control	0x0108	Timer3 control register
Timer3IntClr	0x010C	Timer3 interrupt clear
Timer3RIS	0x0110	Timer3 raw interrupt status
Timer3MIS	0x0114	Timer3 masked interrupt status
Timer3BGLoad	0x0118	Background load value for Timer3
–	0x0120	Reserved
–	0x0140	Reserved
–	0x0160	Reserved
–	0x0164	Reserved
–	0x0168	Reserved
–	0x01A0	Reserved
–	0x01C0	Reserved
–	0x01E0	Reserved
–	0x01E4	Reserved
–	0x01E8	Reserved

Table 3.12.2 SFR list (3/3) base address = 0x4001_1000

Register Name	Address (base+)	Description
Timer4Load	0x0000	Timer4 Load value
Timer4Value	0x0004	The current value for Timer4
Timer4Control	0x0008	Timer4 control register
Timer4IntClr	0x000C	Timer4 interrupt clear
Timer4RIS	0x0010	Timer4 raw interrupt status
Timer4MIS	0x0014	Timer4 masked interrupt status
Timer4BGLoad	0x0018	Background load value for Timer4
Timer4Mode	0x001C	Timer4 mode register
–	0x0020	Reserved
–	0x0040	Reserved
–	0x0060	Reserved
–	0x0064	Reserved
–	0x0068	Reserved
Timer4Compare1	0x00A0	Timer4 Compare value
Timer4CmpIntClr1	0x00C0	Timer4 Compare Interrupt clear
Timer4CmpEn	0x00E0	Timer4 Compare Enable
Timer4CmpRIS	0x00E4	Timer4 Compare raw interrupt status
Timer4CmpMIS	0x00E8	Timer4 Compare masked int status
Timer4BGCmp	0x00EC	Background compare value for Timer4
:	:	:
Timer5Load	0x0100	Timer5 Load value
Timer5Value	0x0104	The current value for Timer5
Timer5Control	0x0108	Timer5 control register
Timer5IntClr	0x010C	Timer5 interrupt clear
Timer5RIS	0x0110	Timer5 raw interrupt status
Timer5MIS	0x0114	Timer5 masked interrupt status
Timer5BGLoad	0x0118	Background load value for Timer5
–	0x0120	Reserved
–	0x0140	Reserved
–	0x0160	Reserved
–	0x0164	Reserved
–	0x0168	Reserved
–	0x01A0	Reserved
–	0x01C0	Reserved
–	0x01E0	Reserved
–	0x01E4	Reserved
–	0x01E8	Reserved

base address = 0x4001_2000

Register Name	Address (base+)	Description
Timer6Load	0x0000	Timer6 Load value
Timer6Value	0x0004	The current value for Timer6
Timer6Control	0x0008	Timer6 control register
Timer6IntClr	0x000C	Timer6 interrupt clear
Timer6RIS	0x0010	Timer6 raw interrupt status
Timer6MIS	0x0014	Timer6 masked interrupt status
Timer6BGLoad	0x0018	Background load value for Timer6
Timer6Mode	0x001C	Timer6 mode register
–	0x0020	Reserved
–	0x0040	Reserved
–	0x0060	Reserved
–	0x0064	Reserved
–	0x0068	Reserved
Timer6Compare1	0x00A0	Timer6 Compare value
Timer6CmpIntClr1	0x00C0	Timer6 Compare Interrupt clear
Timer6CmpEn	0x00E0	Timer6 Compare Enable
Timer6CmpRIS	0x00E4	Timer6 Compare raw interrupt status
Timer6CmpMIS	0x00E8	Timer6 Compare masked int status
Timer6BGCmp	0x00EC	Background compare value for Timer6
:	:	:
Timer7Load	0x0100	Timer7 Load value
Timer7Value	0x0104	The current value for Timer7
Timer7Control	0x0108	Timer7 control register
Timer7IntClr	0x010C	Timer7 interrupt clear
Timer7RIS	0x0110	Timer7 raw interrupt status
Timer7MIS	0x0114	Timer7 masked interrupt status
Timer7BGLoad	0x0118	Background load value for Timer7
–	0x0120	Reserved
–	0x0140	Reserved
–	0x0160	Reserved
–	0x0164	Reserved
–	0x0168	Reserved
–	0x01A0	Reserved
–	0x01C0	Reserved
–	0x01E0	Reserved
–	0x01E4	Reserved
–	0x01E8	Reserved

1. Timer0Load (Timer Load Register)

Address = (0x4000_F0000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	TIM0SD[15:0]	R/W	0x00	Set a Timer0 interval value.

[Explanation]

a. <TIMxSD[15:0]>

This register sets the timer period.

It uses a decrement counter, which can be set to 0x0001 to 0xFFFF (Set a value obtained by subtracting 1 from the desired timer period. It cannot be set to 0x0000).

To use the 8-bit counter, the value of the upper 8 bits is ignored.

When Wrapping-Operation is used in the periodic timer mode, this value is also used for reloading.

When a value is written in this register, the value will be updated immediately.

To update the value when the decrement counter value becomes 0x0000, write a value to the Timer0BGLoad register, which will be described later.

In addition, the read data will be the same value as in the Timer0BGLoad register.

- TimerxLoad (Timer x Load value register) (x = 0 ~ 7)

  Refer to the description on Timer0Load because the structures and explanations on the above registers are the same as Timer0Load. Also, refer to Table 3.12.2 SFR list for register names and addresses.

2. Timer0Value (Timer Data Register)

Address = (0x4000_F000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	TIM0CD[15:0]	RO	0x00	Current counter value for the Timer0

[Explanation]

a. <TIMxCD[15:0]>

This register is used to read the current timer value.

This register indicates the current value in the decrement counter.

TimerxValue (Timer x value register) (x = 0 ~ 7)

Refer to the description on Timer0Value because the structures and explanations on the above registers are the same as Timer0Value. Also, refer to Table 3.12.2 SFR list for register names and addresses.

3. Timer0Control (Timer Control Register)

Address = (0x4000_F000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	TIM0EN	R/W	0y0	Timer0 enable bit 0: Stop 1: Operate
[6]	TIM0MOD	R/W	0y0	Set a Timer0 mode 0: Free-run mode 1: Periodic timer mode
[5]	TIM0INTE	R/W	0y0	Enable/disable the Timer0 interrupt 0: Disable the interrupt 1: Enable the interrupt
[4]	–	–	Undefined	Read undefined. Write as zero.
[3:2]	TIM0PRS	R/W	0y00	Set a Timer0 prescaler 00: Same magnification 01: Divide by 16 10: Divide by 256 11: This setting cannot be used
[1]	TIM0SIZE	R/W	0y0	Switch between the Timer0 8/16 counters 0: 8-bit counter 1: 16-bit counter
[0]	TIM0OSCTL	R/W	0y0	Switch between the Timer0 One-Shot/Wrapping counters 0: Wrapping mode 1: One-Shot mode

[Explanation]

a. <TIMxEN>

This bit is used to execute/stop the timer operation.

0y0: Stop

0y1: Operate

b. <TIMxMOD>

This bit is used to switch between timer operation modes.

c. <TIMxINTE>

Enables/disables mask of the timer interrupt.

d. <TIMxPRS>

Sets the prescale by which the timer source clock is divided.

e. <TIMxSIZE>

Selects the 8- or 16-bit counter.

f. <TIMxOSCTL>

Selects the one-shot or wrapping counter.

TimerxControl (Timer x Control register) (x = 0 ~ 7)

Refer to the description on Timer0Control because the structures and explanations on the above registers are the same as Timer0Control. Also, refer to Table 3.12.2 SFR list for register names and addresses.

4. Timer0IntClr (Timer Interrupt Clear Register)

Address = (0x4000_F000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	TIM0INTCLR	WO	Undefined	Clear the Timer0 interrupt

[Explanation]

a. <TIMxINTCLR>

This register is used to clear the timer interrupt.

Any value can be written. Writing the value to this register will clear the interrupt.

(Any bus width is also selectable. 8, 16, and 32 are supported as the bus width.)

TimerxIntClr (Timer x Interrupt Clear register) (x = 0 ~ 7)

Refer to the description on Timer0IntClr because the structures and explanations on the above registers are the same as Timer0IntClr. Also, refer to Table 3.12.2 SFR list for register names and addresses.

5. Timer0RIS (Timer Interrupt Raw Flag Register)

Address = (0x4000_F000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	TIM0RIF	RO	0y0	Timer0 interrupt flag 0: Interrupt not present 1: Interrupt present

[Explanation]

a. <TIMxRIF>

This register indicates the interrupt status of the internal counter. It indicates the internal status regardless of the interrupt control condition set in TIMxCR<TIMxINTE>.

TimerxRIS (Timer x Interrupt Raw Flag register) (x = 0 ~ 7)

Refer to the description on Timer0RIS because the structures and explanations on the above registers are the same as Timer0RIS. Also, refer to Table 3.12.2 SFR list for register names and addresses.

6. Timer0MIS (Timer Interrupt Masked Flag Register)

Address = (0x4000_F000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	TIM0MIF	RO	0y0	Timer0 interrupt flag 0: Interrupt not present 1: Interrupt present

[Explanation]

a. <TIMxMIF>

This register indicates the interrupt status after passing through the interrupt mask circuit.

The flag status changes depending on the interrupt control condition set in TIMxCR< TIMxINTE>.

(When TIMxCR< TIMxINTE>=0, it always indicates "0.")

TimerxMIS (Timer x Interrupt Masked Flag register) (x = 0 ~ 7)

Refer to the description on Timer0MIS because the structures and explanations on the above registers are the same as Timer0MIS. Also, refer to Table 3.12.2 SFR list for register names and addresses.

7. Timer0BGLoad (Timer Back Ground Counter Data Register)

Address = (0x4000_F000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	TIM0BSD[15:0]	R/W	0x00	Set the Timer0 and BG counter interval value.

[Explanation]

a. <TIMxBSD[15:0]>

This register is used to set the counter value for background.

This value is used for reloading to the counter before the periodic mode is enabled.

This register provides the alternative method to access to the TimerxLoad register.

The difference is that even when a value is written in the TimerXBGLoad register, the counter will not resume immediately from the new value. When this register is read, the same value as that from the TimerxLoad register is returned.

TimerxBGLoad (Timer x Back Ground Counter Data register) (x = 0 ~ 7)

Refer to the description on Timer0BGLoad because the structures and explanations on the above registers are the same as Timer0BGLoad. Also, refer to Table 3.12.2 SFR list for register names and addresses.

8. Timer0Mode (Timer mode register)

Address = (0x4000_F000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read undefined. Write as zero.
[6]	PWM Mode	R/W	0y0	Select a PWM mode: 0: PWM Disable 1: PWM Enable
[5:4]	PWM Period	R/W	0y00	Select a period in PWM mode: 00 : 2^8 - 1 01 : 2^9 - 1 10 : 2^{10} - 1 11 : 2^{11} - 1
[3:0]	–	–	Undefined	Read undefined. Write as zero.

[Explanation]

a. <PWM Mode>
Selects PWM mode enable.

b. <PWM Period>
Selects a period in PWM mode.

TimerxMode (Timer x mode register) (x = 0, 2, 4, 6)

Refer to the description on Timer0Mode because the structures and explanations on the above registers are the same as Timer0Mode. Also, refer to Table 3.12.2 SFR list for register names and addresses.

9. Timer0Compare1(Timer Compare Value)

Address = (0x4000_F000) + 0x00A0

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	TIM0CPD	R/W	0x00	Set the value to be compared with the Timer0 counter value: 0x0001 ~ 0xFFFF

[Explanation]

a. <TIMxCPD>

TimerxCompare1 (Timer x Compare Value register) (x = 0, 2, 4, 6)

Refer to the description on Timer0Compare1 because the structures and explanations on the above registers are the same as Timer0Compare1. Also, refer to Table 3.12.2 SFR list for register names and addresses.

10. Timer0CmpIntClr1(Timer Compare Interrupt Clear Register)

Address = (0x4000_F000) + 0x00C0

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	TIM0CMINTCLR	WO	Undefined	Clear the Timer0 compare interrupt

[Explanation]

a. <TIMxCMINTCLR>

This register is used to clear the timer capture interrupt.

Any value can be written. Writing the value to this register will clear the interrupt.

(Any bus width is also selectable. 8, 16, and 32 are supported as the bus width.)

TimerxCmpIntClr1 (Timer x Compare Interrupt Clear register) (x = 0 ~ 7)

Refer to the description on Timer0CmpIntClr1 because the structures and explanations on the above registers are the same as Timer0CmpIntClr1. Also, refer to Table 3.12.2 SFR list for register names and addresses.

11. Timer0CmpEn (Timer Compare Enable Register)

Address = (0x4000_F000) + 0x00E0

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	TIM0CPE	R/W	0y0	Enable/disable the Timer0 compare operation. 0: Disable 1: Enable

[Explanation]

a. <TIMxCPE>

This register is used to enable the timer compare operation.

This register is also used for masking the interrupt.

TimerxCmpEn (Timer x Compare Enable register) (x = 0 ~ 7)

Refer to the description on Timer0CmpEn because the structures and explanations on the above registers are the same as Timer0CmpEn. Also, refer to Table 3.12.2 SFR list for register names and addresses.

12. Timer0CmpRIS (Timer Compare raw interrupt status Register)

Address = (0x4000_F000) + 0x00E4

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	TIM0CRIF	RO	0y0	Timer0 compare raw interrupt flag 0: Interrupt not present 1: Interrupt present

[Explanation]

a. <TIMxCRIF>

This register indicates the raw interrupt status generated by the compare. It indicates the interrupt status of the internal compare (which is not masked). Regardless of the interrupt control status set in TIMxCPMIS, the internal status is indicated.

TimerxCmpRIS (Timer x Compare raw interrupt status register) (x = 0 ~ 7)

Refer to the description on Timer0CmpRIS because the structures and explanations on the above registers are the same as Timer0CmpRIS. Also, refer to Table 3.12.2 SFR list for register names and addresses.

13. Timer0CmpMIS (Timer Compare Masked interrupt status Register)

Address = (0x4000_F000) + 0x00E8

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	TIM0CMIF	RO	0y0	Timer0 compare interrupt flag 0: Interrupt not present 1: Interrupt present

[Explanation]

a. <TIMxCMIF>

This register indicates the masked compare interrupt status.

TimerxCmpMIS (Timer x Compare Masked interrupt stateus register) (x = 0 ~ 7)

Refer to the description on Timer0CmpMIS because the structures and explanations on the above registers are the same as Timer0CmpMIS. Also, refer to the Table 3.12.2 SFR list for register names and addresses.

14. Timer0BGCmp (Timer Back Ground Compare Register)

Address = (0x4000_F000) + 0x00EC

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	TIM0BGCPD	R/W	0x0000	Set the BackGand value to be compared with the Timer0 counter value: 0x0001 ~ 0xFFFF

If a compare value to be reloaded during the timer operation in the periodic mode is written in the TIMxBGCPDT register, the count continues with the current value, and when the count value becomes "0," the value of the TIMxBGCPDT register is shifted into the TIMxCPDT compare register.

To use the PWM mode, satisfy the following conditions:

1 < (setting value of TIMxBGCPD) < 2^n-1

0 ≠ (setting value of TIMxBGCPD)

Do not write a value in the TIMxBGCPDT register when the count value is "0."

TimerxBGCmp (Timer x Back Ground Compare register) (x = 0, 2, 4, 6)

Refer to the description on Timer0BGCmp because the structures and explanations on the above registers are the same as Timer0BGCmp. Also, refer to the Table 3.12.2 SFR list for register names and addresses.

3.13 UART

This LSI contains four UART channels. The features of each channel are shown in the table below:

	Channels 0 to 3			
Transmit FIFO	8 bits wide / 32 tiers deep			
Receive FIFO	12 bits wide / 32 tiers deep			
Transmit-receive data format	Data: 5, 6, 7, or 8 bits can be set. Parity: With / Without STOP:1, 2bit			
FIFO ON/OFF	ON (FIFO mode) / OFF (character mode)			
Interrupt	(1) Combined interrupts of each interrupt source are output to an interrupt controller. (2) Enable/disable can be set on an interrupt source basis			
Baud rate generator	Baud rate generator for generating UART internal clocks shared between transmit and receive. Baud rate up to 3.0 Mbps (when PCLK = 144 MHz)			
DMA	Supported			
	Channel 0	Channel 1	Channel 2	Channel 3
Hardware flow control	RTS support CTS support		N/A	
Control pin	U0TXD U0RXD U0CTSn U0RTSn	U1TXD U1RXD U1CTSn U1RTSn	U2TXD U2RXD	U3TXD U3RXD

(1) UART transmit-receive data format

Transmit-receive data format			
START	DATA (LSB -> MSB)	PARITY	STOP

(2) Transmit FIFO data format

Bit	11	10	9	8	7	6	5	4	3	2	1	0
	Overrun error flag	Break error flag	Parity error flag	Framing error flag	Transmit DATA (MSB <- LSB)							
8-bit data transmit					1	1	1	1	1	1	1	1
7-bit data transmit					0	1	1	1	1	1	1	1
6-bit data transmit					0	0	1	1	1	1	1	1
5-bit data transmit					0	0	0	1	1	1	1	1

Data is received being aligned from the LSB. When receiving data less than 8 bits, the highest-order bit is 0.

3.13.1 Block Diagram

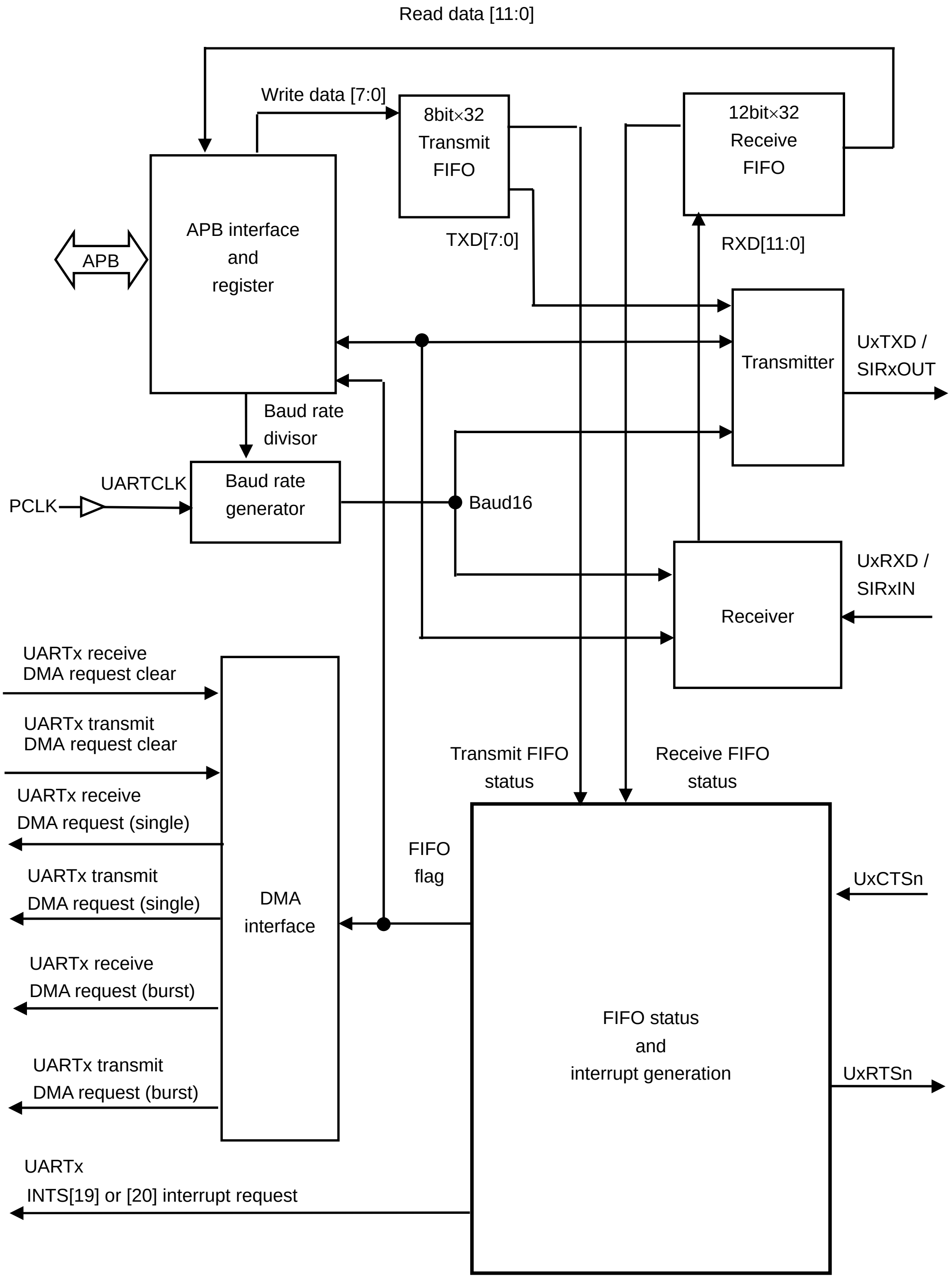


Figure 3.13.1 Block Diagram of UART Channels 0, 1

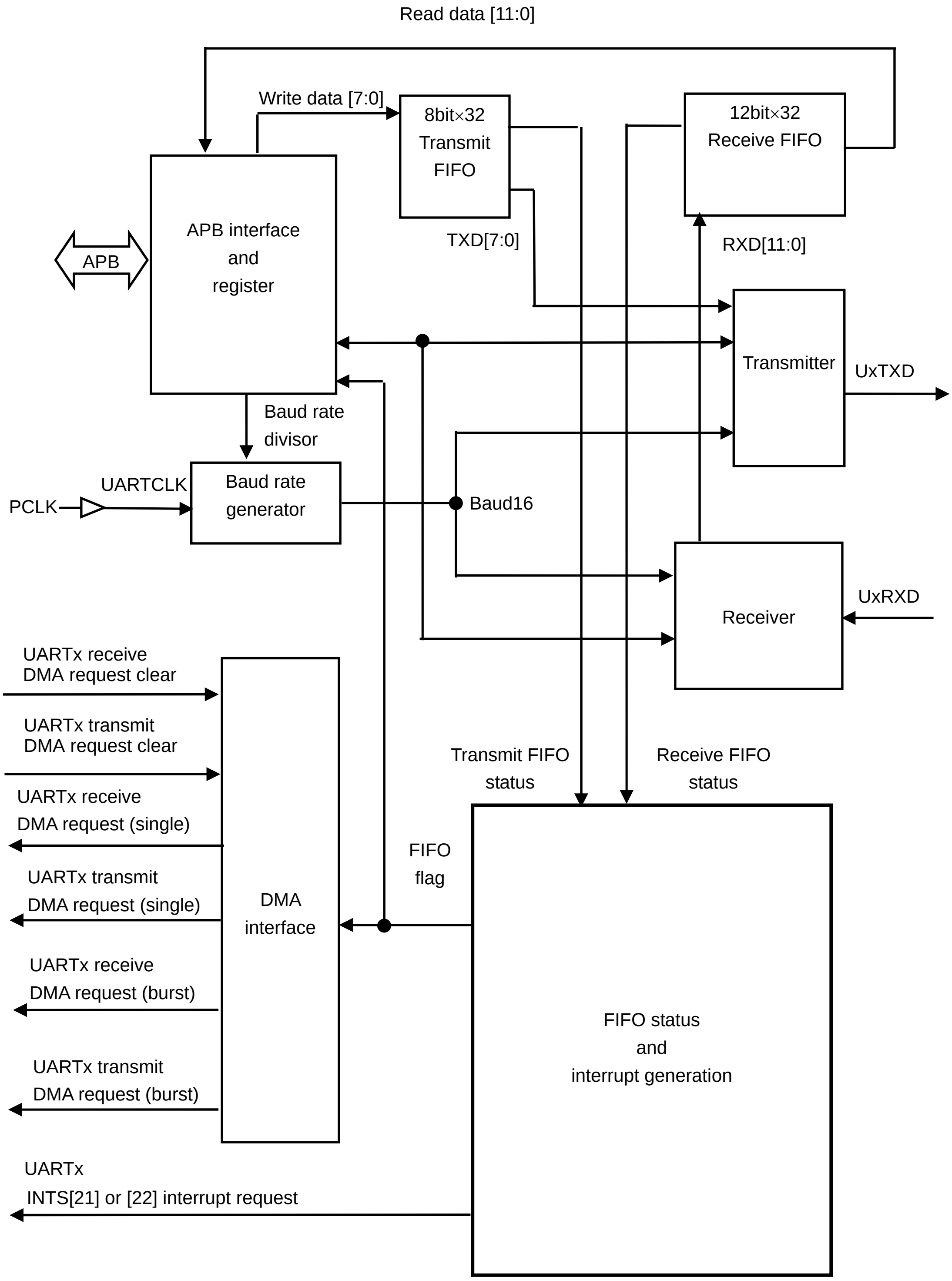


Figure 3.13.2 Block Diagram of UART Channels 2, 3

3.13.1.1 Functional Description

(1) Baud rate generator

The baud rate generator is composed of the internal clocks (Baud16) that generate the timing of UART transmit-receive control.

(2) Transmit FIFO

The transmit FIFO is an 8-bit wide, 32-tier deep, FIFO memory buffer. CPU data written across the APB interface is stored in the FIFO until it is read out by the transmit logic. You can disable the transmit FIFO to act like a one-byte holding register.

(3) Receive FIFO

The receive FIFO is a 12-bit wide, 32-tier deep, FIFO memory buffer. Received data and corresponding error bits are stored in the receive FIFO by the receive logic until read out by the CPU across the APB interface. You can disable the receive FIFO to act like a one-byte holding register.

(4) Transmit logic

The transmit logic performs parallel-to-serial conversion on the data read from the transmit FIFO. Control logic outputs the serial bit stream beginning with a start bit, data bits with the Least Significant Bit (LSB) first, followed by the parity bit, and then the stop bits according to the programmed configuration in control registers.

(5) Receive logic

The receive logic performs serial-to-parallel conversion on the received bit stream after a valid start pulse has been detected. Overrun, parity, frame error checking, and line break detection are also performed, and the data related to their error bits is written to the receive FIFO.

Hardware flow control

The hardware flow control function is selectable and can control serial data flows using the UxRTSn output and UxCTSn input signals.

Figure 3.13.5 shows how two devices can communicate with each other using hardware flow control.

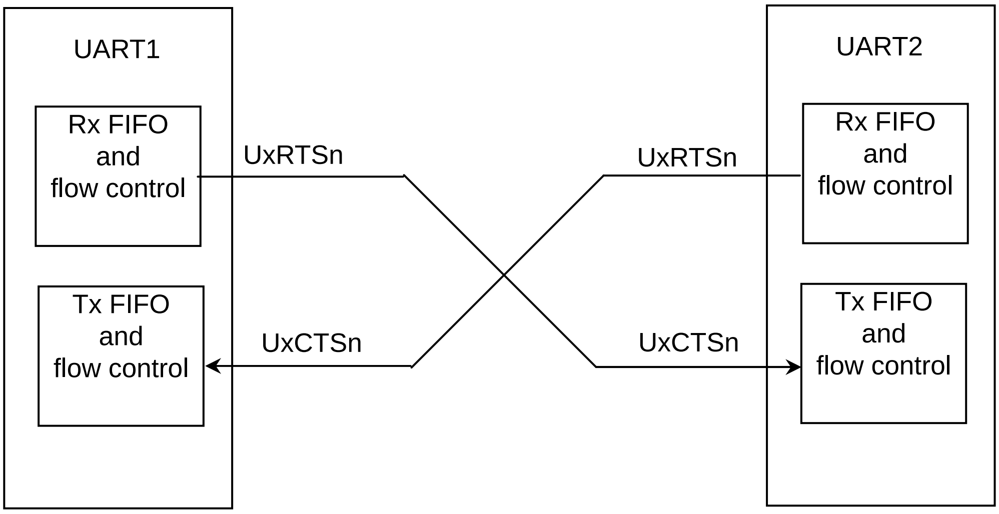


Figure 3.13.5 Hardware Flow Control between Two Similar Devices

RTS flow control

The RTS flow control logic is linked to the programmable receive FIFO watermark levels. When RTS flow control is enabled, the UxRTSn is asserted until the receive FIFO is filled up to the watermark level. When the receive FIFO watermark level is reached, the UxRTSn signal is deasserted, indicating that there is no more room to receive any more data. The transmission of data is expected to cease after the current character has been transmitted.

The UxRTSn signal is reasserted when data has been read out of the receive FIFO so that it is filled to less than the watermark level. If RTS flow control is disabled and the UART is still enabled, then data is received until the receive FIFO is full, or no more data is transmitted to it.

CTS flow control

If CTS flow control is enabled, then the transmitter checks the UxCTSn signal before transmitting the next byte. If the UxCTSn signal is asserted, it transmits the byte; otherwise, transmission does not occur.

The data continues to be transmitted while UxCTSn is asserted, and the transmit FIFO is not empty. If the transmit FIFO is empty, no data is transmitted even if the UxCTSn signal is asserted.

If the UxCTSn signal is deasserted and CTS flow control is enabled, then the current character transmission is completed before stopping. If CTS flow control is disabled and the UART is enabled, then the data continues to be transmitted until the transmit FIFO is empty.

Table 3.13 Control Bits to Enable and Disable Hardware Flow Control

UARTxCR		Description
CTSEn	RTSEn	
1	1	Both RTS and CTS flow control are enabled
1	0	Only CTS flow control is enabled
0	1	Only RTS flow control is enabled
0	0	Both RTS and CTS flow control are disabled

(6) DM A interface

The UART provides an interface to connect to a DMA controller. The DMA operation of the UART is controlled using the UART DMA Control Register, UARTDMACR. The DMA interface includes the following signals:

UARTx receive (DMA request: Single)

Single character DMA transfer request signal, asserted by the UART. For receive, one character consists of up to 12 bits. This signal is asserted when the receive FIFO contains at least one character.

UARTx receive (DMA request: Burst)

Burst DMA transfer request signal, asserted by the UART. This signal is asserted when the receive FIFO contains more characters than the programmed watermark level. You can program the watermark level for each FIFO using the UARTIFLS Register.

UARTx receive (DMA clear)

DMA request clear signal, asserted by a DMA controller to clear the receive request signals. If DMA burst transfer is requested, the clear signal is asserted during the transfer of the last data in the burst.

UARTx transmit (DMA request: Single)

Single character DMA transfer request signal, asserted by the UART. For transmit, one character consists of up to eight bits. This signal is asserted when there is at least one empty location in the transmit FIFO.

UARTx transmit (DMA request: Burst)

Burst DMA transfer request signal, asserted by the UART. This signal is asserted when the transmit FIFO contains less characters than the watermark level. You can program the watermark level for each FIFO using the UARTIFLS Register.

UARTx transmit (DMA clear)

DMA request clear signal, asserted by a DMA controller to clear the transmit request signals. If DMA burst transfer is requested, the clear signal is asserted during the transfer of the last data in the burst.

Since the burst transfer and single transfer request signals are not mutually exclusive, they can both be asserted at the same time. For example, when there is more data than the watermark level in the receive FIFO, the burst transfer request and the single transfer request are asserted. When the amount of data left in the receive FIFO is less than the watermark level, only the single request is asserted. This is useful for situations where the number of characters left to be received in the stream is less than a burst.

For example, if 19 characters have to be received and the watermark level is programmed to be four, the DMA controller then transfers four bursts of four characters and three single transfers to complete the stream.

Note) For the remaining three characters, the UART cannot assert the burst request.

(7) Interrupt generation logic

Individual maskable active HIGH interrupts are generated by the UART. You can enable or disable the individual interrupts by changing the mask bits in the UARTIMSC Register.

A combined interrupt output is also generated as an OR function of the individual interrupt requests.

The G-HOST outputs only the combined interrupt outputs UARTINTR_UARTX to an interrupt controller.

(8) Interrupt generation timing

Interrupt type	Mode	Interrupt generation timing
UARTREINTR	Overrun error interrupt	After a stop bit for excess data is received
	Break error interrupt	After a stop bit is received
	Parity error interrupt	After parity data is received
	Framing error interrupt	Bit data resulting in exceeded framing is received
UARTRTINTR	Receive timeout interrupt	After 511 clocks of Baud16 after data is captured in the receive FIFO
UARTTXINTR	Transmit interrupt	After the last data (MSB data) is transmitted
UARTRXINTR	Receive interrupt	After a stop bit is received
UARTMSIINTR	CTS interrupt	After a UxCTSn input changes

Note) A stop bit refers to the last stop bit. (Select a stop bit using UARXTLCR_H<STP2>. Selecting 1/2 bit is possible.)

(9) Interrupt

UARTINTR

The interrupts can be combined into a single output, that is an OR function of the individual masked sources. You can connect this output to an interrupt controller to set another level of masking on an individual peripheral basis.

The combined UART interrupt is asserted if any of the above interrupts are asserted and enabled.

UARTREINTR

The error interrupt is asserted when an error occurs in the reception of data by the UART. The interrupt can be caused by a number of different error conditions:

- UARTOEINTR due to an overrun error
- UARTBEINTR due to a break in the reception of data
- UARTPEINTR due to a parity error in received characters
- UARTFEINTR due to a framing error in received characters

You can determine the cause of the interrupt by reading the UARTRIS register or the UARTMIS Register. It can be cleared by writing to the relevant bits (bits 7 to 10 are the error clear bits) of the UARTxICR Register.

UARTRTINTR

The receive timeout interrupt is asserted when the receive FIFO is not empty, and no more data is received during the subsequent 32-bit period. The receive timeout interrupt is cleared either when the FIFO becomes empty through reading all the data (or by reading the holding register), or when "1" is written to the corresponding bit of the UARTxICR Register.

UARTTXINTR

The transmit interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO reaches the programmed trigger level. When this event occurs, the transmit interrupt is asserted HIGH. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level, or by clearing the interrupt.
- If the FIFOs are disabled (register having a depth of one byte) and there is no data present in the register's location, the transmit interrupt is asserted HIGH. It is cleared by performing a single write to the transmit FIFO, or by clearing the interrupt.

UARTRXINTR

The receive interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the receive FIFO reaches the programmed trigger level. When this event occurs, the receive interrupt is asserted HIGH. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level, or by clearing the interrupt.
- If the FIFOs are disabled (register having a depth of one byte) and data is received thereby filling the register, the receive interrupt is asserted HIGH. The receive interrupt is cleared by performing a single read of the receive FIFO, or by clearing the interrupt.

UARTMSIINTR

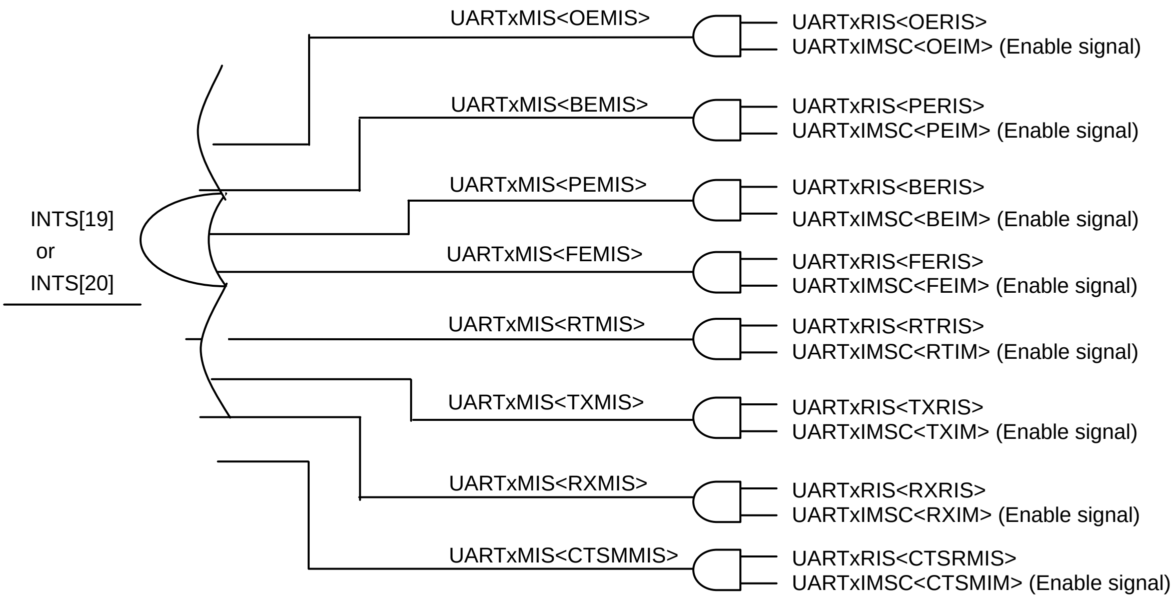
The modem status interrupt is asserted if any of the modem status lines (UxCTSn) change.

- UARTCTSINTR due to a change in UxCTSn modem status line

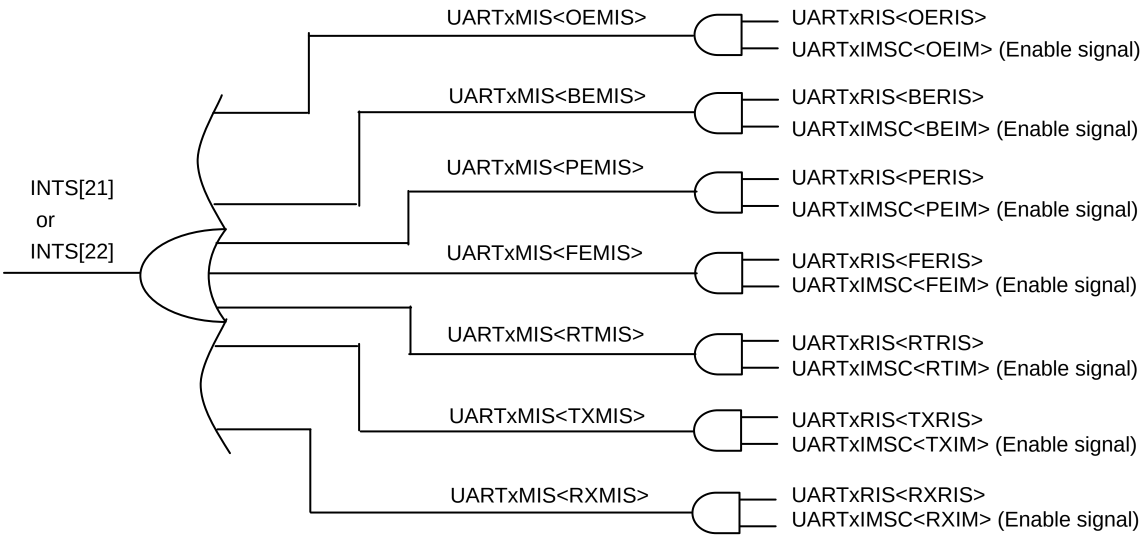
It is cleared by writing "1" to the corresponding bit(s) in the UARTxICR Register, depending on the modem status lines that generated the interrupt.

(10) Block diagram of UART interrupt

1) Block diagram of UART0,1 interrupt



2) Block diagram of UART2,3 interrupt



3.13.2 Registers

The following lists the registers.

- UART0

base address = 0x4001_9000

Register Name	Address (base+)	Description
UART0DR	0x000	Register for reading and writing data
UART0SR/ UART0ECR	0x004	Receive status register (read) / Error clear register (write)
–	0x008-0x014	Reserved
UART0FR	0x018	Flag register (read only)
–	0x01C	Reserved
-	0x020	Reserved
UART0IBRD	0x024	Integer baud rate divisor register
UART0FBRD	0x028	Fractional baud rate divisor register
UART0LCR_H	0x02C	Data format control register, HIGH byte
UART0CR	0x030	Control register
UART0IFLS	0x034	Interrupt FIFO level selection register
UART0IMSC	0x038	Interrupt enable set/clear
UART0RIS	0x03C	Pre-interrupt enable status
UART0MIS	0x040	Post-interrupt enable status
UART0ICR	0x044	Interrupt clear register
UART0DMACR	0x048	DMA control register
–	0x04C-0x07C	Reserved
–	0x080-0x08C	Reserved
–	0x090-0xFCC	Reserved
–	0xFD0-0xFDC	Reserved
–	0xFE0	Reserved
–	0xFE4	Reserved
–	0xFE8	Reserved
–	0xFEC	Reserved
–	0xFF0	Reserved
–	0xFF4	Reserved
–	0xFF8	Reserved
–	0xFFC	Reserved

Note) You must disable the UART before any of the registers are reprogrammed.
When the UART is disabled in the middle of transmission or reception, it completes the current transmission or reception before stopping.

- UART1 base address = 0x4001_A000

Register Name	Address (base+)	Description
UART1DR	0x000	Register for reading and writing data
UART1SR/ UART1ECR	0x004	Receive status register (read) / Error clear register (write)
–	0x008-0x014	Reserved
UART1FR	0x018	Flag register (read only)
–	0x01C	Reserved
UART1IBRD	0x024	Integer baud rate divisor register
UART1FBRD	0x028	Fractional baud rate divisor register
UART1LCR_H	0x02C	Data format control register, HIGH byte
UART1CR	0x030	Control register
UART1IFLS	0x034	Interrupt FIFO level selection register
UART1IMSC	0x038	Interrupt enable set/clear
UART1RIS	0x03C	Pre-interrupt enable status
UART1MIS	0x040	Post-interrupt enable status
UART1ICR	0x044	Interrupt clear register
UART1DMACR	0x048	DMA control register
–	0x04C-0x07C	Reserved
–	0x080-0x08C	Reserved
–	0x090-0xFCC	Reserved
–	0xFD0-0xFDC	Reserved
–	0xFE0	Reserved
–	0xFE4	Reserved
–	0xFE8	Reserved
–	0xFEC	Reserved
–	0xFF0	Reserved
–	0xFF4	Reserved
–	0xFF8	Reserved
–	0xFFC	Reserved

Note) You must disable the UART before any of the registers are reprogrammed.
When the UART is disabled in the middle of transmission or reception, it completes the current transmission or reception before stopping.

- UART2

base address = 0x4001_B000

Register Name	Address (base+)	Description
UART2DR	0x000	Register for reading and writing data
UART2SR/ UART2ECR	0x004	Receive status register (read) / Error clear register (write)
–	0x008-0x014	Reserved
UART2FR	0x018	Flag register (read only)
–	0x01C	Reserved
UART2IBRD	0x024	Integer baud rate divisor register
UART2FBRD	0x028	Fractional baud rate divisor register
UART2LCR_H	0x02C	Data format control register, HIGH byte
UART2CR	0x030	Control register
UART2IFLS	0x034	Interrupt FIFO level selection register
UART2IMSC	0x038	Interrupt enable set/clear
UART2RIS	0x03C	Pre-interrupt enable status
UART2MIS	0x040	Post-interrupt enable status
UART2ICR	0x044	Interrupt clear register
UART2DMACR	0x048	DMA control register
–	0x04C-0x07C	Reserved
–	0x080-0x08C	Reserved
–	0x090-0xFCC	Reserved
–	0xFD0-0xFDC	Reserved
–	0xFE0	Reserved
–	0xFE4	Reserved
–	0xFE8	Reserved
–	0xFEC	Reserved
–	0xFF0	Reserved
–	0xFF4	Reserved
–	0xFF8	Reserved
–	0xFFC	Reserved

Note) You must disable the UART before any of the registers are reprogrammed.
When the UART is disabled in the middle of transmission or reception, it completes the current transmission or reception before stopping.

- UART3

base address = 0x4001_C000

Register Name	Address (base+)	Description
UART3DR	0x000	Register for reading and writing data
UART3SR/ UART3ECR	0x004	Receive status register (read) / Error clear register (write)
–	0x008-0x014	Reserved
UART3FR	0x018	Flag register (read only)
–	0x01C	Reserved
UART3IBRD	0x024	Integer baud rate divisor register
UART3FBRD	0x028	Fractional baud rate divisor register
UART3LCR_H	0x02C	Data format control register, HIGH byte
UART3CR	0x030	Control register
UART3IFLS	0x034	Interrupt FIFO level selection register
UART3IMSC	0x038	Interrupt enable set/clear
UART3RIS	0x03C	Pre-interrupt enable status
UART3MIS	0x040	Post-interrupt enable status
UART3ICR	0x044	Interrupt clear register
UART3DMACR	0x048	DMA control register
–	0x04C-0x07C	Reserved
–	0x080-0x08C	Reserved
–	0x090-0xFCC	Reserved
–	0xFD0-0xFDC	Reserved
–	0xFE0	Reserved
–	0xFE4	Reserved
–	0xFE8	Reserved
–	0xFEC	Reserved
–	0xFF0	Reserved
–	0xFF4	Reserved
–	0xFF8	Reserved
–	0xFFC	Reserved

Note) You must disable the UART before any of the registers are reprogrammed.
When the UART is disabled in the middle of transmission or reception, it completes the current transmission or reception before stopping.

1. UART0DR (UART0 Data Register)

Address = (0x4001_9000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read undefined. Write as zero.
[11]	OE	RO	Undefined	Overrun error Read 0y0: Space present in FIFO 0y1: Overrun error flag Write Disabled
[10]	BE	RO	Undefined	Break error Read 0y0: No error 0y1: Error exists Write Disabled
[9]	PE	RO	Undefined	Parity error Read 0y0: No error 0y1: Error exists Write Disabled
[8]	FE	RO	Undefined	Framing error Read 0y0: No error 0y1: Error exists Write Disabled
[7:0]	DATA	R/W	Undefined	Read Received data Write Transmitted data

2. UART1DR (UART1 Data Register)

Address = (0x4001_A000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read undefined. Write as zero.
[11]	OE	RO	Undefined	Overrun error Read 0y0: Space present in FIFO 0y1: Overrun error flag Write Disabled
[10]	BE	RO	Undefined	Break error Read 0y0: No error 0y1: Error exists Write Disabled
[9]	PE	RO	Undefined	Parity error Read 0y0: No error 0y1: Error exists Write Disabled
[8]	FE	RO	Undefined	Framing error Read 0y0: No error 0y1: Error exists Write Disabled
[7:0]	DATA	R/W	Undefined	Read Received data Write Transmitted data

3. UART2DR (UART2 Data Register)

Address = (0x4001_B000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read undefined. Write as zero.
[11]	OE	RO	Undefined	Overrun error Read 0y0: Space present in FIFO 0y1: Overrun error flag Write Disabled
[10]	BE	RO	Undefined	Break error Read 0y0: No error 0y1: Error exists Write Disabled
[9]	PE	RO	Undefined	Parity error Read 0y0: No error 0y1: Error exists Write Disabled
[8]	FE	RO	Undefined	Framing error Read 0y0: No error 0y1: Error exists Write Disabled
[7:0]	DATA	R/W	Undefined	Read Received data Write Transmitted data

4. UART3DR (UART3 Data Register)

Address = (0x4001_C000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read undefined. Write as zero.
[11]	OE	RO	Undefined	Overrun error Read 0y0: Space present in FIFO 0y1: Overrun error flag Write Disabled
[10]	BE	RO	Undefined	Break error Read 0y0: No error 0y1: Error exists Write Disabled
[9]	PE	RO	Undefined	Parity error Read 0y0: No error 0y1: Error exists Write Disabled
[8]	FE	RO	Undefined	Framing error Read 0y0: No error 0y1: Error exists Write Disabled
[7:0]	DATA	R/W	Undefined	Read Received data Write Transmitted data

[Explanation]

a. <OE>

This bit is set to "1" if data is received and the FIFO is already full. Data received when the FIFO is full will not be updated in the FIFO but will be discarded.

This bit is cleared to "0" once there is an empty space in the FIFO and new data can be written to it.

b. <BE>

This bit is set to "1" if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits).

c. <PE>

When this bit is set to "1," it indicates that the parity of the received data does not match the parity defined by Bits 2 and 7 in the UARTLCR_H Register.

d. <FE>

When this bit is set to "1," it indicates that the received data did not have a valid stop bit (a valid stop bit is 1).

5. UART0SR/UART0ECR (UART0 Receive Status Register / UART0 Error Clear Register)

UART0SR and UART0ECR are mapped in the same address.

The function differs between read and write.

Address = (0x4001_9000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	–	Read undefined. Write as zero.
[3]	OE	RO	0y0	Overrun error 0y0: Space present in FIFO 0y1: Overrun error flag
[2]	BE	RO	0y0	Break error 0y0: No error 0y1: Error exists
[1]	PE	RO	0y0	Parity error 0y0: No error 0y1: Error exists
[0]	FE	RO	0y0	Framing error 0y0: No error 0y1: Error exists

Address = (0x4001_9000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	WO	–	When a write to this register is performed, framing, parity, break, and overrun errors will be cleared. There is no impact on data values. The register address is the same as the UART0SR Register.

6. UART1SR /UART1ECR (UART1 Receive Status Register / UART1 Error Clear Register)

Address = (0x4001_A000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	–	Read undefined. Write as zero.
[3]	OE	RO	0y0	Overrun error 0y0: Space present in FIFO 0y1: Overrun error flag
[2]	BE	RO	0y0	Break error 0y0: No error 0y1: Error exists
[1]	PE	RO	0y0	Parity error 0y0: No error 0y1: Error exists
[0]	FE	RO	0y0	Framing error 0y0: No error 0y1: Error exists

Address = (0x4001_A000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	WO	–	When a write to this register is performed, framing, parity, break, and overrun errors will be cleared. There is no impact on data values. The register address is the same as the UART1SR Register.

7. UART2SR/UART2ECR (UART2 Receive Status Register /UART2 Error Clear Register)

Address = (0x4001_B000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	–	Read undefined. Write as zero.
[3]	OE	RO	0y0	Overrun error 0y0: Space present in FIFO 0y1: Overrun error flag
[2]	BE	RO	0y0	Break error 0y0: No error 0y1: Error exists
[1]	PE	RO	0y0	Parity error 0y0: No error 0y1: Error exists
[0]	FE	RO	0y0	Framing error 0y0: No error 0y1: Error exists

Address = (0x4001_B000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	WO	–	When a write to this register is performed, framing, parity, break, and overrun errors will be cleared. There is no impact on data values. The register address is the same as the UART2SR Register.

8. UART3SR/UART3ECR (UART3 Receive Status Register / UART3 Error Clear Register)

Address = (0x4001_C000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	–	Read undefined. Write as zero.
[3]	OE	RO	0y0	Overrun error 0y0: Space present in FIFO 0y1: Overrun error flag
[2]	BE	RO	0y0	Break error 0y0: No error 0y1: Error exists
[1]	PE	RO	0y0	Parity error 0y0: No error 0y1: Error exists
[0]	FE	RO	0y0	Framing error 0y0: No error 0y1: Error exists

Address = (0x4001_C000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	WO	–	When a write to this register is performed, framing, parity, break, and overrun errors will be cleared. There is no impact on data values. The register address is the same as the UART3SR Register.

Note 1) The UARTxSR/UARTxECR Register is the receive status register/error clear register. Receive status can also be read from UARTxSR. If the status is read from this register, then the status information for break, framing and parity corresponds to the data read from UARTxDR prior to reading UARTxSR. The status information for overrun is set immediately when an overrun condition occurs. A write to UARTxECR clears the framing, parity, break, and overrun errors. All the bits are cleared to "0" on reset.

Note 2) The received data must be read first from UARTxDR before reading the error status associated with that data from UARTxSR. This read sequence cannot be reversed, because the status register UARTxSR is updated only when a read occurs from the data register UARTxDR. However, the status information can also be directly obtained by reading the UARTxDR Register.

[Explanation]

a. <OE>

This bit is set to "1" if data is received and the FIFO is already full. Data received when the FIFO is full will not be updated in the FIFO but will be discarded.

This bit is cleared to "0" once there is an empty space in the FIFO and new data can be written to it.

b. <BE>

This bit is set to "1" if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits).

c. <PE>

When this bit is set to"1," it indicates that the parity of the received data does not match the parity defined by Bits 2 and 7 in the UARTxLCR_H Register.

d. <FE>

When this bit is set to "1," it indicates that the received data did not have a valid stop bit (a valid stop bit is 1).

9. UART0FR (UART0 Flag Register)

In terms of meanings, the bits <TXFE>, <RXFF>, <TXFF>, and <RXFE> of the following register depend on the status of UART0LCR_H<FEN>.

Address = (0x4001_9000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description	
				FIFO mode (FEN = 1)	Character mode (FEN = 0)
[31:9]	–	–	Undefined	Read undefined.	
[8]	–	–	Undefined	Reserved	
[7]	TXFE	RO	0y1	Transmit FIFO empty flag 0y1: Empty 0y0: Not empty	Transmit holding register is empty flag 0y1: Empty 0y0: Not empty
[6]	RXFF	RO	0y0	Receive FIFO is full flag 0y1: Full 0y0: Not full	Receive holding register is full flag 0y1: Full 0y0: Not full
[5]	TXFF	RO	0y0	Transmit FIFO is full flag 0y1: Full 0y0: Not full	Transmit holding register is full flag 0y1: Full 0y0: Not full
[4]	RXFE	RO	0y1	Receive FIFO empty flag 0y1: Empty 0y0: Not empty	Receive holding register is empty flag 0y1: Empty 0y0: Not empty
[3]	BUSY	RO	0y0	BUSY flag: 0y1: Data is being transmitted (BUSY) 0y0: Stop	
[2]	–	–	Undefined	Reserved	
[1]	–	–	Undefined	Reserved	
[0]	CTS	RO	Undefined	Transmittable clear flag 0y1: When modem status input is "0"	

10. UART1FR (UART1 Flag Register)

Address = (0x4001_A000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description	
				FIFO mode (FEN = "1")	Character mode (FEN = "1")
[31:9]	–	–	Undefined	Read undefined.	
[8]	–	–	Undefined	Reserved	
[7]	TXFE	RO	0y1	Transmit FIFO empty flag 0y1: Empty 0y0: Not empty	Transmit holding register is empty flag 0y1: Empty 0y0: Not empty
[6]	RXFF	RO	0y0	Receive FIFO is full flag 0y1: Full 0y0: Not full	Receive holding register is full flag 0y1: Full 0y0: Not full
[5]	TXFF	RO	0y0	Transmit FIFO is full flag 0y1: Full 0y0: Not full	Transmit holding register is full flag 0y1: Full 0y0: Not full
[4]	RXFE	RO	0y1	Receive FIFO empty flag 0y1: Empty 0y0: Not empty	Receive holding register is empty flag 0y1: Empty 0y0: Not empty
[3]	BUSY	RO	0y0	BUSY flag: 0y1: Data is being transmitted (BUSY) 0y0: Stop	
[2]	–	–	Undefined	Reserved	
[1]	–	–	Undefined	Reserved	
[0]	CTS	RO	Undefined	Transmittable clear flag 0y1: When modem status input is "0"	

11. UART2FR (UART2 Flag Register)

Address = (0x4001_B000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description	
				FIFO mode (FEN = 1)	Character mode (FEN = 0)
[31:9]	–	–	Undefined	Read undefined.	
[8]	–	–	Undefined	Reserved	
[7]	TXFE	RO	0y1	Transmit FIFO empty flag 0y1: Empty 0y0: Not empty	Transmit holding register is empty flag 0y1: Empty 0y0: Not empty
[6]	RXFF	RO	0y0	Receive FIFO is full flag 0y1: Full 0y0: Not full	Receive holding register is full flag 0y1: Full 0y0: Not full
[5]	TXFF	RO	0y0	Transmit FIFO is full flag 0y1: Full 0y0: Not full	Transmit holding register is full flag 0y1: Full 0y0: Not full
[4]	RXFE	RO	0y1	Receive FIFO empty flag 0y1: Empty 0y0: Not empty	Receive holding register is empty flag 0y1: Empty 0y0: Not empty
[3]	BUSY	RO	0y0	BUSY flag: 0y1: Data is being transmitted (BUSY) 0y0: Stop	
[2:0]	-	-	Undefined	Read undefined.	

12. UART3FR (UART3 Flag Register)

Address = (0x4001_C000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description	
				FIFO mode (FEN = 1)	Character mode (FEN = 0)
[31:9]	–	–	Undefined	Read undefined.	
[8]	–	–	Undefined	Reserved	
[7]	TXFE	RO	0y1	Transmit FIFO empty flag 0y1: Empty 0y0: Not empty	Transmit holding register is empty flag 0y1: Empty 0y0: Not empty
[6]	RXFF	RO	0y0	Receive FIFO is full flag 0y1: Full 0y0: Not full	Receive holding register is full flag 0y1: Full 0y0: Not full
[5]	TXFF	RO	0y0	Transmit FIFO is full flag 0y1: Full 0y0: Not full	Transmit holding register is full flag 0y1: Full 0y0: Not full
[4]	RXFE	RO	0y1	Receive FIFO empty flag 0y1: Empty 0y0: Not empty	Receive holding register is empty flag 0y1: Empty 0y0: Not empty
[3]	BUSY	RO	0y0	BUSY flag: 0y1: Data is being transmitted (BUSY) 0y0: Stop	
[2:0]	-	-	Undefined	Read undefined.	

[Explanation]

a. <BUSY>

If this bit is set to "1," the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register.

b. <CTS>

Clear to send (UxCTSn): This bit is set to "1," when the modem status input is "0."

13. UART0IBRD (UART0 Integer Baud Rate Divisor Register)

Address = (0x4001_9000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	BAUDDIVINT	R/W	0x0000	Baud rate integer 0x0001 ~ 0xffff

14. UART1IBRD (UART1 Integer Baud Rate Divisor Register)

Address = (0x4001_A000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	BAUDDIVINT	R/W	0x0000	Baud rate integer 0x0001 ~ 0xffff

15. UART2IBRD (UART2 Integer Baud Rate Divisor Register)

Address = (0x4001_B000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	BAUDDIVINT	R/W	0x0000	Baud rate integer 0x0001 ~ 0xffff

16. UART3IBRD (UART3 Integer Baud Rate Divisor Register)

Address = (0x4001_C000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	BAUDDIVINT	R/W	0x0000	Baud rate integer 0x0001 ~ 0xffff

[Explanation]

a. <BAUDDIVINT>

Together with the fractional baud rate divisor explained next, this serves as the baud rate divisor BAUDDIV.

Note) To internally update the contents of UARTxIBRD, a UARTxLCR_H write must always be performed at the end. See UARTxLCR_H for more information.

17. UART0FBRD (UART0 Fractional Baud Rate Divisor Register)

Address = (0x4001_9000) + 0x0028

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:0]	BAUDDIVFRAC	R/W	0y000000	Baud rate fraction 0x01 ~ 0x3f

18. UART1FBRD (UART1 Fractional Baud Rate Divisor Register)

Address = (0x4001_A000) + 0x0028

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:0]	BAUDDIVFRAC	R/W	0y000000	Baud rate fraction 0x01 ~ 0x3f

19. UART2FBRD (UART2 Fractional Baud Rate Divisor Register)

Address = (0x4001_B000) + 0x0028

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:0]	BAUDDIVFRAC	R/W	0y000000	Baud rate fraction 0x01 ~ 0x3f

20. UART3FBRD (UART3 Fractional Baud Rate Divisor Register)

Address = (0x4001_C000) + 0x0028

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:0]	BAUDDIVFRAC	R/W	0y000000	Baud rate fraction 0x01 ~ 0x3f

[Explanation]

a. <BAUDDIVFRAC>

You can calculate the 6-bit number (BAUDDIVFRAC) by taking the fractional part of the required baud rate divisor and multiplying it by 64 (2^n , where n is the width of the UARTxFBRD Register) and adding 0.5 to account for rounding errors:

$$\text{BAUDDIVFRAC} = \text{INT (fraction} \times 2^n + 0.5)$$

Remark: How to Calculate the Baud Rate

You can obtain the baud rate by calculating the divisor of the baud rate to be set.

The baud rate divisor is calculated as follows:

BAUDDIV (baud rate divisor) = $(f_{UARTCLK})/(16 \times \text{Baud rate})$

= Integer value (BAUDDIVINT). Fractional value (BAUDDIVFRAC)

(Note 1) where fUARTCLK is the UART reference clock frequency (= PCLK).

(Note 2) BAUDDIV is comprised of the integer value (BAUDDIVINT) and the fractional value (BAUDDIVFRAC).

(Note 3) To internally update the contents of UARTxFBRD, UARTxLCR_H must always be written at the end. See UARTxLCR_H for more information.

Example: How to calculate the register setting value when you want to generate a baud rate of 115.2 kbps with $f_{UARTCLK}$ = 144MHz

Baud rate divisor = $(144 \times 10^6)/(16 \times 115200) = 78.125$

This means Integer = 78 and Divisor = 0.125.

Therefore, fractional part, $0.125 \times 64 + 0.5 = 8.5$

For the set value of fractional baud rate divisor, take 0x8, the integer part of 8.5.

BAUDDIVINT = 0x4E

BAUDDIVFRAC = 0x8

- Error from theoretical value

  Generated baud rate divisor = 78+ 8/64 = 78.125

  Generated baud rate =(144× 106)/ (16 × 78.125)= 115200

  Error = (115200 – 115200)/ 115200 × 100 = 0.000%

  The maximum error using a 6-bit UARTxFBRD Register = 1/64 × 100 = 1.56%.

  This occurs when BAUDDIVFRAC = 1, and the error is cumulative over 64 clock ticks.

Standard baud rate setting examples

When operating at $f_{UARTCLK}$ = 144MHz

Required bit rate (bps)	Integer value to be set (divisor)	Fractional value to be set (divisor)	Generated bit rate (bps)	Theoretical value error(%)
921600	0x9	0x31	921600	0.0000%
460800	0x13	0x22	460800	0.0000%
230400	0x27	0x4	230400	0.0000%
115200	0x4E	0x8	115200	0.0000%
76800	0x75	0xC	76800	0.0000%
57600	0x9C	0x10	57600	0.0000%
38400	0xEA	0x18	38400	0.0000%
19200	0x1D4	0x30	19200	0.0000%
14400	0x271	0x0	14400	0.0000%
9600	0x3A9	0x20	9600	0.0000%
2400	0xEA6	0x0	2400	0.0000%
1200	0x1D4C	0x0	1200	0.0000%

When operating at $f_{UARTCLK}$ = 12 MHz

Required bit rate (bps)	Integer value to be set (divisor)	Fractional value to be set (divisor)	Generated bit rate (bps)	Theoretical value error (%)
921600	0x0	0x34	921600	0.160%
460800	0x1	0x28	461538	0.160%
230400	0x3	0x10	230769	0.160%
115200	0x6	0x21	115108	-0.080%
76800	0x9	0x31	76800	0.000%
57600	0xD	0x1	57623	0.040%
38400	0x13	0x22	38400	0.000%
19200	0x27	0x4	19200	0.000%
14400	0x34	0x5	14401	0.010%
9600	0x4E	0x8	9600	0.000%
2400	0x138	0x20	2400	0.000%
1200	0x271	0x0	1200	0.000%

21. UART0LCR_H (UART0 Data Format Control Register)

Address = (0x4001_9000) + 0x002C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	SPS	R/W	0y0	Stick parity select (See Table 3.13 the parity truth table)
[6:5]	WLEN	R/W	0y00	Word length 0y00: 5 bits 0y01 : 6 bits 0y10: 7 bits 0y11 : 8 bits
[4]	FEN	R/W	0y0	FIFO control 0y1: FIFO mode 0y0: Character mode
[3]	STP2	R/W	0y0	Stop bit length 0y0: 1 bit 0y1: 2 bits
[2]	EPS	R/W	0y0	Parity bit select (See Table 3.13 the parity truth table) 0y1: Even parity 0y0: Odd parity
[1]	PEN	R/W	0y0	Parity control (See Table 3.13 the parity truth table) 0y0: Disable 0y1: Enable
[0]	BRK	R/W	0y0	Send break 0y0: Disabled 0y1: Send break

22. UART1LCR_H (UART1 Data Format Control Register)

Address = (0x4001_A000) + 0x002C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	SPS	R/W	0y0	Stick parity select See Table 3.13 the parity truth table
[6:5]	WLEN	R/W	0y00	Word length 0y00: 5 bits 0y01 : 6 bits 0y10: 7 bits 0y11 : 8 bits
[4]	FEN	R/W	0y0	FIFO control 0y1: FIFO mode 0y0: Character mode
[3]	STP2	R/W	0y0	Stop bit length 0y0: 1 bit 0y1: 2 bits
[2]	EPS	R/W	0y0	Even parity select (See Table 3.13 the parity truth table) 0y1: Even parity 0y0: Odd parity
[1]	PEN	R/W	0y0	Parity control (See Table 3.13 the parity truth table) 0y0: Disable 0y1: Enable
[0]	BRK	R/W	0y0	Send break 0y0: Disabled 0y1: Send break

23. UART2LCR_H (UART2 Data Format Control Register)

Address = (0x4001_B000) + 0x002C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	SPS	R/W	0y0	Stick parity select See Table 3.13 the parity truth table
[6:5]	WLEN	R/W	0y00	Word length 0y00: 5 bits 0y01 : 6 bits 0y10: 7 bits 0y11 : 8 bits
[4]	FEN	R/W	0y0	FIFO control 0y1: FIFO mode 0y0: Character mode
[3]	STP2	R/W	0y0	Stop bit length 0y0: 1 bit 0y1: 2 bits
[2]	EPS	R/W	0y0	Even parity select (See Table 3.13 the parity truth table) 0y1: Even parity 0y0: Odd parity
[1]	PEN	R/W	0y0	Parity control (See Table 3.13 the parity truth table) 0y0: Disable 0y1: Enable
[0]	BRK	R/W	0y0	Send break 0y0: Disabled 0y1: Send break

24. UART3LCR_H (UART3 Data Format Control Register)

Address = (0x4001_C000) + 0x002C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	SPS	R/W	0y0	Stick parity select See Table 3.13 the parity truth table
[6:5]	WLEN	R/W	0y00	Word length 0y00: 5 bits 0y01 : 6 bits 0y10: 7 bits 0y11 : 8 bits
[4]	FEN	R/W	0y0	FIFO control 0y1: FIFO mode 0y0: Character mode
[3]	STP2	R/W	0y0	Stop bit length 0y0: 1 bit 0y1: 2 bits
[2]	EPS	R/W	0y0	Even parity select (See Table 3.13 the parity truth table) 0y1: Even parity 0y0: Odd parity
[1]	PEN	R/W	0y0	Parity control (See Table 3.13the parity truth table) 0y0: Disable 0y1: Enable
[0]	BRK	R/W	0y0	Send break 0y0: Disabled 0y1: Send break

[Explanation]

a. <SPS>

If Bits 1, 2, and 7 of the UARTxLCR_H Register are set, then the parity bit is transmitted and checked as 0. If Bits 1 and 7 are set and Bit 2 is 0, then the parity bit is transmitted and checked as 1. If this bit is cleared, the stick parity is disabled. For the parity truth table of SPS, EPS, and PEN bits, see Table 3.13.

b. <WLEN>

These bits indicate the number of data bits transmitted or received in a frame.

c. <FEN>

If this bit is set to "1," the transmit and receive FIFO buffers are enabled

(FIFO mode).

If this bit is cleared to "0," these FIFOs are disabled (character mode), that is, the FIFOs become 1-byte-deep holding registers.

d. <STP2>

If this bit is set to "1," two stop bits are transmitted at the end of the frame. The receive logic does not check for two (second) stop bits being received.

e. <EPS>

If this bit is set to "1," even parity is generated and checked during transmission and reception. In this check, the UART checks for an even number of ones in the data and parity bits. If this bit is cleared to "0," the odd parity check is executed to check for an odd number of ones. This bit has no effect when the parity is disabled by clearing the parity enable (Bit 1) to "0." See Table 3.13.

f. <PEN>

If this bit is set to "1," parity checking and generation are enabled. In other settings, parity is disabled and no parity bit is added to the data frame. For the parity truth table of SPS, EPS, and PEN bits, see Table 3.13.

g. <BRK>

If this bit is set to "1," a low-level is continually output on the UxTXD output, after completing transmission of the current character. To generate break conditions, the software must assert this bit for at least a transmission time of one complete frame. Even when break conditions are generated, the contents of the transmit FIFO will not be affected.

Note) The UARTxLCR_H, UARTxIBRD, and UARTxFBRD registers are updated on a single write strobe generated by UARTxLCR_H write. So, to internally update the contents of UARTxIBRD or UARTxFBRD, a UARTxLCR_H write must always be performed at the end. Therefore, to update these three registers, there are two possible sequences:

- UARTxIBRD write, UARTxFBRD write, and UARTxLCR_H write
- UARTxFBRD write, UARTxIBRD write, and UARTxLCR_H write

To update UARTxIBRD or UARTxFBRD only:

- UARTxIBRD write (or UARTxFBRD write) and UARTxLCR_H write

The table below is a truth table for UARTxLCR_H <SPS>, <EPS>, and <PEN>.

Table 3.13 Truth Table for UARTxLCR_H <SPS>, <EPS>, and <PEN>

Parity enable (PEN)	Even parity select (EPS)	Stick parity select (SPS)	Parity bit (transmitted or checked)
0	x	x	Not transmitted or checked
1	1	0	Even parity
1	0	0	Odd parity
1	0	1	1
1	1	1	0

25. UART0CR (UART0 Control Register)

Address = (0x4001_9000) + 0x0030

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15]	CTSEn	R/W	0y0	CTS hardware flow control enable 0y0: Disable 0y1: Enable
[14]	RTSEn	R/W	0y0	RTS hardware flow control enable 0y0: Disable 0y1: Enable
[13:12]	–	–	Undefined	Read undefined. Write as zero
[11]	RTS	R/W	0y0	Complement of the UART transmit request (U0RTSn) modem status output 0y0: Modem status "1" output 0y1: Modem status "0" output
[10]	–	–	Undefined	Read undefined. Write as zero
[9]	RXE	R/W	0y1	UART receive enable 0y0: Disable 0y1: Enable
[8]	TXE	R/W	0y1	UART transmit enable 0y0: Disable 0y1: Enable
[7]	–	R/W	0y0	Read undefined. Write as zero.
[6:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero
[1]	–	–	Undefined	Read undefined. Write as zero
[0]	UARTEN	R/W	0y0	UART enable 0y0: Disable 0y1: Enable

26. UART1CR (UART1 Control Register)

Address = (0x4001_A000) + 0x0030

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15]	CTSEn	R/W	0y0	CTS hardware flow control enable 0y0: Disable 0y1: Enable
[14]	RTSEn	R/W	0y0	RTS hardware flow control enable 0y0: Disable 0y1: Enable
[13:12]	–	–	Undefined	Read undefined. Write as zero
[11]	RTS	R/W	0y0	Complement of the UART transmit request (U1RTSn) modem status output 0y0: Modem status "1" output 0y1: Modem status "0" output
[10]	–	–	Undefined	Read undefined. Write as zero
[9]	RXE	R/W	0y1	UART receive enable 0y0: Disable 0y1: Enable
[8]	TXE	R/W	0y1	UART transmit enable 0y0: Disable 0y1: Enable
[7]	–	R/W	0y0	Read undefined. Write as zero.
[6:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero
[1]	–	–	Undefined	Read undefined. Write as zero
[0]	UARTEN	R/W	0y0	UART enable 0y0: Disable 0y1: Enable

27. UART2CR (UART2 Control Register)

Address = (0x4001_B000) + 0x0030

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15]	–	–	Undefined	Read undefined. Write as zero
[14]	–	–	Undefined	Read undefined. Write as zero
[13:12]	–	–	Undefined	Read undefined. Write as zero
[11]	–	–	Undefined	Read undefined. Write as zero
[10]	–	–	Undefined	Read undefined. Write as zero
[9]	RXE	R/W	0y1	UART receive enable 0y0: Disable 0y1: Enable
[8]	TXE	R/W	0y1	UART transmit enable 0y0: Disable 0y1: Enable
[7]	–	R/W	0y0	Read undefined. Write as zero.
[6:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero
[1]	–	–	Undefined	Read undefined. Write as zero
[0]	UARTEN	R/W	0y0	UART enable 0y0: Disable 0y1: Enable

28. UART3CR (UART3 Control Register)

Address = (0x4001_C000) + 0x0030

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15]	–	–	Undefined	Read undefined. Write as zero.
[14]	–	–	Undefined	Read undefined. Write as zero.
[13:12]	–	–	Undefined	Read undefined. Write as zero
[11]	–	–	Undefined	Read undefined. Write as zero.
[10]	–	–	Undefined	Read undefined. Write as zero.
[9]	RXE	R/W	0y1	UART receive enable 0y0: Disable 0y1: Enable
[8]	TXE	R/W	0y1	UART transmit enable 0y0: Disable 0y1: Enable
[7]	–	R/W	0y0	Read undefined. Write as zero.
[6:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	UARTEN	R/W	0y0	UART enable 0y0: Disable 0y1: Enable

[Explanation]

a. <CTSEn>

If this bit is set to "1," CTS hardware flow control is enabled. Data is only transmitted when the UxCTS n signal is asserted.

b. <RTSEn>

If this bit is set to "1," RTS hardware flow control is enabled. Data is only requested when there is space in the receive FIFO.

c. <RTS >

This bit is the UART transmit request (UxRTSn) modem status output signal. When the bit is programmed to "1," "0" is output.

d. <RXE>

If this bit is set to "1," the receive circuit of the UART is enabled. When the UART is disabled in the middle of reception, it completes the current receive before stopping.

e. <TXE>

If this bit is set to "1," the transmit circuit of the UART is enabled. When the UART is disabled in the middle of transmission, it completes the current transmission before stopping.

f. <UARTEN>

If this bit is set to "1," the UART circuit is enabled. If the UART is disabled in the middle of transmission or reception, it completes the current transmission or reception before stopping.

29. UART0IFLS (UART0 Interrupt FIFO Level Selection Register)

Address = (0x4001_9000) + 0x0034

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:3]	RXIFLSEL	R/W	0y010	Receive interrupt FIFO level select (1 word = 12 bits) 0y000: When the 4th word is stored in the receive FIFO 0y001: When the 8th word is stored in the receive FIFO 0y010: When the 16th word is stored in the receive FIFO 0y011: When the 24th word is stored in the receive FIFO 0y100: When the 28th word is stored in the receive FIFO 0y101~0y111: Reserved
[2:0]	TXIFLSEL	R/W	0y010	Transmit interrupt FIFO level select (1 word = 8 bits) 0y000: When the transmit FIFO is left with 4 words 0y001: When the transmit FIFO is left with 8 words 0y010: When the transmit FIFO is left with 16 words 0y011: When the transmit FIFO is left with 24 words 0y100: When the transmit FIFO is left with 28 words 0y101~0y111: Reserved

30. UART1IFLS (UART1 Interrupt FIFO Level Selection Register)

Address = (0x4001_A000) + 0x0034

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:3]	RXIFLSEL	R/W	0y010	Receive interrupt FIFO level select (1 word = 12 bits) 0y000: When the 4th word is stored in the receive FIFO 0y001: When the 8th word is stored in the receive FIFO 0y010: When the 16th word is stored in the receive FIFO 0y011: When the 24th word is stored in the receive FIFO 0y100: When the 28th word is stored in the receive FIFO 0y101~0y111: Reserved
[2:0]	TXIFLSEL	R/W	0y010	Transmit interrupt FIFO level select (1 word = 8 bits) 0y000: When the transmit FIFO is left with 4 words 0y001: When the transmit FIFO is left with 8 words 0y010: When the transmit FIFO is left with 16 words 0y011: When the transmit FIFO is left with 24 words 0y100: When the transmit FIFO is left with 28 words 0y101~0y111: Reserved

31. UART2IFLS (UART2 Interrupt FIFO Level Selection Register)

Address = (0x4001_B000) + 0x0034

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:3]	RXIFLSEL	R/W	0y010	Receive interrupt FIFO level select (1 word = 12 bits) 0y000: When the 4th word is stored in the receive FIFO 0y001: When the 8th word is stored in the receive FIFO 0y010: When the 16th word is stored in the receive FIFO 0y011: When the 24th word is stored in the receive FIFO 0y100: When the 28th word is stored in the receive FIFO 0y101~0y111: Reserved
[2:0]	TXIFLSEL	R/W	0y010	Transmit interrupt FIFO level select (1 word = 8 bits) 0y000: When the transmit FIFO is left with 4 words 0y001: When the transmit FIFO is left with 8 words 0y010: When the transmit FIFO is left with 16 words 0y011: When the transmit FIFO is left with 24 words 0y100: When the transmit FIFO is left with 28 words 0y101~0y111: Reserved

32. UART3IFLS (UART3 Interrupt FIFO Level Selection Register)

Address = (0x4001_C000) + 0x0034

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read undefined. Write as zero.
[5:3]	RXIFLSEL	R/W	0y010	Receive interrupt FIFO level select (1 word = 12 bits) 0y000: When the 4th word is stored in the receive FIFO 0y001: When the 8th word is stored in the receive FIFO 0y010: When the 16th word is stored in the receive FIFO 0y011: When the 24th word is stored in the receive FIFO 0y100: When the 28th word is stored in the receive FIFO 0y101~0y111: Reserved
[2:0]	TXIFLSEL	R/W	0y010	Transmit interrupt FIFO level select (1 word = 8 bits) 0y000: When the transmit FIFO is left with 4 words 0y001: When the transmit FIFO is left with 8 words 0y010: When the transmit FIFO is left with 16 words 0y011: When the transmit FIFO is left with 24 words 0y100: When the transmit FIFO is left with 28 words 0y101~0y111: Reserved

[Explanation]

The UARTxIFLS Register is the interrupt FIFO level select register. You can use this register to define the FIFO level that triggers the assertion of UARTTXINTR and UARTRXINTR.

The interrupts are generated based on a transition through a FIFO level rather than being based on the level. For example, after 2-word data is received, an interrupt is generated when the third-word data is stored in the receive FIFO.

33. UART0IMSC (UART0 Interrupt Enable Set/Clear Register)

Address = (0x4001_9000) + 0x0038

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIM	R/W	0y0	Overrun error interrupt enable 0y0: Do not enable 0y1: Enable
[9]	BEIM	R/W	0y0	Break error interrupt enable 0y0: Do not enable 0y1: Enable
[8]	PEIM	R/W	0y0	Parity error interrupt enable 0y0: Do not enable 0y1: Enable
[7]	FEIM	R/W	0y0	Framing error interrupt enable 0y0: Do not enable 0y1: Enable
[6]	RTIM	R/W	0y0	Receive timeout interrupt enable 0y0: Do not enable 0y1: Enable
[5]	TXIM	R/W	0y0	Transmit FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[4]	RXIM	R/W	0y0	Receive FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSMIM	R/W	0y0	U0CTSn modem interrupt enable 0y0: Do not enable 0y1: Enable
[0]	–	–	Undefined	Read undefined. Write as zero.

34. UART1IMSC (UART1 Interrupt Enable Set/Clear Register)

Address = (0x4001_A000) + 0x0038

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIM	R/W	0y0	Overrun error interrupt enable 0y0: Do not enable 0y1: Enable
[9]	BEIM	R/W	0y0	Break error interrupt enable 0y0: Do not enable 0y1: Enable
[8]	PEIM	R/W	0y0	Parity error interrupt enable 0y0: Do not enable 0y1: Enable
[7]	FEIM	R/W	0y0	Framing error interrupt enable 0y0: Do not enable 0y1: Enable
[6]	RTIM	R/W	0y0	Receive timeout interrupt enable 0y0: Do not enable 0y1: Enable
[5]	TXIM	R/W	0y0	Transmit FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[4]	RXIM	R/W	0y0	Receive FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSMIM	R/W	0y0	U1CTSn modem interrupt enable 0y0: Do not enable 0y1: Enable
[0]	–	–	Undefined	Read undefined. Write as zero.

35. UART2IMSC (UART2 Interrupt Enable Set/Clear Register)

Address = (0x4001_B000) + 0x0038

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIM	R/W	0y0	Overrun error interrupt enable 0y0: Do not enable 0y1: Enable
[9]	BEIM	R/W	0y0	Break error interrupt enable 0y0: Do not enable 0y1: Enable
[8]	PEIM	R/W	0y0	Parity error interrupt enable 0y0: Do not enable 0y1: Enable
[7]	FEIM	R/W	0y0	Framing error interrupt enable 0y0: Do not enable 0y1: Enable
[6]	RTIM	R/W	0y0	Receive timeout interrupt enable 0y0: Do not enable 0y1: Enable
[5]	TXIM	R/W	0y0	Transmit FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[4]	RXIM	R/W	0y0	Receive FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

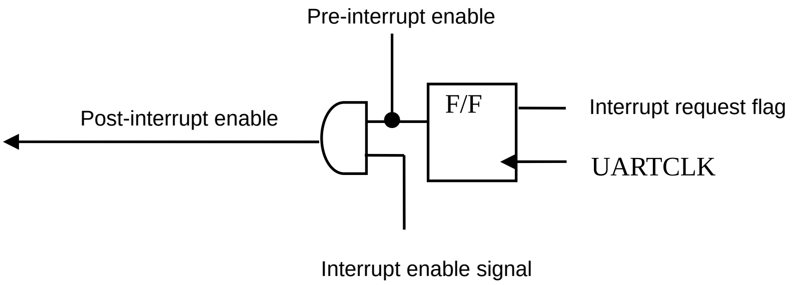
36. UART3IMSC (UART3 Interrupt Enable Set/Clear Register)

Address = (0x4001_C000) + 0x0038

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIM	R/W	0y0	Overrun error interrupt enable 0y0: Do not enable 0y1: Enable
[9]	BEIM	R/W	0y0	Break error interrupt enable 0y0: Do not enable 0y1: Enable
[8]	PEIM	R/W	0y0	Parity error interrupt enable 0y0: Do not enable 0y1: Enable
[7]	FEIM	R/W	0y0	Framing error interrupt enable 0y0: Do not enable 0y1: Enable
[6]	RTIM	R/W	0y0	Receive timeout interrupt enable 0y0: Do not enable 0y1: Enable
[5]	TXIM	R/W	0y0	Transmit FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[4]	RXIM	R/W	0y0	Receive FIFO interrupt enable 0y0: Do not enable 0y1: Enable
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

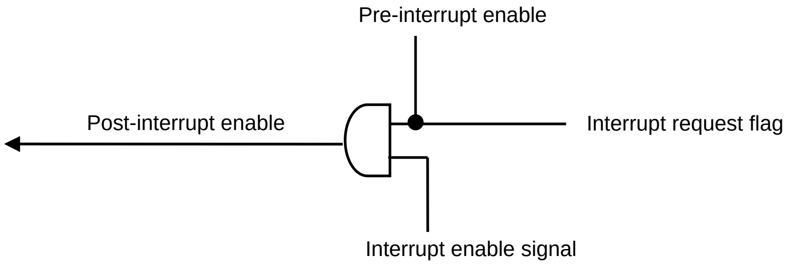
Block diagram of UART interrupt occurring

(1) Block diagram of error flag (BE, PE, FE)



* The interrupt request flag changes in real time. This is saved by the F/F.
To clear it, perform a WR to the corresponding bits of the clear-only register.

(2) Block diagram of error flag (OE)



* Only the OE overrun flag changes in real time and the interrupt is also not saved. It is cleared by performing a RD of the receive FIFO.

[Explanation]

For read, this register returns the current mask values for associated interrupts. If "1" is written in specific bits, this register enables their interrupts.

37. UART0RIS (UART0 Pre-Interrupt Enable Status Register)

Address = (0x4001_9000) + 0x003C

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OERIS	RO	0y0	Overrun error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BERIS	RO	0y0	Break error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PERIS	RO	0y0	Parity error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FERIS	RO	0y0	Framing error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTRIS	RO	0y0	Receive timeout pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXRIS	RO	0y0	Transmit pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXRIS	RO	0y0	Receive pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSRMIS	RO	Undefined	U0CTSn modem pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

38. UART1RIS (UART1 Pre-Interrupt Enable Status Register)

Address = (0x4001_A000) + 0x003C

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OERIS	RO	0y0	Overrun error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BERIS	RO	0y0	Break error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PERIS	RO	0y0	Parity error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FERIS	RO	0y0	Framing error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTRIS	RO	0y0	Receive timeout pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXRIS	RO	0y0	Transmit pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXRIS	RO	0y0	Receive pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSRMIS	RO	Undefined	U1CTSn modem pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

39. UART2RIS (UART2 Pre-Interrupt Enable Status Register)

Address = (0x4001_B000) + 0x003C

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OERIS	RO	0y0	Overrun error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BERIS	RO	0y0	Break error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PERIS	RO	0y0	Parity error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FERIS	RO	0y0	Framing error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTRIS	RO	0y0	Receive timeout pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXRIS	RO	0y0	Transmit pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXRIS	RO	0y0	Receive pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

40. UART3RIS (UART3 Pre-Interrupt Enable Status Register)

Address = (0x4001_C000) + 0x003C

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OERIS	RO	0y0	Overrun error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BERIS	RO	0y0	Break error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PERIS	RO	0y0	Parity error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FERIS	RO	0y0	Framing error pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTRIS	RO	0y0	Receive timeout pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXRIS	RO	0y0	Transmit pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXRIS	RO	0y0	Receive pre-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

[Explanation]

During read, this register returns the before-enable status of the corresponding interrupt.

41. UART0MIS (UART0 Post-Interrupt Enable Status Register)

Address = (0x4001_9000) + 0x0040

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEMIS	RO	0y0	Overrun error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BEMIS	RO	0y0	Break error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PEMIS	RO	0y0	Parity error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FEMIS	RO	0y0	Framing error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTMIS	RO	0y0	Receive timeout post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXMIS	RO	0y0	Transmit post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXMIS	RO	0y0	Receive post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSMMIS	RO	Undefined	U0CTSn modem post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

42. UART1MIS (UART1 Post-Interrupt Enable Status Register)

Address = (0x4001_A000) + 0x0040

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEMIS	RO	0y0	Overrun error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BEMIS	RO	0y0	Break error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PEMIS	RO	0y0	Parity error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FEMIS	RO	0y0	Framing error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTMIS	RO	0y0	Receive timeout post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXMIS	RO	0y0	Transmit post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXMIS	RO	0y0	Receive post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSMMIS	RO	Undefined	U1CTSn modem post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

43. UART2MIS (UART2 Post-Interrupt Enable Status Register)

Address = (0x4001_B000) + 0x0040

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEMIS	RO	0y0	Overrun error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BEMIS	RO	0y0	Break error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PEMIS	RO	0y0	Parity error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FEMIS	RO	0y0	Framing error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTMIS	RO	0y0	Receive timeout post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXMIS	RO	0y0	Transmit post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXMIS	RO	0y0	Receive post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

44. UART3MIS (UART3 Post-Interrupt Enable Status Register)

Address = (0x4001_C000) + 0x0040

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEMIS	RO	0y0	Overrun error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[9]	BEMIS	RO	0y0	Break error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[8]	PEMIS	RO	0y0	Parity error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[7]	FEMIS	RO	0y0	Framing error post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RTMIS	RO	0y0	Receive timeout post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	TXMIS	RO	0y0	Transmit post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RXMIS	RO	0y0	Receive post-interrupt enable status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to "0" when reset. The modem status interrupt bits are undefined after reset.

[Explanation]

During read, this register returns the after-enable status of the corresponding interrupt.

45. UART0ICR (UART0 Interrupt Clear Register)

Address = (0x4001_9000) + 0x0044

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIC	WO	Undefined	Overrun error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[9]	BEIC	WO	Undefined	Break error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[8]	PEIC	WO	Undefined	Parity error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[7]	FEIC	WO	Undefined	Framing error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[6]	RTIC	WO	Undefined	Receive timeout interrupt clear 0y0: Disabled 0y1: Interrupt clear
[5]	TXIC	WO	Undefined	Transmit interrupt clear 0y0: Disabled 0y1: Interrupt clear
[4]	RXIC	WO	Undefined	Receive interrupt clear 0y0: Disabled 0y1: Interrupt clear
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSMIC	WO	Undefined	U0CTSn modem interrupt clear 0y0: Disabled 0y1: Interrupt clear
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) The UARTxICR Register is the interrupt clear register and is write-only. On a write of a '1', the corresponding interrupt is cleared. A write of a '0' has no effect.

46. UART1ICR (UART1 Interrupt Clear Register)

Address = (0x4001_A000) + 0x0044

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIC	WO	Undefined	Overrun error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[9]	BEIC	WO	Undefined	Break error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[8]	PEIC	WO	Undefined	Parity error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[7]	FEIC	WO	Undefined	Framing error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[6]	RTIC	WO	Undefined	Receive timeout interrupt clear 0y0: Disabled 0y1: Interrupt clear
[5]	TXIC	WO	Undefined	Transmit interrupt clear 0y0: Disabled 0y1: Interrupt clear
[4]	RXIC	WO	Undefined	Receive interrupt clear 0y0: Disabled 0y1: Interrupt clear
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	CTSMIC	WO	Undefined	U1CTSn modem interrupt clear 0y0: Disabled 0y1: Interrupt clear
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) The UARTxICR Register is the interrupt clear register and is write-only. On a write of a '1', the corresponding interrupt is cleared. A write of a '0' has no effect.

47. UART2ICR (UART2 Interrupt Clear Register)

Address = (0x4001_B000) + 0x0044

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIC	WO	Undefined	Overrun error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[9]	BEIC	WO	Undefined	Break error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[8]	PEIC	WO	Undefined	Parity error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[7]	FEIC	WO	Undefined	Framing error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[6]	RTIC	WO	Undefined	Receive timeout interrupt clear 0y0: Disabled 0y1: Interrupt clear
[5]	TXIC	WO	Undefined	Transmit interrupt clear 0y0: Disabled 0y1: Interrupt clear
[4]	RXIC	WO	Undefined	Receive interrupt clear 0y0: Disabled 0y1: Interrupt clear
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) The UARTxICR Register is the interrupt clear register and is write-only. On a write of a '1', the corresponding interrupt is cleared. A write of a '0' has no effect.

48. UART3ICR (UART3 Interrupt Clear Register)

Address = (0x4001_C000) + 0x0044

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read undefined. Write as zero.
[10]	OEIC	WO	Undefined	Overrun error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[9]	BEIC	WO	Undefined	Break error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[8]	PEIC	WO	Undefined	Parity error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[7]	FEIC	WO	Undefined	Framing error interrupt clear 0y0: Disabled 0y1: Interrupt clear
[6]	RTIC	WO	Undefined	Receive timeout interrupt clear 0y0: Disabled 0y1: Interrupt clear
[5]	TXIC	WO	Undefined	Transmit interrupt clear 0y0: Disabled 0y1: Interrupt clear
[4]	RXIC	WO	Undefined	Receive interrupt clear 0y0: Disabled 0y1: Interrupt clear
[3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

Note) The UARTxICR Register is the interrupt clear register and is write-only. On a write of a '1', the corresponding interrupt is cleared. A write of a '0' has no effect.

[Explanation]

On a write of a '1', the corresponding interrupt is cleared. A write of a '0' has no effect.

49. UART0DMACR (UART0 DMA Control Register)

Address = (0x4001_9000) + 0x0048

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	–	Read undefined. Write as zero.
[2]	DMAONERR	R/W	0y0	DMA on error 0y1: Error asserted 0y0: No error asserted
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA enable 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA enable 0y0: Disable 0y1: Enable

Note 1) For example, if 19 characters have to be received and the watermark level is programmed to be four, the DMA controller then transfers four bursts of four characters and three single transfers to complete the stream.

Note 2) To transfer data in a transmit or receive FIFO using the DMAC, set the bus width to 8 bits.

50. UART1DMACR (UART01DMA Control Register)

Address = (0x4001_A000) + 0x0048

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	–	Read undefined. Write as zero.
[2]	DMAONERR	R/W	0y0	DMA on error 0y1: Error asserted 0y0: No error asserted
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA enable 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA enable 0y0: Disable 0y1: Enable

51. UART2DMACR (UART02DMA Control Register)

Address = (0x4001 B000) + 0x0048

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	–	Read undefined. Write as zero.
[2]	DMAONERR	R/W	0y0	DMA on error 0y1: Error asserted 0y0: No error asserted
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA enable 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA enable 0y0: Disable 0y1: Enable

52. UART3DMACR (UART03DMA Control Register)

Address = (0x4001 C000) + 0x0048

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	–	Read undefined. Write as zero.
[2]	DMAONERR	R/W	0y0	DMA on error 0y1: Error asserted 0y0: No error asserted
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA enable 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA enable 0y0: Disable 0y1: Enable

[Explanation]

a. <DMAONERR>

If this bit is set to "1," the DMA receive request outputs, UARTxRXDMASREQ or UARTxRXDMABREQ, are disabled when the UART error interrupt is asserted.

3.14 I^2C

3.14.1 Overview

This module operates in I^2C bus mode and is compliant with the I^2C bus standard (PHILIPS specification) (*1).

This module has the following features:

- Contains two channels (Ch0 and Ch1).
- Allows selection between master and slave transmission/reception.
- Allows selection between transmission and reception.
- Supports multimaster mode. (Arbitration and recognition of clock synchronization are possible.)
- Supports standard communication mode and fast communication mode:
  Maximum communication baud rate in master mode: 94.74 KHz/360.00 KHz@f_{PCLK}=144 MHz
  Maximum communication baud rate in slave mode: 100.00 KHz/400.00 KHz@f_{PCLK}=144 MHz
- Supports the addressing format of 7 bits only.
- Supports transfer data sizes of 1 to 8 bits.
- Supports one level of transfer (send or receive) completion interrupt
- Can enable or disable interrupts. (Interrupt source for the I2C ch0: I2CINT0, Interrupt source for the I2C ch1: I2CINT1)

Also, this module supports a unique free data format.

(*1) Complies with the fast mode. However, the following are excluded.

Note) This module does not support part of the I2C bus standard.

I^2C bus standard: item	I^2C standard	This IP
Support for the standard mode (- 100 KHz)	Required	Supported
Support for the fast mode (- 400 KHz)	Required	Supported
Support for the Hs (High speed) mode (- 3.4 Mbps)	Required	Not supported
Support for 7-bit addressing	Required	Supported
Support for 10-bit addressing	Required	Not supported
Start byte	Required	Not supported
Support for noise canceller	Required	Supported (digital)
Support for slope control	Required	Not supported
Support for I/O during power OFF	Required	Not supported
Schmitt (VIH/VIL)	VDD x 0.3/VDD x 0.7	Supported
Output current @ VOL = 0.4 V, VDD > 2 V	3mA	Supported

This module is connected to an external device via the PB4 (I2C0SDA) and PB5 (I2C0SCL) of the I^2C ch0, and via the PB6 (I2C1SDA) and PB7 (I2C1SCL) of the I^2C ch1.

Settings for each pin are shown below.

	GPIOBFR1 <PB7F1 to PB0F1>	GPIOBFR2 <PB7F2 to PB0F2>	GPIOBOPD <PB7ODE to PB0ODE>
I^2C ch0	- - 11 - - - -	- - 00 - - - -	- - 1 1 - - - -
I^2C ch1	1 1 - - - - - -	0 0 - - - - - -	1 1 - - - - - -

(Note) - : Don't care

This LSI contains the I2C with two channels (Ch0 and Ch1). Since 2 channels operate in the same way, only ch0 is described in the following sections unless otherwise specified.

3.14.1.1 I^{2}C Bus Mode

This I^{2}C bus is used to connect a device through the I2C0SDA and I2C0SCL and allows communication with multiple devices.

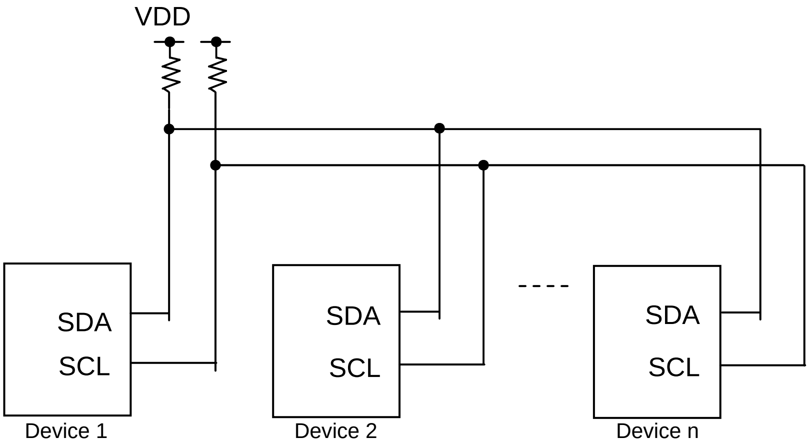


Figure 3.14.1 Device connection

This module operates as a master/slave device on the I^{2}C bus. The master device drives the serial clock line (SCL) of the bus, sends 8-bit addresses, and sends or receives data of 1 to 8 bits. The slave device receives 8-bit addresses and sends or receives serial data of 1 to 8 bits in synchronization with the serial clock on the bus.

The device that operated as a receiver can output an acknowledge signal after reception of serial data and the device that operated as a transmitter can receive that acknowledge signal, regardless of whether the device is the master or slave. The master device can output a clock for the acknowledge signal.

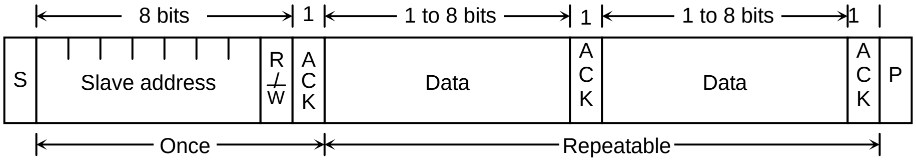
In multimaster mode in which multiple masters exist on the same bus, synchronization of serial clocks and arbitration lost to maintain consistency of serial data are supported.

3.14.2 Data Format in I^{2}C Bus Mode

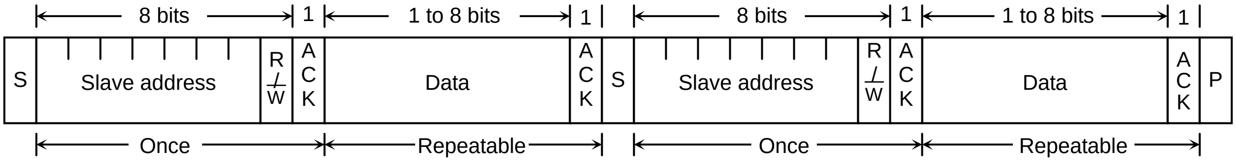
The following figure shows the data format in I^{2}C bus mode:

3.14.2.1 Addressing Format

(a) Addressing format



(b) Addressing format (with restart)



S: Start condition
R/W: Direction bit
ACK: Acknowledge bit
P: Stop condition

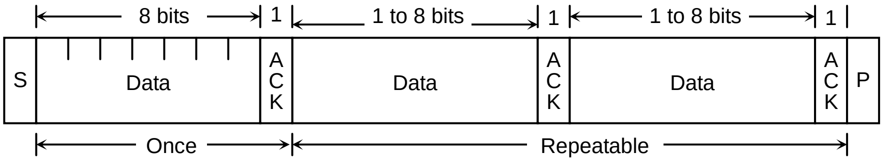
Figure 3.14.2 Data format in I^{2}C bus mode

3.14.2.2 Free Data Format

The free data format is the format in which one master device and one slave device communicate with each other.

In the free data format, a slave address and a direction bit are treated as data.

(a) Free data format (transfer format in which a master device transfers data to a slave device)



S: Start condition
R/W: Direction bit
ACK: Acknowledge bit
P: Stop condition

Figure 3.14.3 Free data format in I^{2}C bus mode

3.14.3 Block Diagram

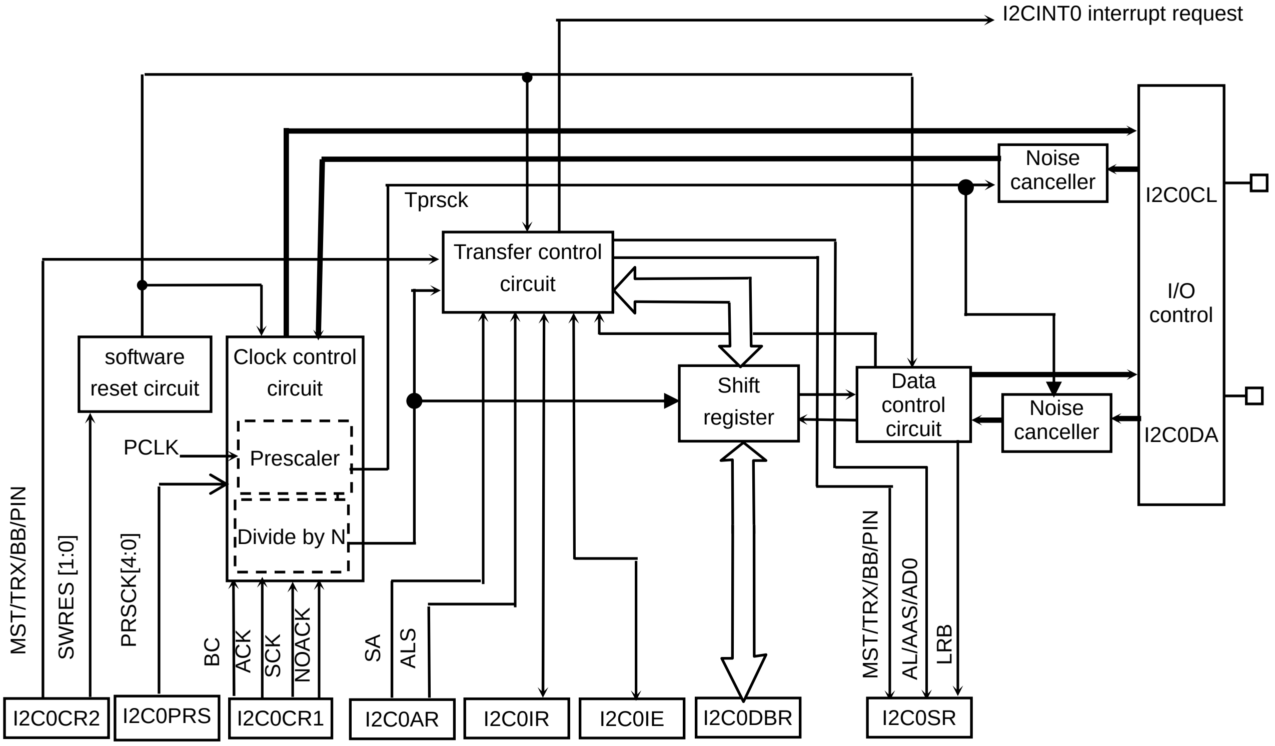


Figure 3.14.4 I^{2}C Channel 0

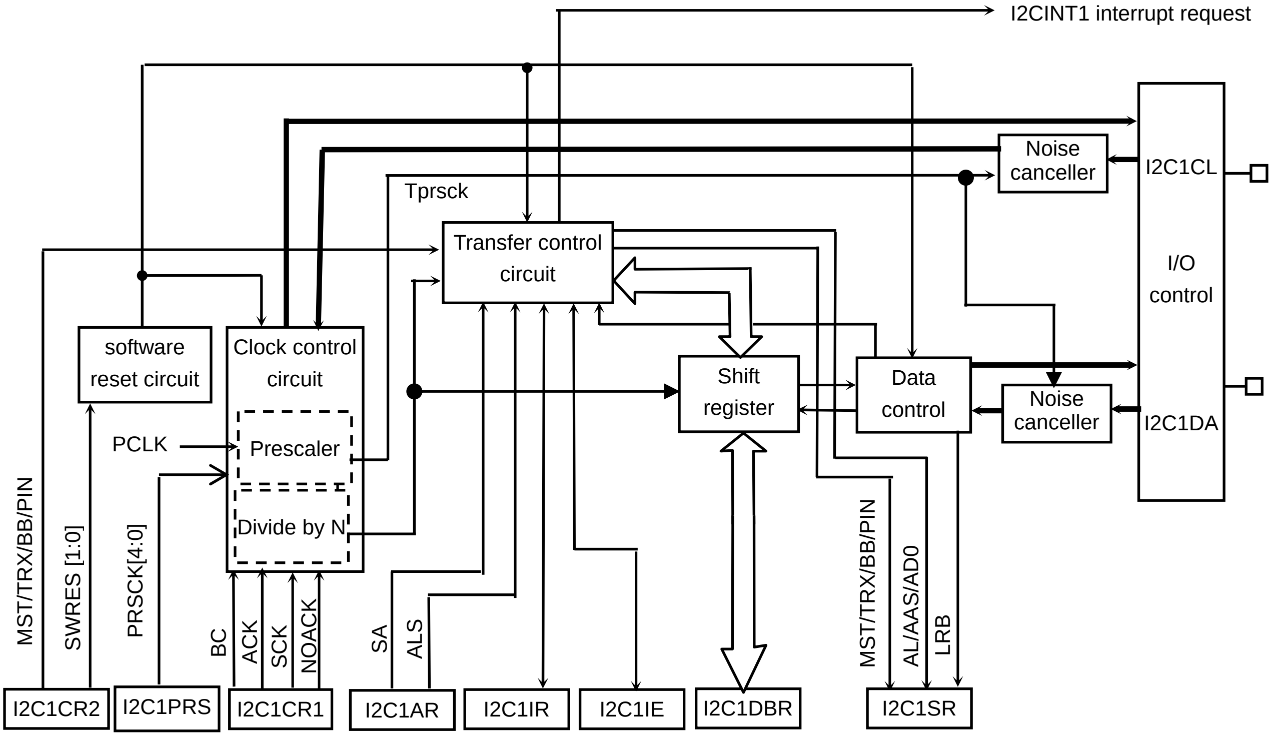


Figure3.14.5 I^{2}C Channel 1

3.14.4 Operation Description

3.14.4.1 Data Transfer Procedures in I^{2}C Bus Mode

1. Initializing the device

   Check that the I2C0SDA and I2C0SCL pins are in HIGH (bus free) state, and then set "1" for I2C0CR2<I2CM> and enable the I^{2}C.

   Next, write "1" in I2C0CR1<ACK>, "0" in I2C0CR1<NOACK>, and "000" in I2C0CR1<BC>, and "count" the clocks for an acknowledge signal. After that, set detection of a slave address matching and a general call to "Enable" and the data length to "8 bits." Also, set t_{HIGH} and t_{LOW} by using I2C0CR1<SCK>.

   Set a slave address for I2C0AR<SA> and "0" for I2C0AR<ALS>, and then set an addressing format.

   Lastly, set "0" for I2C0CR2<MST>, I2C0CR2<TRX>, and I2C0CR2<BB>, "1" for I2C0CR2<PIN>, and "00" for I2C0CR2<SWRES[1:0]>. Set the initial state to the slave receiver mode.

Note) Initialization of the I^{2}C should be completed after all the devices that connect to the bus have been initialized and within the specific time in which no devices generate a start condition. When this rule cannot be complied with, another device may start transfer before initialization of the I^{2}C is completed, which causes failure to receive data.

(Sample program) Initializing the device

```
while ( ( (GPIOBDATA) & 0x00000030 ) != 0x00000030 ) { }   ; Check if the external pin is in the "H" state.
                                                            (Check the bus free state)
(I2C0CR2) ←   0x00000018                                    ; Enable the I2C.
(I2C0CR1) ←   0x00000016                                    ; Acknowledgement mode. Set I2C0CR1<SCK> to
                                                            "110."
(I2C0AR)  ←   0x000000A0                                    ; Set the slave address to "1010000" as an
                                                            addressing format.
(I2C0CR2) ←   0x00000018                                    ; Set the slave receiver mode
```

2. Generating a start condition and slave address

Check the bus free state (I2C0SR<BB>="0").

Set I2C0CR1<ACK> to "1" and write data on the slave address that sends data to I2C0DBR and a direction bit. When "1" is written in I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB>, and I2C0CR2<PIN>, a start condition, slave address, and direction bit are output to the bus. When the start condition has been output, it will take the time of t_{HIGH} before the I2C0SCL pin output falls.

After that, an I2CINT0 interrupt request is generated at the falling edge of the 9th clock of I2C0SCL, and I2C0SR<PIN> is cleared to "0." I2C0SCL is held "Low" while I2C0SR<PIN> is set to "0." Only when an acknowledge signal is returned from the slave device, the hardware changes I2C0SR<TRX> according to the direction bit at the timing when the I2CINT0 interrupt request is generated.

Note 1) Before writing a value in I2C0DBR to output a slave address, detect the bus free state in software.

Note 2) Another master device may transfer data during the period from when a slave address is written to when a start condition is generated. Therefore, check the bus free state again in the software within 98.0 μs (the shortest transfer period defined in the I^2C bus standard for the standard mode) or 23.7 μs (the shortest transfer period defined in the I^2C bus standard for the fast mode) after the slave address to which data is output has been written. A start condition should be generated only when the bus free state is confirmed.

(Sample program) Generating a start condition

```
while ( ( (I2C0SR) & 0x00000020 ) != 0x00000000 ) { }      ; Check the bus free state

(I2C0DBR) ←    0x000000CB                                    ; Transmit slave address: 0x65, Direction bit: "1"
(I2C0CR2) ←    0x000000F8                                    ; Set I2C0CR2<MST>, <TRX>, <BB>, and <PIN> to "1"
```

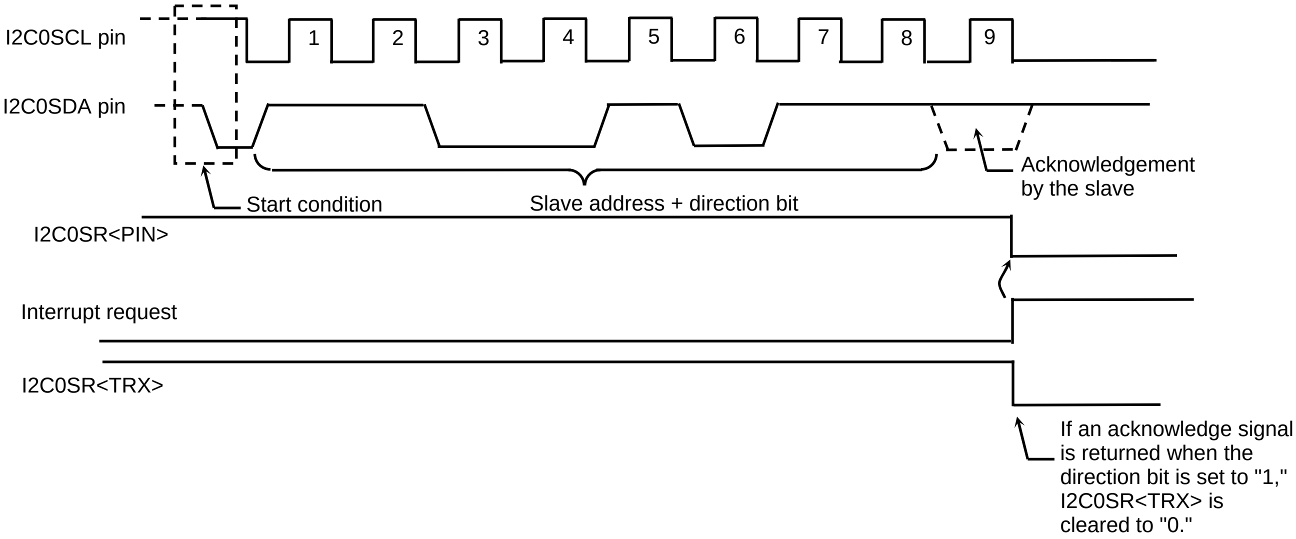

I2C0SCL pin
1
2
3
4
5
6
7
8
9
I2C0SDA pin
Start condition
Slave address + direction bit
Acknowledgement by the slave
I2C0SR<PIN>
Interrupt request
I2C0SR<TRX>
If an acknowledge signal is returned when the direction bit is set to "1," I2C0SR<TRX> is cleared to "0."

Figure 3.14.6 Generating a start condition and slave address

3. Transferring data of 1 word

Check the status of I2C0SR<MST> during 1-word transfer completion interrupt processing to determine whether the module is in the master mode or slave mode.

(1) When I2C0SR<MST> is set to "1" (master mode)

Check the status of I2C0SR<TRX> to determine if it is a transmitter or receiver.

a. When I2C0SR<TRX> is set to "1" (transmitter mode)

Check the status of the acknowledge signal from the receiver by using the I2C0SR<LRB> flag. When I2C0SR<LRB> is set to "0," write the send data in I2C0DBR because the receiver requests the next data.

However, if you want to change the number of transferred bits, set I2C0CR1<BC> again and set I2C0CR1<ACK> to "1," and then write the send data in I2C0DBR.

When the data has been written, I2C0SR<PIN> is set to "1," serial clocks are generated from I2C0SCL, and data is sent from I2C0SDA.

Once transmission has completed, an I2CINT0 interrupt request is generated, I2C0SR<PIN> is set to "0," and I2C0SCL is held "Low." If multiple words need to be transferred, repeat the procedures starting from the step of checking the I2C0SR<LRB> flag status described above.

When I2C0SR<LRB> is set to "1," perform the processing for generating a stop condition (which is described later) to finish the data transfer because the receiver does not request data.

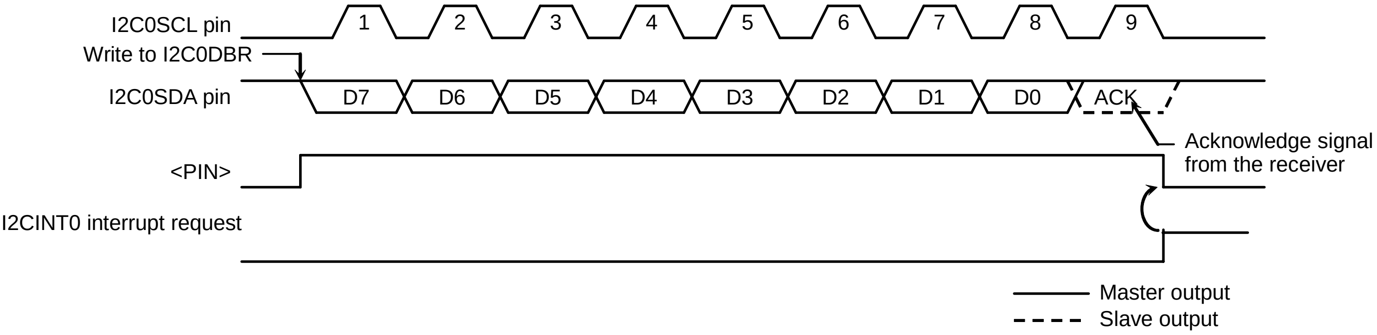


Figure 3.14.7 When I2C0CR1<BC>="000" and I2C0CR1<ACK>="1"

b. When I2C0SR<TRX> is set to "0" (receiver mode)

When dummy data (0x00) is written in I2C0DBR or "1" is set for I2C0CR2<PIN>, a transfer clock of 1 word and an acknowledge signal are output.

When an I2CINT0 interrupt request, which indicates completion of reception, has been generated, read the received data from I2C0DBR.

However, if you want to change the number of received data bits, set I2C0CR1<BC> again and set I2C0CR1<ACK> to "1," and then write dummy data (0x00) or set "1" for I2C0CR2<PIN>.

(Read data is unfixed immediately after a slave address has been sent.)

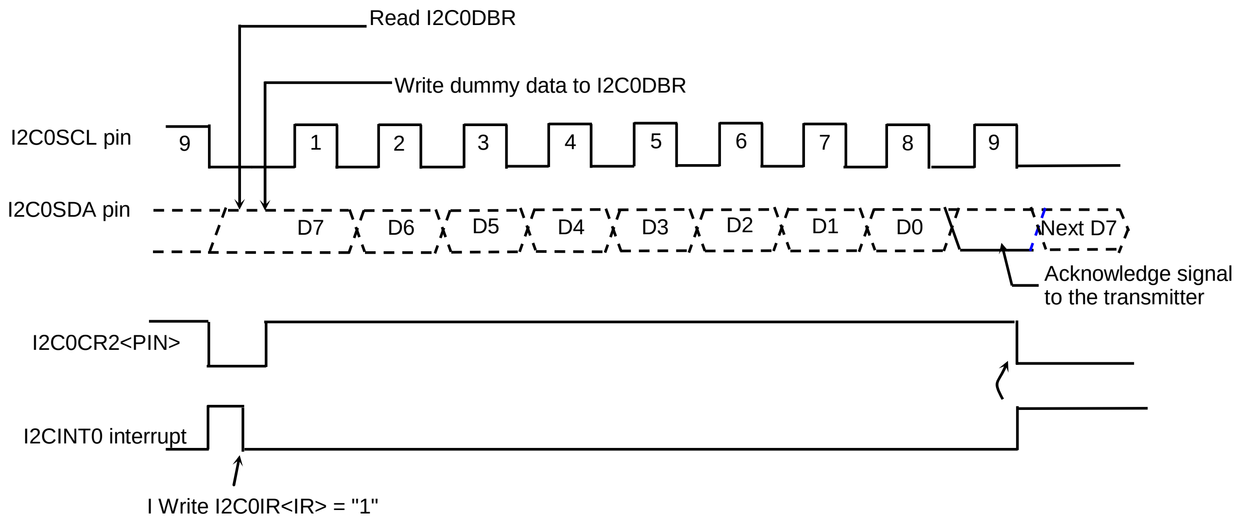


Figure 3.14.8 When I2C0CR1<BC> = "000" and I2C0CR1<ACK> = "1"

c. When I2C0SR<TRX> is set to "0" (to receive the last word)

Perform pseudo communication, which does not output acknowledge signals, to determine if the word is the last.

The following describes its flow:

To finish data transmission of the transmitter, perform the following processing before receiving the last data.

1. Read the received data from I2C0DBR.
2. Clear I2C0CR1<ACK> to "0" and set I2C0CR1<BC> to "000."
3. To set "1" for I2C0CR2<PIN>, write dummy data (0x00) in I2C0DBR.

When I2C0CR2<PIN> is set to "1," data of 1 word, which will not cause generation of acknowledge clocks, is transferred. When data of 1 word has been transferred, perform the following:

1. Read the received data from I2C0DBR.
2. Clear I2C0CR1<ACK> to "0" and set I2C0CR1<BC> to "001." (Issue a negative acknowledge)
3. Set dummy data (0x00) for I2C0DBR, or set "1" for I2C0CR2<PIN>.

When I2C0CR2<PIN> is set to "1," data of 1 bit is transferred. The SDA line on the bus is held "High" because the master operates as the receiver at this time. The transmitter receives this "H" level as a negative acknowledge signal, with which the receiver can notify the transmitter of completion of transmission. In this 1-bit transfer reception end interrupt processing, a stop condition is generated to finish data transfer.

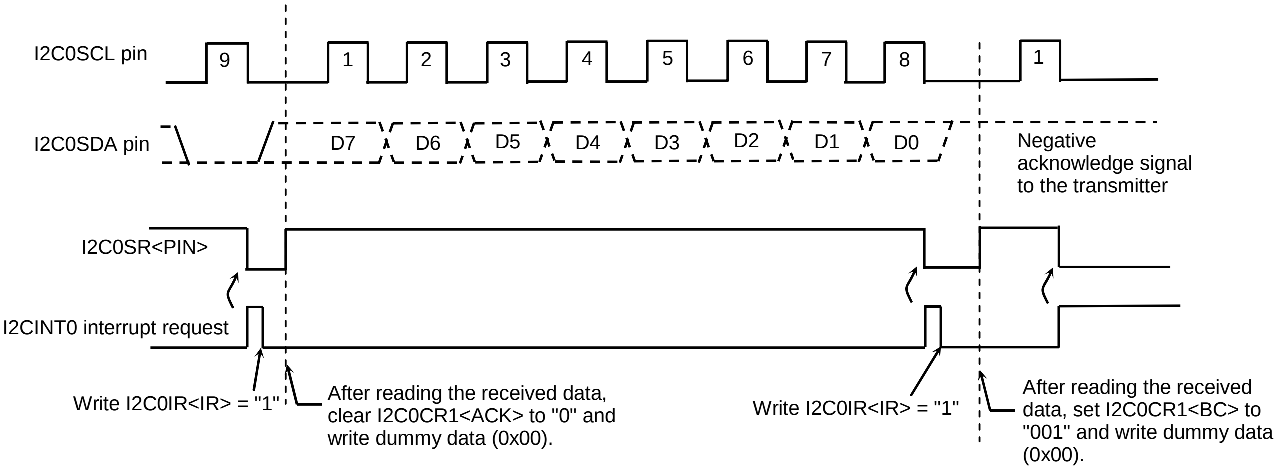

I2C0SCL pin
9
1
2
3
4
5
6
7
8
1
I2C0SDA pin
D7
D6
D5
D4
D3
D2
D1
D0
Negative acknowledge signal to the transmitter
I2C0SR<PIN>
I2CINT0 interrupt request
Write I2C0IR<IR> = "1"
After reading the received data, clear I2C0CR1<ACK> to "0" and write dummy data (0x00).
Write I2C0IR<IR> = "1"
After reading the received data, set I2C0CR1<BC> to "001" and write dummy data (0x00).

Figure 3.14.9 Processing to finish data transmission in master receiver mode

(2) When I2C0SR<MST> is set to "0" (slave mode)

In the slave mode, perform the processing to be done in the normal slave mode or the processing to be done when the I^2C loses arbitration and goes into slave mode.

In the slave mode, an I2CINT0 interrupt request is generated in the following cases:

- After an acknowledge signal is output when I2C0CR1<NOACK> is set to "0" and the received slave address matches with the slave address set in I2C0AR<SA>
- After an acknowledge signal is output when I2C0CR1<NOACK> is set to "0" and a general call is received
- When the slave addresses match, or when the data transfer after reception of a general call is complete

When the I^2C operates in the master mode and loses arbitration, it is changed to the slave mode. An I2CINT0 interrupt request is generated when transfer of the word in which arbitration was lost is complete. Table 3.14.1 shows I2CINT0 interrupt request generation after arbitration is lost, and the operations of I2C0SR<PIN>.

Table 3.14.1 I2CINT0 interrupt request and I2C0SR<PIN> operations when arbitration is lost

	When arbitration is lost while a slave address is being transmitted in master mode	When arbitration is lost while data is being transmitted in master transmitter mode
I2CINT0 Interrupt request	The I2CINT0 interrupt request is generated at completion of word transfer	
I2C0SR<PIN>	I2C0SR<PIN> is cleared to "0."	

When an I2CINT0 interrupt request is generated, I2C0SR<PIN> is reset to "0" and I2C0SCL is held "Low." When data is written in I2C0DBR or "1" is set for I2C0CR2<PIN>, I2C0SCL is released after t_{LOW}.

Test I2C0SR<AL>, I2C0SR<TRX>, I2C0SR<AAS>, and I2C0SR<AD0> to branch the processing. Table 3.14.2 shows the states in the slave mode and required actions.

Table 3.14.2 Processing in the slave mode

I2C0SR <TRX>	I2C0SR <AL>	I2C0SR <AAS>	I2C0SR <AD0>	State	Process
1	1	1	0	The serial bus interface circuit loses arbitration when sending a slave address, and receives the slave address of the serial bus interface circuit sent from another master, whose direction bit is set to "1."	Set the number of bits in 1 word in I2C0CR1<BC> and write the data to be sent to I2C0DBR.
	0	1	0	In the slave receiver mode, the circuit receives the slave address of the serial bus interface circuit sent from the master, whose direction bit is set to "1."	
		0	0	In the slave transmitter mode, 1-word data transfer is finished.	When I2C0SR<LRB> is tested and it is found that I2C0SR<LRB> is set to "1," the receiver does not request the next data. Set I2C0CR2<PIN> to "1," reset I2C0CR2<TRX> to "0," and then release the bus. When I2C0SR<LRB> is reset to "0," the receiver requests the next data. Set the number of bits in 1 word in I2C0CR1<BC>, and write the data to be sent to I2C0DBR.
0	1	1	1/0	The serial bus interface circuit loses arbitration when sending a slave address, and receives the slave address of the serial bus interface circuit sent from another master, whose direction bit is set to "0," or a general call.	To set "1" for I2C0SR<PIN>, write dummy data (0x00) in I2C0DBR. Or, write "1" in I2C0CR2<PIN>
		0	0	The serial bus interface circuit loses arbitration when sending a slave address or data, and the data transfer of that word is finished.	The serial bus interface circuit has been set to the slave mode. Clear I2C0SR<AL> to "0," and to set I2C0SR<PIN> to "1," write dummy data (0x00) in I2C0DBR.
	0	1	1/0	In the slave receiver mode, the circuit receives the slave address of the serial bus interface circuit sent from the master, whose direction bit is set to "0," or a general call.	To set "1" for I2C0SR<PIN>, write dummy data (0x00) in I2C0DBR. Or, write "1" in I2C0CR2<PIN>
		0	1/0	In the slave receiver mode, 1-word data reception is finished.	Set the number of bits in 1 word in I2C0CR1<BC> and read the received data from I2C0DBR and write dummy data (0x00) in it.

Note) If I2C0AR is set to 0x00 in slave mode, it is determined that the slave addresses are matched and I2C0SR<TRX> is set to "1" when the START byte (0x01), which is defined in the I^{2}C bus standard, is received. Do not set 0x00 for I2C0AR<SA>.

4. Generating a stop condition

If "1" is written in I2C0CR2<MST>, I2C0CR2<TRX>, and I2C0CR2<PIN> and "0" is written in I2C0CR2<BB> when I2C0SR<BB> is set to "1," the sequence for outputting a stop condition onto the bus is started. Do not rewrite data in I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB>, and I2C0CR2<PIN> before a stop condition is generated on the bus.

If the I2C0SCL line is held by another device when a stop condition is generated, a stop condition is generated after the I2C0SCL line is released.

The time of t_{HIGH} is required after the I2C0SCL line is released until a stop condition is generated.

(Sample program) Generating a stop condition

```
(I2C0CR2) ←    0x000000D8                                  ; Set I2C0CR2<MST>,<TRX>, and <PIN> to
                                                            "1" and I2C0CR2<BB> to "0."
while ( ( (I2C0SR) & 0x00000020 ) != 0x00000000 ) { }      ; Check the bus free state
```

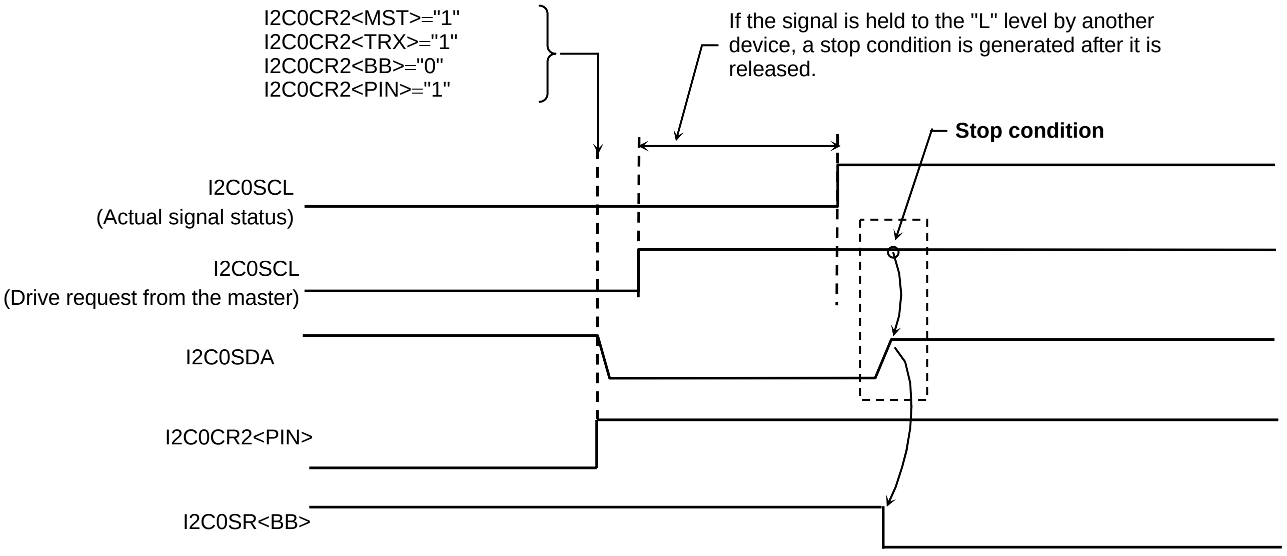


Figure 3.14.5 Generating a stop condition

5. Procedures for restart

Restart procedures are performed to change the transfer direction without stopping data transfer from a master device to a slave device. The following describes the procedures for generating restart condition:

First, write "0" in I2C0CR2<MST>, I2C0CR2<TRX>, and I2C0CR2<BB>, and "1" in I2C0CR2<PIN>, which causes I2C0SDA to be held "High" and I2C0SCL to be released.

Because this status is not a stop condition, the other devices detect that the bus remains busy.

Check the status of I2C0SR<BB> and wait until it becomes "0" to confirm that I2C0SCL of the I2C is released.

Next, check the status of I2C0SR<LRB> and wait until it becomes "1" to confirm that the I2C0SCL line on the bus is not held "Low."

After confirming that the bus is released by performing the above procedures, generate a start condition by following the procedures described in the above 2 "Generating a start condition and slave address."

Note that the software must wait for at least 4.7 μs (according to the standard mode I^2C bus standard) or 0.6 μs (according to the fast mode I^2C bus standard) to satisfy the setup time for the restart condition.

Note) When a master device operates as a receiver, data transmission from the slave device operating as a transmitter must be finished before a restart condition is generated. To finish the data transmission, a negative acknowledge signal at the "H" level should be received by the slave device. In this case, "1" is set for I2C0SR<LRB> before a restart condition is generated. Therefore, even if it is checked that I2C0SR<LRB> is set to "1" during the procedures for generating a restart condition, the rise of the SCL line cannot be checked. To check the status of the I2C0SCL line, read the status of the port. (Before reading the status of the port, GPIOBFR1 and GPIODOR must be set to "1" and "0," respectively.)

(Sample program) Generating a restart condition

```
(I2C0CR2) ←    0x00000018                                   ; Set I2C0CR2<MST>,<TRX>, and <BB> to
                                                              "0" and I2C0CR2<PIN> to "1."
while ( ( (I2C0SR) & 0x00000020 ) != 0x00000000 ) { }      ; Wait until I2C0SR<BB> is set to "0."
while ( ( (I2C0SR) & 0x00000001 ) != 0x00000001 ) { }      ; Wait until I2C0SR<LRB> is set to "1."

while ( ( (GPIOBDATA) & 0x00000010 ) != 0x00000010 ) { }   ; Wait until the I2C0SCL line is set to "1."

                                                            ; Wait processing by the software

(I2C0CR2) ←    0x000000F8                                   ; Set I2C0CR2<MST>,<TRX>,<BB>, and
                                                            <PIN> to "1."
```

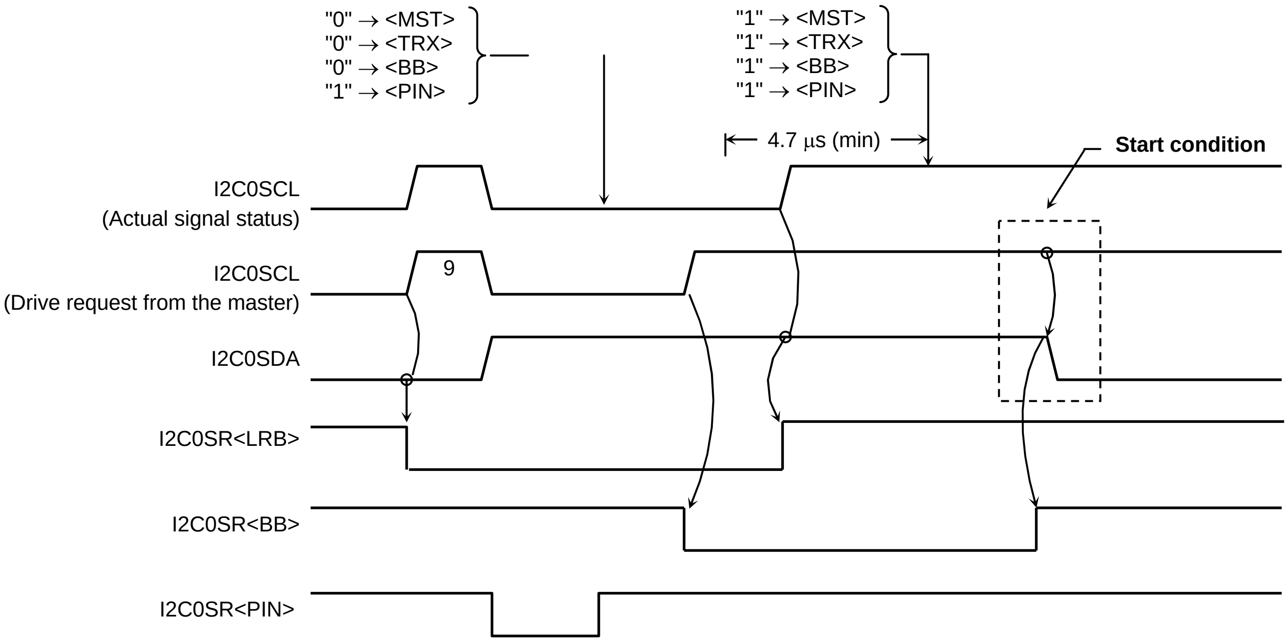


Figure 3.14.6 Timing chart to generate a restart condition

Note) Do not write "0" in <MST> when <MST> is set to "0." (This disables restart)

3.14.5 Explanation of the Register

The following lists the SFRs:

base address = 0x4001_3000

Register Name	Address (base+)	Description
I2C0CR1	0x0000	I^2C0 Control Register 1
I2C0DBR	0x0004	I^2C0 Data Buffer Register
I2C0AR	0x0008	I^2C0 (Slave) Address Register
I2C0CR2	0x000C	I^2C0 Control Register 2
I2C0SR		I^2C0 Status Register
I2C0PRS	0x0010	I^2C0 Prescaler Clock Set Register
I2C0IE	0x0014	I^2C0 Interrupt Enable Register
I2C0IR	0x0018	I^2C0 Interrupt Register

base address = 0x4001_4000

Register Name	Address (base+)	Description
I2C1CR1	0x0000	I^2C1 Control Register 1
I2C1DBR	0x0004	I^2C1 Data Buffer Register
I2C1AR	0x0008	I^2C1 (Slave) Address Register
I2C1CR2	0x000C	I^2C1 Control Register 2
I2C1SR		I^2C1 Status Register
I2C1PRS	0x0010	I^2C1 Prescaler Clock Set Register
I2C1IE	0x0014	I^2C1 Interrupt Enable Register
I2C1IR	0x0018	I^2C1 Interrupt Register

1. I2C0CR1 (I^{2}C0 Control Register 1)

Address = (0x4001_3000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:5]	BC[2:0]	R/W	0y000	Select the number of transfer bits 0y000: 8 bits 0y100: 4 bits 0y001: 1 bit 0y101: 5 bits 0y010: 2 bits 0y110: 6 bits 0y011: 3 bits 0y111: 7 bits
[4]	ACK	R/W	0y0	Selection of clock generation and recognition for an acknowledge signal 0y0: Not available 0y1: Available
[3]	NOACK	R/W	0y0	Detection of slave address matching and general call 0y0: Detect. 0y1: Do not detect.
[2:0]	SCK[2:0]	R/W	0y000	Select a serial clock frequency 0y000: n=0 0y100: n=4 0y001: n=1 0y101: n=5 0y010: n=2 0y110: n=6 0y011: n=3 0y111: n=7

[Explanation]

a. <BC[2:0]>

These bits are used to select the number of transfer bits.

0y000: 8 bits 0y100: 4 bits

0y001: 1 bit 0y101: 5 bits

0y010: 2 bits 0y110: 6 bits

0y011: 3 bits 0y111: 7 bits

b. <ACK>

This bit is used to select clock generation and recognition for an acknowledge signal.

0y0: Not available

0y1: Available

c. <NOACK>

In slave operation, this bit is used to make the selection of whether or not to detect slave address matching and a general call.

0y0: Detect.

0y1: Do not detect.

When I2C0AR<ALS> = 1, this bit has no meaning.

When <NOACK> = 0, detection of slave address matching and a general call is tried. When a slave address matching or a general call is detected, the SDA line is held "Low" for the duration of the 9th clock (acknowledge clock) output from the master and acknowledgment is returned.

When <NOACK> = 1, detection of slave address matching and a general call is not performed. At the time of a slave address matching or a general call, the SDA line is released (held High) for the duration of the 9th clock (acknowledge clock) output from the master and acknowledgment is not returned.

d. <SCK[2:0]>

This bit is used to set the rate of the serial clock output from the master.

The prescaler clock divided according to I2C0PRS<PRSCK[4:0]> is used as the reference clock of serial clock generation. The prescaler clock is further divided according to I2C0CR1<SCK[2:0]> and used as the serial clock. The default setting of the prescaler clock is "divide by 1" (= f_{PCLK}).

Note) Refer to 3.14.6.3 "Serial Clock" for division setting by using I2C0CR1<SCK[2:0]>.

Writing to this register must be done before a start condition is generated or after a stop condition was generated or between the instant when an address or data transfer interrupt occurs and the instant when an internal interrupt is released. Do not write during transfer of an address or data.

2. I2C0DBR (I2C0 Data Buffer Register)

Address = (0x4001_3000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined.
[7:0]	DB[7:0]	RO	0x00	Read: Receive data is read (Note)

Address = (0x4001_3000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	DB[7:0]	WO	0x00	Write: Send data is written (Note)

Note) This register is initialized only after hardware reset. It is not initialized after software reset. (The final data is held.)

[Explanation]

a. <DB[7:0]>

These bits are used to store data for serial transfer.

When this register operates as a send device, it writes the data to be sent to DB[7:0] left justified.

When this register operates as a receive device, it stores the data received by serial transfer in DB[7:0] right justified.

As for address transmission from the master, the address of transfer target device is written to I2C0DBR<DB[7:1]> and any of the following bits is written to I2C0DBR<DB[0]> as the direction bit of transfer:
"0": Master/send – slave/receive
"1": Master/receive – slave/send

When "0" is written to all the bits of the I2C0DBR register, a general call can be sent to the bus.

In both of the send and receive modes, a write to the I2C0DBR register releases the internal interrupt after the current transfer and begins the next transfer.

I2C0DBR can be used as a read/write buffer. However, do not use it as both buffers at the same time. Also, access the register at each data transfer.

Note) When the register is set to operate as a receiver, writing data in I2C0DBR before received data is read damages the receive data.

3. I2C0AR (I2C0 (Slave) Address Register)

Address = (0x4001_3000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:1]	SA[6:0]	R/W	0y0000000	Set a slave address.
[0]	ALS	R/W	0y0	Make a selection of whether or not to set the address recognition mode. 0y0: Recognize the address (I^2C bus mode). 0y1: Do not recognize the address (free data format).

[Explanation]

a. <SA[6:0]>

These bits are used to set the address of the device that operates as a slave device (7 bits).

When slave addresses are set so as to be recognized by I2C0AR<ALS> described later, the operation at the time of data transfer is decided by a 7-bit address (plus 1 direction bit) that the master sends immediately after the start condition.

b. <ALS>

This bit is used to set the address recognition mode.

0y0: Recognize the address (I^2C bus mode).

0y1: Do not recognize the address (free data format).

When this module operates as a slave device, this register makes a selection of whether or not the 8-bit data that the master sends immediately after the start condition is recognized as a 7-bit address plus 1 direction bit.

When the 8-bit data is so recognized, this module operates in I^2C bus mode. Otherwise, it operates in free data format.

When <ALS> = 0, the device compares the 7-bit address sent from the master against the I2C0AR<SA[6:0]> setting. When the two match, the device decides whether the communication direction is send or received by the direction bit, and when I2C0CR1<NOACK> = 0, the device holds the SDA line "Low" for the duration of the acknowledge clock (9th clock) output from the master.

Thereafter, this device continues to perform transfer operation as a slave send/receive device until a stop condition or a start condition by the restart procedure appears on the bus.

If the 7-bit address and the I2C0AR<SA[6:0]> setting does not match, this device thereafter continues to release (holds High) the SDA/SCL line and does not join in the transfer operation until a stop condition or a start condition by the restart procedure appears on the bus (I^2C bus mode operation and slave address match detection).

If a 7-bit address plus 1 direction bit sent from the master is zeros at all bit positions (general call) and I2C0CR1<NOACK> = 0, the device outputs Acknowledge (Low) to operate as a slave receive device regardless of the slave address set to I2C0AR<SA[6:0]> (I^2C bus mode operation and general call detection).

When I2C0CR1<NOACK> = 1, the device neither outputs Acknowledge nor operates as a slave device even when it detects a slave address matching or a general call.

When <ALS> = 1, the device receives a 7-bit address plus 1 direction bit sent from the master as data and holds the SDA line Low for the duration of the acknowledge clock (9th clock) output from the master.

Thereafter, this device continues to perform transfer operation as a slave receive device until a stop condition or a start condition by the restart procedure appears on the bus (free format operation). The I2C0CR1<NOACK> value has no effect on this operation.

Write to this register must be done before a start condition is generated or after a stop condition was generated. Data cannot be written during transfer.

4. I2C0CR2 (I^{2}C0 Control Register 2) (Write Only)

Address = (0x4001_3000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	MST	WO	0y0	Select the master or slave. 0y0: Slave 0y1: Master
[6]	TRX	WO	0y0	Select send or receive. 0y0: Receiver 0y1: Transmitter
[5]	BB	WO	0y0	Select generation of start or stop condition. 0y0: Generation of stop condition 0y1: Generation of start condition
[4]	PIN	WO	0y1	Set release of interrupt service requests. 0y0: Do nothing 0y1: Release interrupt requests.
[3]	I2CM	WO	0y0	I^{2}C operation control 0y0: Disable 0y1: Enable
[2]	–	–	Undefined	Read undefined. Write as zero.
[1:0]	SWRES[1:0]	WO	0y00	Occurrence of software reset Write "10" and "01" in this order for occurrence of software reset.

[Explanation]

a. <MST>

This bit is used to select the master or slave.

0y0: Slave

0y1: Master

Note) Refer to 3.14.6.4, "Master/Slave Selection."

b. <TRX>

This bit is used to select send or receive.

0y0: Receiver

0y1: Transmitter

Note) Refer to 3.14.6.5, "Transmitter/Receiver Selection."

c. <BB>

This bit is used to select generation of start or stop condition.

0y0: Generation of stop condition

0y1: Generation of start condition

Note) Refer to 3.14.6.6, "Generation of Start/Stop Condition."

d. <PIN>

This bit is used to set release of interrupt service requests for I2C communication.

0y0: Do nothing

0y1: Release interrupt service requests.

Note) Refer to 3.14.6.7, "Interrupt Service Request and Release."

e. <I2CM>

This bit is used to disable/enable an I^2C operation.

0y0: Disable

0y1: Enable

It is not possible to disable the operation during transfer. Read the status register to make sure that transfer is completed before disabling the operation.

f. <SWRES[1:0]>

When "10" and "01" are written to these two bits in this order, software reset takes place. (Reset width = One f_{PCLK} clock)

If software reset takes place, the SCL and SDA lines are released (held "High") forcibly to abort transfer operation even when it is ongoing. Also, all the settings except I2C0CR2<I2CM> are initialized. (I2C0DBR is not initialized.)

For software reset, be sure to write "0" to I2C0CR2[7:4].

5. I2C0SR (I2C0 Status Register) (Read Only)

Address = (0x4001_3000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7]	MST	RO	0y0	Monitor master/slave selection. 0y0: Slave 0y1: Master
[6]	TRX	RO	0y0	Monitor send/receive selection. 0y0: Receiver 0y1: Transmitter
[5]	BB	RO	0y0	Monitor bus status. 0y0: Bus free 0y1: Bus busy
[4]	PIN	RO	0y1	Monitor interrupt service request status and SCL line status. 0y0: Interrupt service is being requested. SCL line is held "Low." 0y1: Interrupt service is not requested. SCL line is free.
[3]	AL	RO	0y0	Monitor detection of arbitration lost. 0y0: Disabled 0y1: Detected
[2]	AAS	RO	0y0	Monitor detection of slave address matching. 0y0: Disabled 0y1: Detected
[1]	AD0	RO	0y0	Monitor detection of a general call. 0y0: Disabled 0y1: Detected
[0]	LRB	RO	0y0	Monitor the bit received last. 0y0: The bit received last is 0. 0y1: The bit received last is 1.

[Explanation]

a. <MST>

This bit is used to monitor the selection of master or slave.

0y0: Slave

0y1: Master

b. <TRX>

This bit is used to monitor selection of send or receive.

0y0: Receiver

0y1: Transmitter

c. <BB>

This bit is used to monitor the bus status.

0y0: Bus free

0y1: Bus busy

This bit is set to "1" after detection of a start condition on the bus. It is cleared to "0" on detection of a stop condition. When the module operates as a slave device, this bit is set to "1" and the device monitors generation of a stop condition even if a match of the address sent from the master is not detected and therefore the device is not involved in transfer operation.

While this bit is set to "1," the start condition cannot be generated.

d. <PIN>

This bit is used to monitor the interrupt service request status and the SCL status.

0y0: Interrupt service is being requested. The SCL line is "Low" OUT.

0y1: Interrupt service is not requested. SCL line is free.

e. <AL>

This bit is used to monitor the detection of arbitration lost.

0y0: Disabled

0y1: Detected

f. <AAS>

This bit is used to monitor the detection of a slave address matching.

0y0: Disabled

0y1: Detected

When the module operates as a slave device, this bit is set to "1" if the address sent from the master matches the I2C0AR<SA[6:0]> value. This bit is then cleared to "0" after the internal interrupt is released and remains unchanged until a stop condition or a start condition by the restart procedure appears on the bus and it is again set to "1" by an address matching detected in address transfer after that start condition.

g. <AD0>

This bit is used to monitor the detection of a general call.

0y0: Disabled

0y1: Detected

This bit is set to "1" on detection of a general call (the SDA line is held Low at address transfer after the start condition) and remains in that state until a stop condition or a start condition by the restart procedure appears on the bus. I2C0SR<AAS> is also set to "1" on reception of a general call but it is cleared to "0" at transfer of the next data as described earlier.

h. <LRB>

This bit is used to monitor the bit received last.

0y0: The bit received last is 0.

0y1: The bit received last is 1.

This is the bit received last monitor.

When "Acknowledge provided" is set, the user reads this bit at interrupt after transfer to check whether the receive device has output an acknowledge signal (the signal is Low). This monitor is effective regardless of whether the device is set as a transmitter or receiver.

Note) Refer to 3.14.6.15 "Register Values after Software Reset."

6. I2C0PRS (I2C0 Prescaler Clock Set Register)

Address = (0x4001_3000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read undefined. Write as zero.
[4:0]	PRSCK[4:0]	R/W	0y00001	Select a prescaler clock frequency for generation of serial clock. 0y00000: This setting cannot be used 0y00001: p = divide by 1 0y00010: ≀ This setting cannot be used 0y00111: 0y01000: p = divide by 8 ≀ 0y10101: p = divide by 21 0y10110: ≀ This setting cannot be used 0y11111:

[Explanation]

a. <PRSCK[4:0]>

These bits are used to select a prescaler clock frequency for generation of serial clock.

0y00000: This setting cannot be used

0y00001: p = divide by 1

0y00010:

≀ This setting cannot be used

0y00111:

0y01000: p = divide by 8

≀

0y10101: p = divide by 21

0y10110:

≀ This setting cannot be used

0y11111:

Note) For 1. I2C0CR1 (I2C0 Control Register 1), refer to 3.14.6.3 "Serial Clock."

* Note *

The setting range for the prescaler clock width varies with the operation frequency (PCLK). Determine the settable range of prescaler setting "p" (I2C0PRS<PRSCK[4:0]>) in a way to meet the condition below.

50 ns < Prescaler clock width Tprsck (ns) ≤ 150 ns

Note) Setting the prescaler out of this range is prohibited regardless of whether the device is set as a transmitter or receiver.

7. I2C0IE (I^2C0 Interrupt Enable Register)

Address = (0x4001_3000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	IE	R/W	0y0	I^2C interrupt 0y0: Disable 0y1: Enable

[Explanation]

a. <IE>

This bit is used to enable/disable the I^2C interrupt output.

0y0: Disable

0y1: Enable

8. I2C0IR (I^2C0 Interrupt Register)

Address = (0x4001_3000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	(	Undefined	Read undefined. Write as zero.
[0]	IS/IC	R/W	0y0	(For read) I^2C interrupt status (before being disabled) 0y0: No interrupts are detected. 0y1: Interrupt occurred. (For write) Clear the I^2C interrupt. 0y0: Disabled 0y1: Clear

[Explanation]

a. <IS/IC>

(For read)

This bit is used to indicate the I^2C interrupt status before being masked by I2C0IE<IE>.

0y0: No interrupts are detected.

0y1: Interrupt occurred.

(For write)

This bit is used to clear the I^2C interrupt.

0y0: Disabled

0y1: Clear

When "1" is written, the I^2C interrupt output (I2CINT0) is cleared.

When "0" is written, nothing happens.

3.14.6 Function

3.14.6.1 Selecting Detection of Slave Address Matching and General Call

The slave device makes the following settings when detecting a slave address matching and a general call.

I2C0CR1<NOACK> is used to enable/disable detection of a slave address matching and a general call in the slave mode.

When I2C0CR1<NOACK> is cleared to "0," the detection of a slave address matching and a general call is enabled.

When I2C0CR1<NOACK> is set to "1," the detection of a slave address matching and a general call is disabled. The slave address or general call sent from the master will be ignored, an acknowledge signal will not be returned, and no I2CINT0 interrupt requests will occur.

In the master mode, the I2C0CR1<NOACK> setting is ignored and has no effect on the operation.

Note) Even if I2C0CR1<NOACK> is cleared to "0" during data transfer in the slave mode, it remains "1" and an acknowledge signal during the data transfer is returned.

3.14.6.2 Selecting the Number of Clocks for Data Transfer and Whether or not Acknowledgement is Provided

(2) Number of clocks for data transfer

The number of clocks for data transfer is set by I2C0CR1<BC> and I2C0CR1<ACK>.

When I2C0CR1<ACK> is set to "1," the device operates in acknowledgement mode.

In acknowledgement mode, the master device generates clocks that correspond to the number of data bits, and then generates the clocks for an acknowledge signal and an I2CINT0 interrupt request.

The slave device counts the clocks that correspond to the number of data bits, and then counts the clocks for an acknowledge signal and generates an I2CINT0 interrupt request.

When I2C0CR1<ACK> is cleared to "0," the device operates in non-acknowledgement mode.

In non-acknowledgement mode, the master device generates clocks that correspond to the number of data bits, and then generates an I2CINT0 interrupt request.

The slave device counts the clocks that correspond to the number of data bits, and then generates an I2CINT0 interrupt request.

When "Acknowledge provided" is set to the receive device, the I2C0SDA pin is driven low for the duration of the acknowledge clock output from the master to request the send device to transfer the next word. Conversely, when "Acknowledge not provided" is set to the receive device, the I2C0SDA pin is released (held High) even for the duration of the acknowledge clock output from the master to notify the send device that transfer of the next word is not requested.

When an address is sent (or before the start condition is generated), the number of transfer bits must be set to 8 with acknowledge provided for both master and slave.

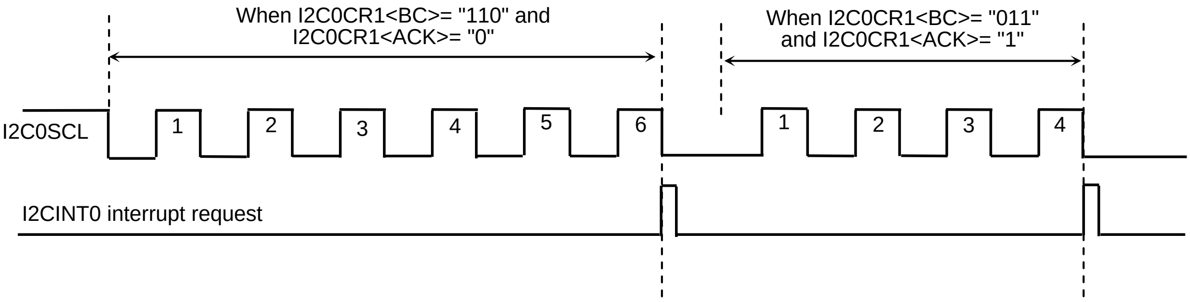


Figure 3.14.7 Number of data transfer clocks, and I2C0CR1<BC>, and I2C0CR1<ACK>

The relationship between the number of clocks for data transfer and I2C0CR1<BC> and I2C0CR1<ACK> is shown in Table 3.14.3.

Table 3.14.3 Number of clocks for data transfer

BC[2:0]	Acknowledgement operation (I2C0CR1<ACK>)			
	0: Not provided		1: Provided	
	Data length	Number of clocks	Data length	Number of clocks
000	8	8	8	9
001	1	1	1	2
010	2	2	2	3
011	3	3	3	4
100	4	4	4	5
101	5	5	5	6
110	6	6	6	7
111	7	7	7	8

<BC> is cleared to "000" by the start condition.

Therefore, a slave address and direction bit are always transferred in 8 bits. If <BC> is not cleared to "000," it holds the value that has already been set.

Note) Before sending or receiving a slave address, set I2C0CR1<ACK>. If I2C0CR1<ACK> is cleared, detection of a slave address matching and a direction bit cannot be performed properly.

(3) Acknowledge output

In acknowledgement mode, I2C0SDA changes as follows during the period of the clocks for an acknowledge signal.

- In master mode:

If the device is set as a transmitter, the I2C0SDA pin is released to receive an acknowledge signal output from the receiver during the period of the clocks for an acknowledge signal.

If the device is set as a receiver, the I2C0SDA is held low to generate an acknowledge signal during the period of the clocks for an acknowledge signal.

- In slave mode:

When the received slave address matches the slave address set in I2C0AR<SA> or a general call is received, I2C0SDA is held low to generate an acknowledge signal during the period of the clocks for an acknowledge signal.

During the data transfer after the slave addresses match or a general call is received, if the device is set as a transmitter, the I2C0SDA pin is released to receive an acknowledge signal output from the receiver during the period of the clocks for an acknowledge signal.

If the device is set as a receiver, the I2C0SDA is held low to generate an acknowledge signal.

Table 3.14.4 shows the states of I2C0SCL and I2C0SDA in the acknowledgement mode.

Note: In non-acknowledgement mode, an acknowledge signal is not output because the clocks for an acknowledge signal are neither generated nor counted.

Table 3.14.4 States of I2C0SCL and I2C0SDA in the acknowledgement mode

Mode	Pin	Condition	Transmitter	Receiver
Master	I2C0SCL	–	Add the clocks for an acknowledge signal	Add the clocks for an acknowledge signal
	I2C0SDA	–	Release the pin to receive an acknowledge signal	Output "Low" to the pin as an acknowledge signal
Slave	I2C0SCL	–	Count the clocks for an acknowledge signal	Count the clocks for an acknowledge signal
	I2C0SDA	When the slave addresses match, or when a general call is received	–	Output "Low" to the pin as an acknowledge signal
		When the slave addresses match, or during data transfer after a general call was received	Release the pin to receive an acknowledge signal	Output "Low" to the pin as an acknowledge signal

3.14.6.3 Serial Clock

(1) Clock source

I2C0CR1<SCK[2:0]> and I2C0PRS<PRSCK[4:0]> are used to set the rate of the serial clock output from the master.

The prescaler clock divided according to I2C0PRS<PRSCK[4:0]> is used as the reference clock of serial clock generation. The prescaler clock is further divided according to I2C0CR1<SCK[2:0]> and used as the serial clock.

The prescaler clock is also used as the basic clock of the internal digital noise canceller. The default setting of the prescaler clock is "divide by 1" (= PCLK).

<About the prescaler clock width (= noise cancellation width)>

The prescaler clock width (Tprsck) (or noise cancellation width) is determined by prescaler setting "p" (I2C0PRS<PRSCK[4:0]>) based on the operation frequency (PCLK) as follows:

Prescaler clock width
(or noise cancellation width)

$$\text{Tprsck(s)} = \frac{1}{f_{\text{PCLK}}(\text{Hz})} \times p$$

* Note *

The setting range for the prescaler clock width varies with the operation frequency (PCLK). Determine the settable range of prescaler setting "p" (I2C0PRS<PRSCK[4:0]>) in a way to meet the condition below.

50 ns < Prescaler clock width Tprsck (ns) ≤ 150 ns

Note) Setting the prescaler out of this range is prohibited regardless of whether the device is set as a transmitter or receiver.

From the above expression, the range in which prescaler settings (@ f_{PCLK} = 144 MHz) can be made is 8 ≤ "p" (I2C0PRS<PRSCK[4:0]>) ≤ 21.

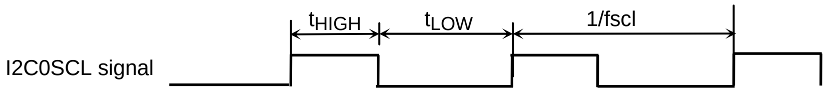
<About the serial transfer rate>

The serial clock rate (Fscl) is determined by the combination of prescaler setting "p" (I2C0PRS<PRSCK[4:0]>, 8 to 21) and serial clock setting I2C0CR1<SCK> based on the operation frequency (PCLK) as follows:

$$\text{Serial clock rate Fscl(Hz)} = \frac{f_{\text{PCLK}}(\text{Hz})}{p \times (2^{n+2} + 16)}$$

I2C0CR1<SCK> determines the HIGH time (t_{HIGH}) and LOW time (t_{LOW}) of the serial clock output from the master.

	t_{HIGH} = i*Tprsck	t_{LOW} = j *Tprsck
SCK[2:0]	i	j
000:	8	12
001:	10	14
010:	14	18
011:	22	26
100:	38	42
101:	70	74
110:	134	138
111:	262	266



t_{HIGH} = (i*Tprsck)
t_{LOW} = (j*Tprsck)
fscl = 1/(t_{HIGH} + t_{LOW})

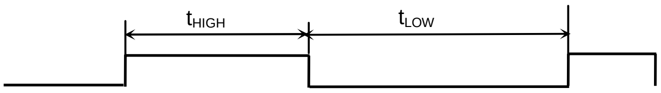
Figure 3.14.8 I2C0SCL output

Note) The rising edge may become blunt due to a combination of bus load capacity and pull-up resistance, and the t_{HIGH} may not be reached. In addition, when a function to keep synchronization with a serial clock from another device (clock synchronization function) is operated, a generated clock may differ from the setting value.

When the device is set as a master, the hold time when a start condition is generated and the setup time when a stop condition is generated become t_{HIGH}[s].

When the device is set as a slave and I2C0CR2<PIN> is set to "1," the time of t_{LOW}[s] is required before releasing the I2C0SCL pin.

In both master and slave modes, the externally input serial clock of 4/Tprsck[s] or 5/Tprsck[s] or more is required for "H" and "L" levels, respectively, regardless of the I2C0CR1<SCK> setting.



t_{HIGH} >= (4/Tprsck)
t_{LOW} >= (5/Tprsck)

Figure 3.14.9 I2C0SCL input

The following lists the setting examples of the serial clock rate (Fscl):

* PCLK=144MHz

SCK[2:0]=(n)			PRSCK[4:0]=(p)			
			0y01000 (divide by 8)	0y10011 (divide by 19)	0y10100 (divide by 20)	0y100101 (divide by 21)
0	0	0	This setting cannot be used	This setting cannot be used	360.00 KHz (400)	342.86 KHz (420)
0	0	1	This setting cannot be used	315.79 KHz (456)	300.00 KHz (480)	285.71 KHz (504)
0	1	0	This setting cannot be used	236.84 KHz (608)	225.00 KHz (640)	214.29 KHz (672)
0	1	1	This setting cannot be used	157.89 KHz (912)	150.00 KHz (960)	142.86 KHz (1008)
1	0	0	225.00 KHz (640)	94.74 KHz (1520)	90.00 KHz (1600)	85.71 KHz (1680)
1	0	1	125.00 KHz (1152)	52.63 KHz (2736)	50.00 KHz (2880)	47.62 KHz (3021)
1	1	0	66.18 KHz (2176)	27.86 KHz (5168)	26.47 KHz (5440)	25.21 KHz (5712)
1	1	1	34.09 KHz (4224)	14.35 KHz (10032)	13.64 KHz (10560)	12.99 KHz (11088)

The number in parentheses indicates the frequency division rate against fPCLK.

Since a function to keep synchronization with a serial clock from another device is provided, there may be cases where the serial clock rate is not constant.

Write to these bits must be done before a start condition is generated or after a stop condition was generated.
Write during transfer will cause unexpected operation.

(2) Clock synchronization

Because the I^{2}C bus is driven by wired-AND due to the pin structure, the master that held the clock line "Low" first disables the clocks from the master that outputs "High" level. Therefore, the master that outputs "High" level must detect the operation and take appropriate action.

Because the I^{2}C has the clock synchronization function, data transfer is performed properly even when there are multiple masters on the bus.

The following describes the procedures for clock synchronization by using the case in which two masters exist on a bus as an example.

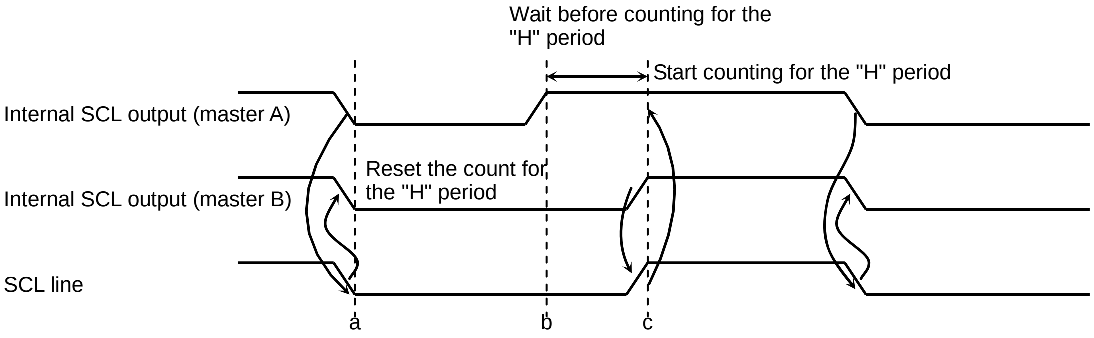

Wait before counting for the "H" period
Start counting for the "H" period
Internal SCL output (master A)
Reset the count for the "H" period
Internal SCL output (master B)
SCL line
a
b
c

Figure 3.14.10 Example of clock synchronization

The master A holds I2C0SCL "Low" at point a, which causes the SCL line on the bus to fall at "Low" level. The master B detects it and resets the count for the "High" period of the master B, and holds I2C0SCL "Low."

The master A finishes counting for the "L" period at point b, which causes I2C0SCL to rise at "H" level. However, because the master B keeps holding the SCL line "Low," the master A does not start counting for the "H" period. When the master A detects that I2C0SCL has been set to "H" by the master B at point c and the SCL line on the bus has reached the "H" level, the master A starts counting for the "H" period.

After that, when the master A has completed counting for the "H" period, it holds I2C0SCL "Low," which causes the SCL line on the bus to fall at "L" level.

As described above, the clock on the bus is decided by the master that has the shortest "H" period and the master that has the longest "L" period out of the masters that are connected to the bus.

3.14.6.4 Master/Slave Selection

When I2C0CR2<MST> is set to "1," the I^{2}C operates as a master device.

When I2C0CR2<MST> is cleared to "0," the I^{2}C operates as a slave device.

I2C0SR<MST> is cleared to "0" by hardware when a stop condition is detected on the bus or when arbitration lost is detected.

3.14.6.5 Transmitter/Receiver Selection

When I2C0CR2<TRX> is set to "1," the I^2C operates as a transmitter. When I2C0CR2<TRX> is cleared to "0," it operates as a receiver.

During data transfer in the I2C bus mode, if the device operates as a slave, the hardware sets I2C0SR<TRX> to "1" if the direction bit (R/W) sent from a master device is set to "1," or clears I2C0SR<TRX> to "0" if the direction bit is set to "0."

If the device operates as a master, when an acknowledge signal is returned from a slave device, the hardware clears I2C0SR<TRX> to "0" if the sent direction bit device is set to "1," or sets I2C0SR<TRX> to "1" if the direction bit is set to "0." If an acknowledge signal is not returned, the current status is retained.

I2C0SR<TRX> is cleared to "0" by hardware when a stop condition is detected on the bus or when arbitration lost is detected. Table 3.14.5 shows the conditions for changing I2C0SR<TRX> in each mode and the changed I2C0SR<TRX> values.

Note) When I2C0CR1<NOACK> is set to "1," I2C0SR<TRX> will not change because detection of a slave address matching and a general call is prohibited.

Table 3.14.5 Operation of I2C0SR<TRX> in each mode

Mode	Direction bit	Change condition	TRX after change
Slave mode	"0"	When the received slave address matches the value set in I2C0AR<SA>	"0"
	"1"		"1"
Master mode	"0"	When an ACK signal is returned	"1"
	"1"		"0"

When the I^2C is used in the free data format, recognition of a slave address and a direction bit is not performed, and the signals are treated as data immediately after the start condition. Therefore, I2C0SR<TRX> is not changed by the hardware.

3.14.6.6 Generation of Start/Stop Condition

When I2C0SR<BB> is set to "0," writing "1" in I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB>, and I2C0CR2<PIN> causes a start condition, the slave address that has been written in the data buffer register, and a direction bit to be output to the bus. Set I2C0CR1<ACK> to "1" before generating a start condition.

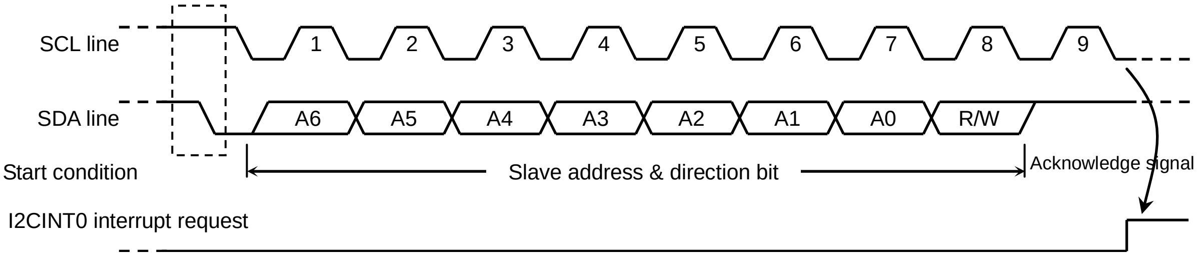


Figure 3.14.11 Generating a start condition and slave address

When I2C0SR<BB> is set to "1," writing "1" in I2C0CR2<MST>, I2C0CR2<TRX>, and I2C0CR2<PIN> and "0" in I2C0CR2<BB> starts the sequence for outputting a stop condition to the bus, and generates a stop condition on the bus.

If the SCL line on the bus is held "Low" by another device when a stop condition is generated, a stop condition is generated after the SCL line is released.

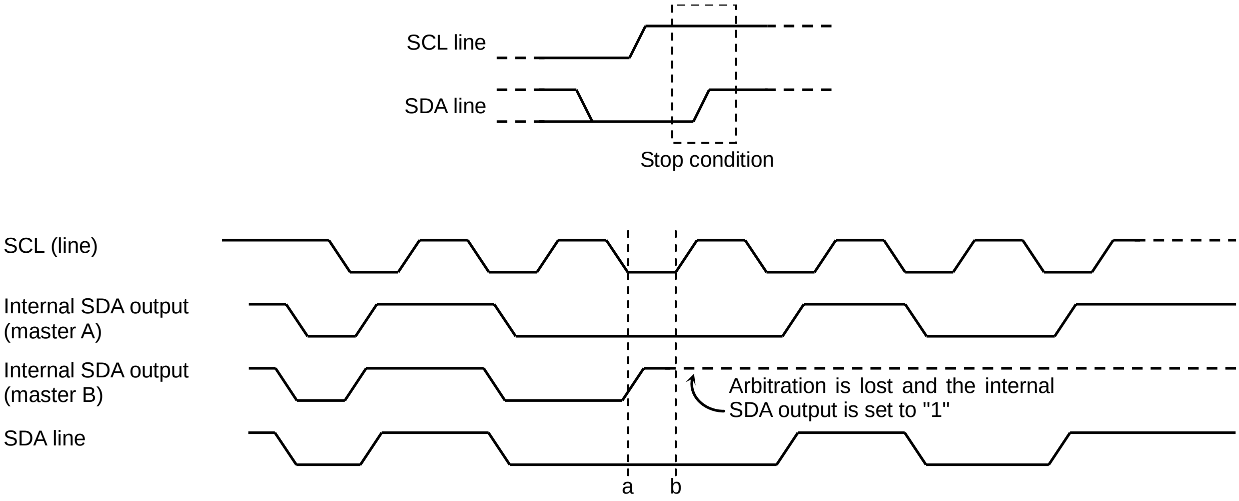

SCL line
SDA line
Stop condition
SCL (line)
Internal SDA output (master A)
Internal SDA output (master B)
SDA line
Arbitration is lost and the internal SDA output is set to "1"
a
b

Figure 3.14.12 Generating a stop condition

In addition, the bus status can be checked by reading the I2C0SR<BB> value. I2C0SR<BB> is set to "1" (bus busy) on detection of a start condition on the bus. It is cleared to "0" (bus free) on detection of a stop condition.

The following shows the general operation status of I2C0SR and the sample settings for each operation.

The bits I2C0CR2<MST>, <TRX>, <BB>, and <PIN>, although they are of originally independent functions, are used in any of the following predetermined combinations according to the I2C0SR status.

I2C0SR			I2C0CR2				Operation
[7]MST	[5]BB	[4]PIN	[7]MST	[6]TRX	[5]BB	[4]PIN	
0	0	1	0	0	0	0	Function as a slave that waits for a start condition
			1	1	1	1	A start condition is generated.
1	1	0	1	1	0	1	A stop condition is generated.
			0	0	0	1	Internal interrupts are released for restarting.

Do not change I2C0CR2<I2CM> by mistake when data is written in these bits.

3.14.6.7 Interrupt Service Request and Release

In the master mode, an I2CINT0 interrupt service request is generated when transferring data for the number of clocks for data transfer, which is set by I2C0CR1<BC> and I2C0CR1<ACK>, has completed.

In the slave mode, an I2CINT0 interrupt request is generated when the following conditions are met in addition to the above.

- After an acknowledge signal is output when I2C0CR1<NOACK> is set to "0" and the received slave address matches with the slave address set in I2C0AR<SA>
- After an acknowledge signal is output when I2C0CR1<NOACK> is set to "0" and a general call is received
- When the slave addresses match, or when the data transfer after reception of a general call is complete
  When an I2CINT0 interrupt request is generated, I2C0SR<PIN> is cleared to "0."
  I2C0SCL is held "Low" while I2C0SR<PIN> is set to "0."

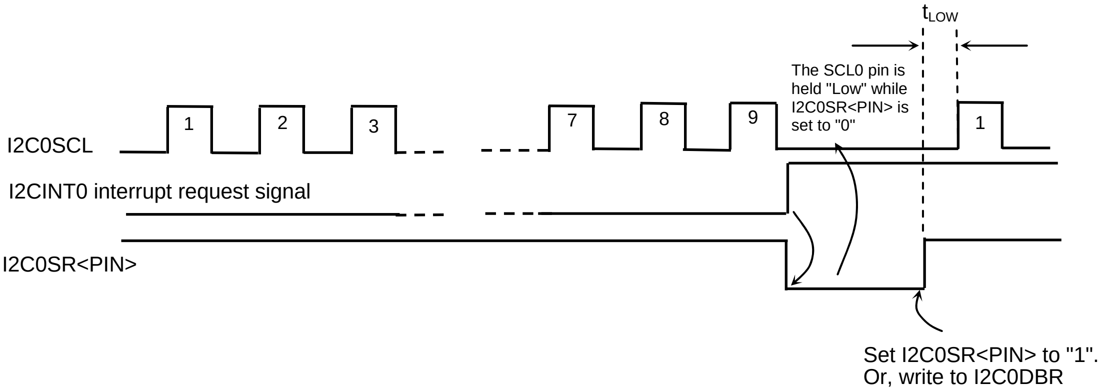

t_{LOW}
The SCL0 pin is held "Low" while I2C0SR<PIN> is set to "0"
I2C0SCL
1
2
3
7
8
9
1
I2CINT0 interrupt request signal
I2C0SR<PIN>
Set I2C0SR<PIN> to "1".
Or, write to I2C0DBR

Figure 3.14.13 I2C0SR<PIN> and I2C0SCL

When data is written in I2C0DBR, I2C0SR<PIN> is set to "1."

It will take the time of t_{LOW} after I2C0SR<PIN> is set to "1" until the I2C0SCL pin is released. When "1" is written to I2C0CR2<PIN> by a program, I2C0SR<PIN> is set to "1." However, if "0" is written, it is not cleared to "0."

3.14.6.8 I2C Bus Mode

When I2C0CR2<I2CM> is set to "1," this module operates in I^{2}C bus mode.

To use the module in I^{2}C bus mode, check that the pin is in the "H" state, and then set I2C0CR2<I2CM> to "1." To switch to the initial state, check that the bus is free, and then set I2C0CR2<I2CM> to "0."

Note) When I2C0CR2<I2CM> is set to "0," a value cannot be written in I2C0CR2[7:4].
Write "1" in I2C0CR2<I2CM> to change to the I^{2}C bus mode before setting values for I2C0CR2.

3.14.6.9 Software Reset

The I^{2}C has the software reset function, which initializes the I^{2}C. When the I^{2}C is locked due to noises, etc., this function can be used to initialize the I^{2}C.

When "10"and "01" are written to I2C0CR2<SWRES[1:0]> in this order, software reset takes place.

Though the I^{2}C is initialized by software reset, the I2C0CR2<I2CM> bit and the I2C0DBR register are not initialized.

3.14.6.10 Monitoring Detection of Arbitration Lost

Since the I^{2}C bus supports the multimaster mode (in which multiple masters exist on the same bus), a bus arbitration method is required to assure the contents of transferred data.

The I^{2}C bus uses the data on the SDA line for bus arbitration.

The following describes the procedures for arbitration by using the case in which two masters exist on a bus as an example.

The master A and master B keep outputting the same data until they reach at the bit at point a. When the master B outputs "1" and the master A output "0" at point a, the SDA line on the bus is driven by wired-AND, which causes the master A to hold the SDA line "Low."

When the SCL line on the bus rises at point b, a slave device captures the SDA line data, that is, the data from the master A.

At this time, the data output from the master B is invalid. This status of the master B is called "arbitration lost." The master B, which has lost arbitration, releases I2C0SDA and I2C0SCL so that the data from the master B does not affect the data of the master A, which has not lost arbitration. If multiple masters send identical data in the 1st word, the arbitration method is continued for the 2nd and subsequent words.

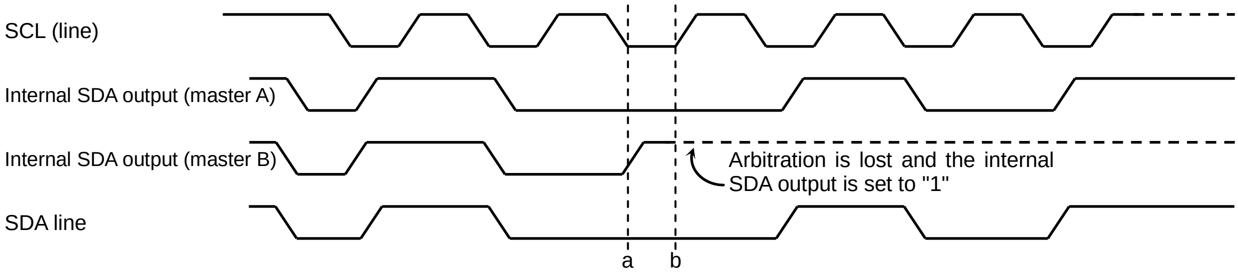

SCL (line)
Internal SDA output (master A)
Internal SDA output (master B)
SDA line
Arbitration is lost and the internal SDA output is set to "1"
a
b

Figure 3.14.14 Arbitration lost

The master B compares the level of the bus SDA line with the level of I2C0SDA at the rising edge of the SCL line. When it detects mismatch, the master B loses arbitration and I2C0SR<AL> is set to "1."

When I2C0SR<AL> is set to "1," I2C0SR<MST> and I2C0SR<TRX> are reset to "0" and the bus is in the slave receiver mode. Therefore, during data transfer after I2C0SR<AL> has been set to "1," the master B stops clock output. When data transfer is complete, I2C0SR<PIN> is cleared to "0" and I2C0SCL is held "Low."

I2C0SR<AL> is reset to "0" when data is written to or read from I2C0DBR or data is written to I2C0CR2.

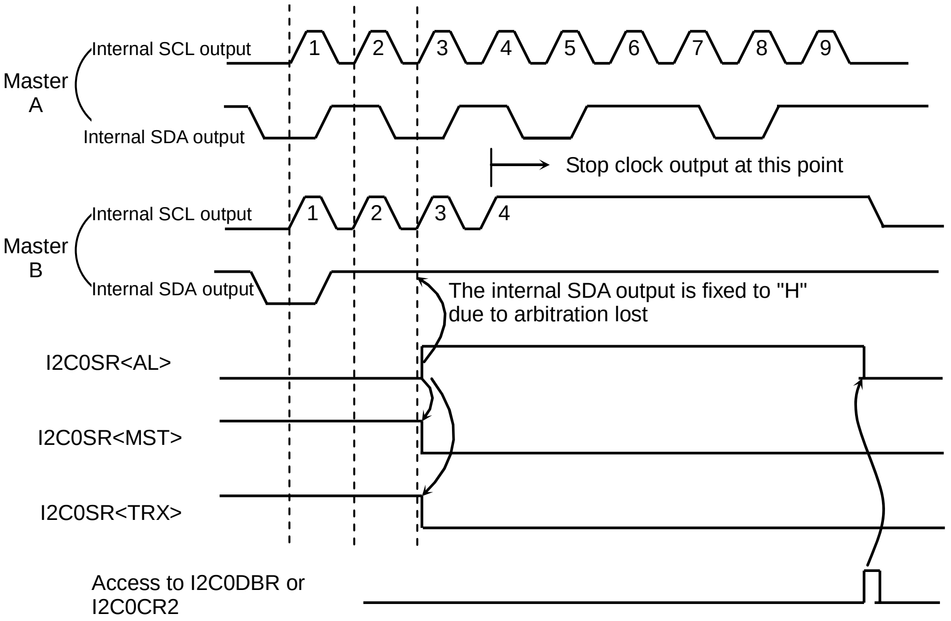


Figure 3.14.15 Arbitration lost operation (The internal flag shown above indicates the master B)

3.14.6.11 Monitoring Detection of Slave Address Matching

When this module is in I^{2}C bus mode (I2C0AR<ALS> = "0") and operates as a slave device, detection of slave address matching is possible.

When I2C0CR1<NOACK> is cleared to "0," detection of address matching is enabled. If a general call or the slave address that is identical to the value set in I2C0AR<SA> is received, I2C0SR<AAS> is set to "1."

When I2C0CR1<NOACK> is set to "1," detection of address matching is disabled. I2C0SR<AAS> is not set to "1" even if a general call or the slave address that is identical to the value set in I2C0AR<SA> is received.

When the module operates in free data format (I2C0AR<ALS>= "1"), detection of address matching does not function. When the first one word is received, I2C0SR<AAS> is set to "1." I2C0SR<AAS> is cleared to "0" when data is written to or read from I2C0DBR.

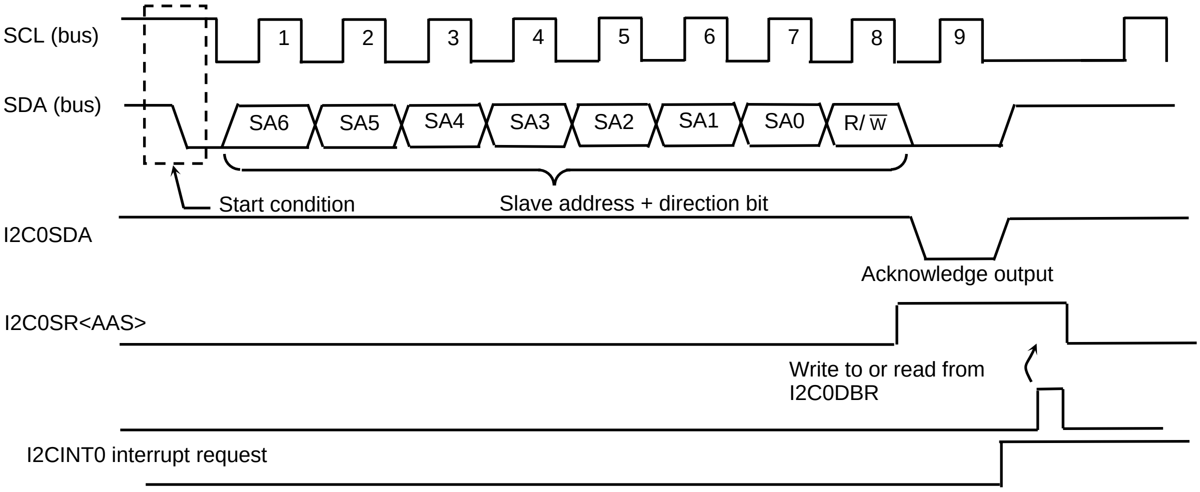


Figure 3.14.16 Change when monitoring slave address matching

3.14.6.12 Monitoring Detection of General Call

When this module is in I^{2}C bus mode (I2C0AR<ALS> = "0") and operates as a slave device, detection of a general call, along with detection of slave address matching, is possible.

When I2C0CR1<NOACK> is set to "0," I2C0SR<AD0> is set to "1" when a general call (all the 8 bits of the data received immediately after a start condition are set to "0") is received.

(I2C0SR<AAS> is also set to "1" at the same time.)

When I2C0CR1<NOACK> is set to "1," detection of a general call is disabled. In this case, I2C0SR<AD0> remains "0" even when a general call is received.

(I2C0SR<AAS> is also not set to "1.")

I2C0SR<AD0> is cleared to "0" when a start condition or stop condition on the bus is detected.

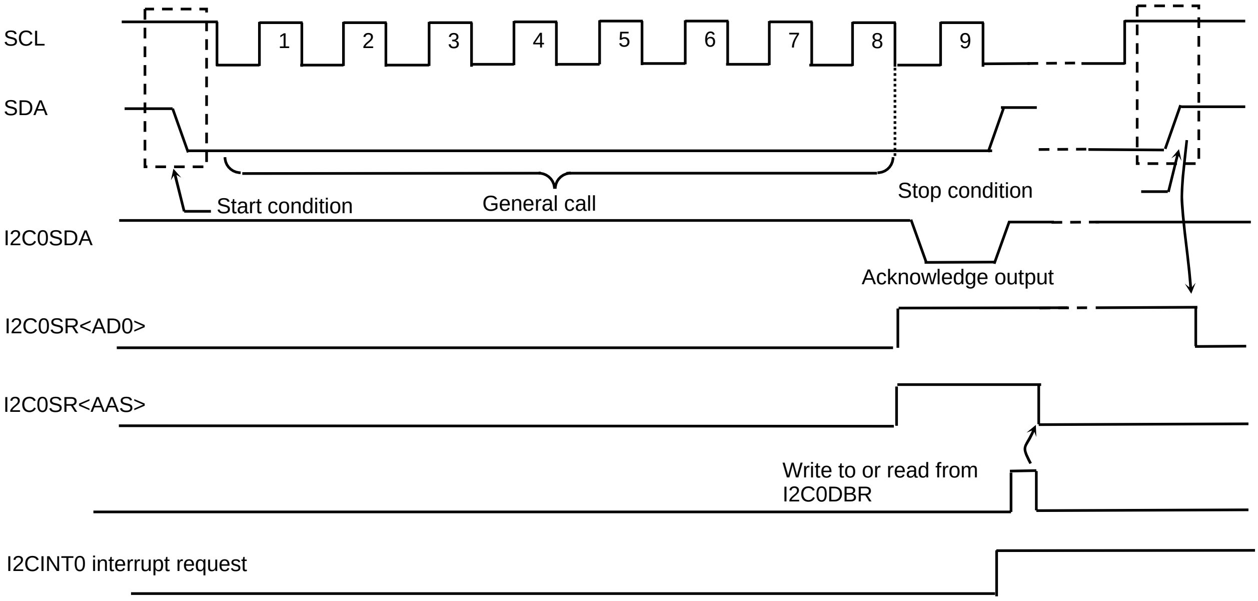


Figure 3.14.17 Change when monitoring detection of a general call

3.14.6.13 Monitoring the Bit Received Last

The value of the SDA line, which has been captured at the rising edge of the SCL line on the bus, is always updated and set in I2C0SR<LRB>.

Therefore, in the acknowledgement mode, an acknowledge signal can be read by reading a value from I2C0SR<LRB> immediately after an I2CINT0 interrupt request is generated.

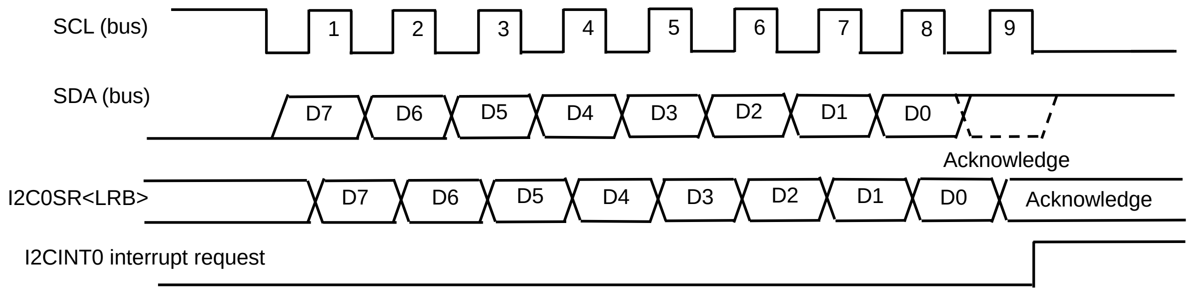


Figure 3.14.18 Change when monitoring the bit received last

3.14.6.14Setting Slave Address and Address Recognition Mode

To use the I^{2}C in I^{2}C bus mode, clear I2C0AR<ALS> to "0" and set a slave address in I2C0AR<SA>.

When using the free data format, which does not recognize a slave address, set I2C0AR<ALS> to "1." When the I^{2}C is used in the free data format, recognition of a slave address and a direction bit is not performed, and the signals are treated as data immediately after the start condition.

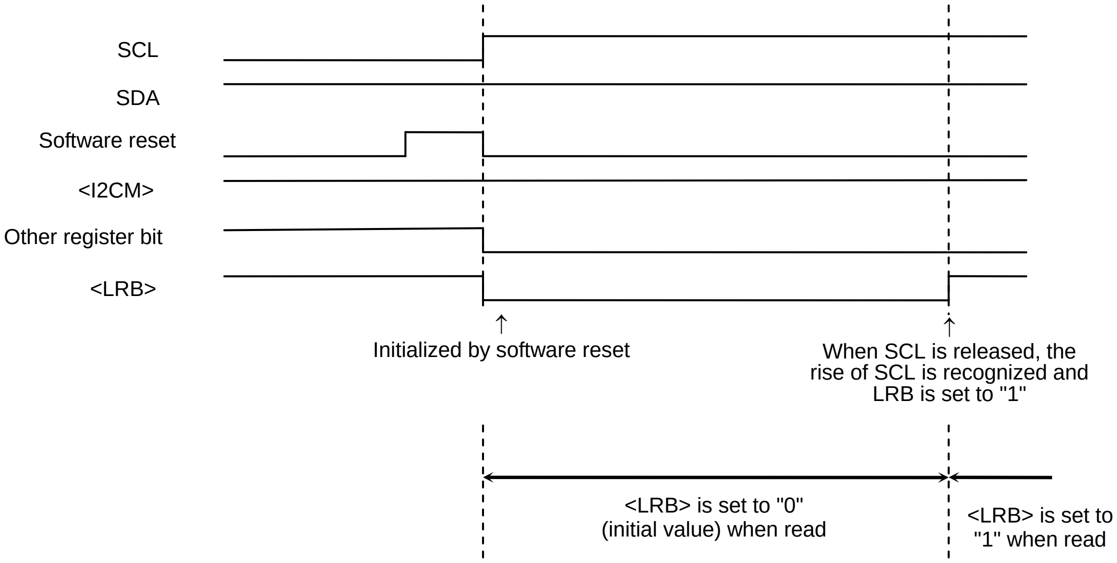
Notes related to specifications

3.14.6.15 Register Values after Software Reset

When software reset is executed, the registers other than I2C0CR2<I2CM> and the internal circuits are initialized and SCL and SDA are released. (Refer to 3.14.6.3 (2) "Clock synchronization.")

Note, however, that a value other than the initial value ("0") may be read from I2C0SR<LRB> depending on when the register is read after the software reset.

<When SCL is released from "0" to "1" by software reset while SDA = "1">

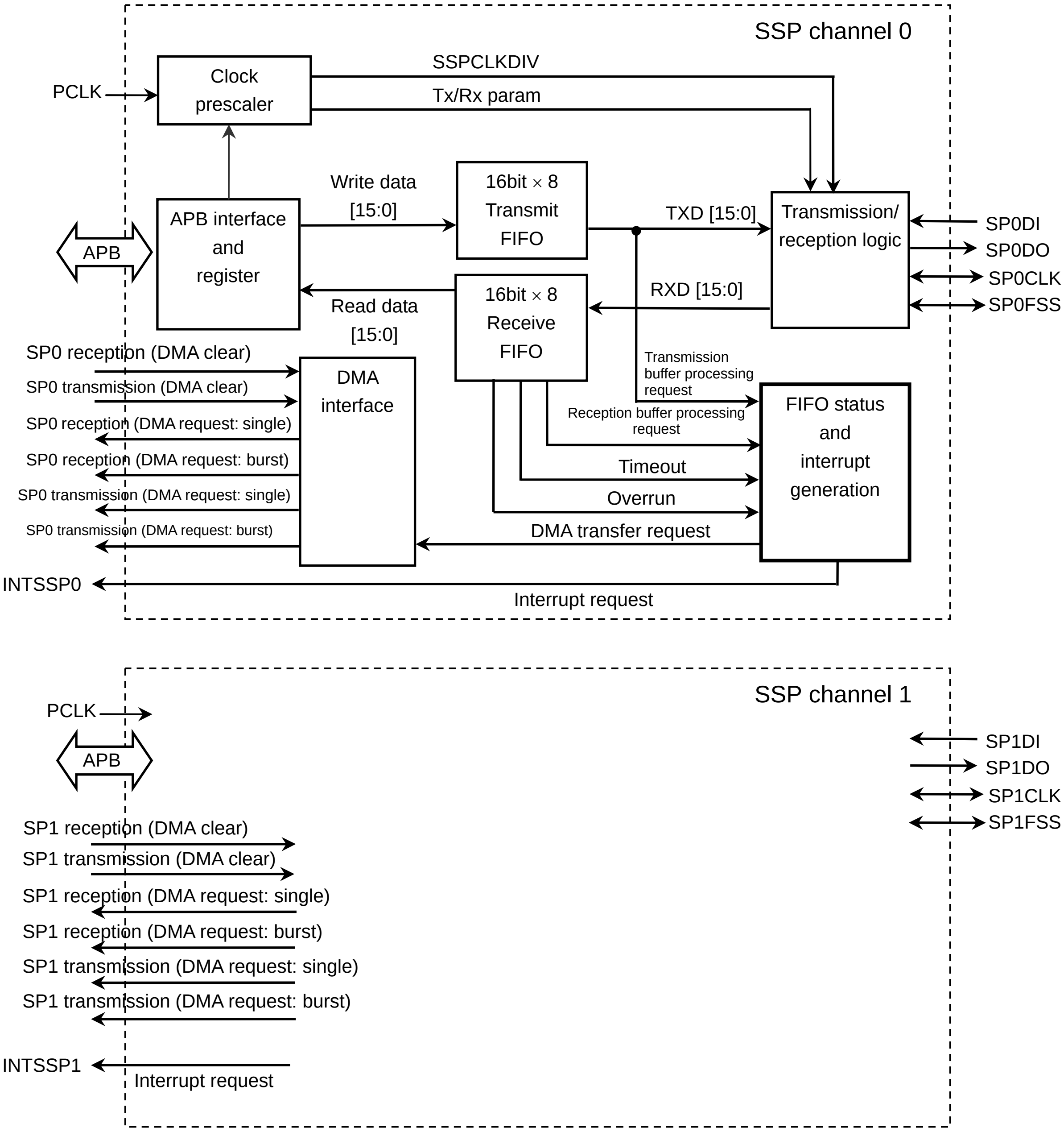

SCL
SDA
Software reset
<I2CM>
Other register bit
<LRB>
Initialized by software reset
When SCL is released, the rise of SCL is recognized and LRB is set to "1"
<LRB> is set to "0" (initial value) when read
<LRB> is set to "1" when read

3.2 SSP (Synchronous Serial Port)

This LSI contains an SSP (Synchronous Serial Port) with four channels.
Each channel has the following features:

	Channels 0 to 3			
Communication protocol	Three types of synchronous serial ports including the SPI			
Operation mode	Master/slave mode			
Transmit FIFO	16 bits wide / 8 tiers deep			
Receive FIFO	16 bits wide / 8 tiers deep			
transmitted/ received data size	4 to 16 bits			
Interrupt type	Transmit interrupt Receive interrupt Receive overrun interrupt Timeout interrupt			
Communication speed	In slave mode: f_{PCLK} (144MHz)/ 8 (max. 18Mbps)			
	In slave mode: f_{PCLK} (144MHz)/ 12 (max. 12Mbps)			
DMA	Supported			
Internal test function	Can use the internal loopback test mode.			
Control pin	Channel 0	Channel 1	Channel 2	Channel 3
	SP0CLK	SP1CLK	SP2CLK	SP3CLK
	SP0FSS	SP1FSS	SP2FSS	SP3FSS
	SP0DO	SP1DO	SP2DO	SP3DO
	SP0DI	SP1DI	SP2DI	SP3DI

3.2.1 Block Diagram


SSP channel 0
PCLK
Clock prescaler
SSPCLKDIV
Tx/Rx param
APB
APB interface and register
Write data [15:0]
Read data [15:0]
16bit × 8 Transmit FIFO
16bit × 8 Receive FIFO
TXD [15:0]
RXD [15:0]
Transmission/ reception logic
SP0DI
SP0DO
SP0CLK
SP0FSS
SP0 reception (DMA clear)
SP0 transmission (DMA clear)
SP0 reception (DMA request: single)
SP0 reception (DMA request: burst)
SP0 transmission (DMA request: single)
SP0 transmission (DMA request: burst)
DMA interface
Transmission buffer processing request
Reception buffer processing request
Timeout
Overrun
DMA transfer request
FIFO status and interrupt generation
INTSSP0
Interrupt request
SSP channel 1
PCLK
APB
SP1DI
SP1DO
SP1CLK
SP1FSS
SP1 reception (DMA clear)
SP1 transmission (DMA clear)
SP1 reception (DMA request: single)
SP1 reception (DMA request: burst)
SP1 transmission (DMA request: single)
SP1 transmission (DMA request: burst)
INTSSP1
Interrupt request

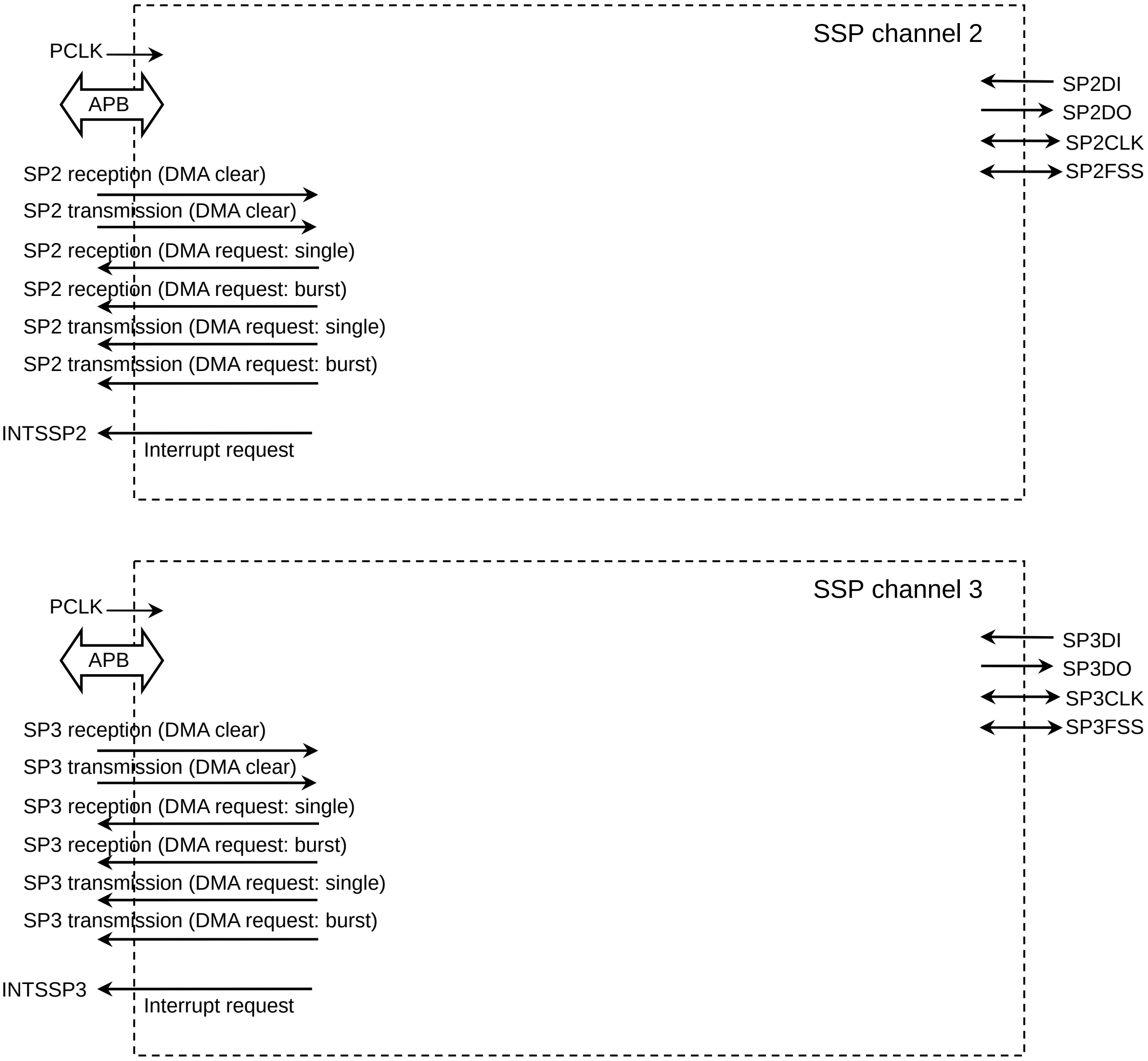

SSP channel 2
PCLK
APB
SP2 reception (DMA clear)
SP2 transmission (DMA clear)
SP2 reception (DMA request: single)
SP2 reception (DMA request: burst)
SP2 transmission (DMA request: single)
SP2 transmission (DMA request: burst)
INTSSP2
Interrupt request
SP2DI
SP2DO
SP2CLK
SP2FSS
SSP channel 3
PCLK
APB
SP3 reception (DMA clear)
SP3 transmission (DMA clear)
SP3 reception (DMA request: single)
SP3 reception (DMA request: burst)
SP3 transmission (DMA request: single)
SP3 transmission (DMA request: burst)
INTSSP3
Interrupt request
SP3DI
SP3DO
SP3CLK
SP3FSS

Figure 3.2.1 SSP block diagram

3.2.2 Overview of SSP

This LSI contains the SSP with 4 channels: channels 0, 1, 2, and 3.Since 4 channels operate in the same way, only channel 0 is described in the following sections.

The SSP is an interface that enables serial communications with the peripheral devices with three types of synchronous serial interface functions.

The SSP performs serial-parallel conversion of the data received from a peripheral device. The transmit path buffers data in the independent 16-bit wide and 8-layered transmit FIFO in the transmit mode, and the receive path buffers data in the 16-bit wide and 8-layered receive FIFO in receive mode. Serial data is transmitted via SP0DO and received via SP0DI.

The SSP contains a programmable prescaler to generate the serial output clock SP0CLK from the input clock PCLK. The operation mode, frame format, and data size of the SSP are programmed in the control registers SSP0CR0 and SSP0CR1.

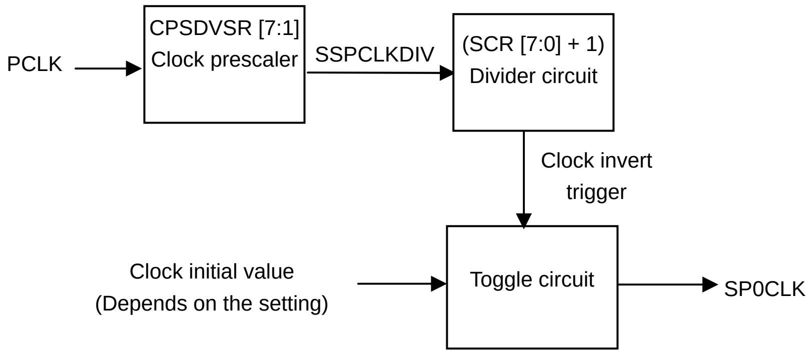
(1) Clock prescaler

When configured as a master, a clock prescaler comprising two free-running serially linked counters is used to provide the serial output clock SP0CLK.

You can program the clock prescaler through the SSP0CPSR register, to divide fPCLK by a factor of 2 to 254 in steps of two. Because the least significant bit of the SSP0CPSR register is not used, division by an odd number is not possible.

The output of the prescaler is further divided by a factor of 1 to 256, which is obtained by adding 1 to the value programmed in the SSP0CR0 control register, to give the master output clock SP0CLK.

$$\text{Bit rate} = f_{PCLK} / (\text{CPSDVSR} \times (1+\text{SCR}))$$



(2) Transmit FIFO

This is a 16-bit wide, 8-layered transmit FIFO buffer, which is shared in master and slave modes.

(3) Receive FIFO

This is a 16-bit wide 8-layered receive FIFO buffer, which is shared in master and slave modes.

(4) Interrupt generation logic

Four HIGH active interrupts, each of which can be masked separately, are generated. Also, individual interrupt requests are combined and output as a single integrated interrupt.

- Transmit interrupt: Interrupt conditional upon TxFIFO having free space equal to or more than half its entire capacity.
  (Number of valid data items in the TxFIFO ≤ 4)
- Receive interrupt: Interrupt conditional upon RxFIFO having valid data equal to or more than half its entire capacity.
  (Number of valid data items in the RxFIFO ≥ 4)
- Timeout interrupt: Interrupts indicating that the data in RxFIFO is not read before the timeout period expires.
- Receive overrun interrupt: Conditional interrupts indicating that data is written to RxFIFO when it is full

When any of the above interrupts is asserted, INTSSP0 is asserted.

(a) Transmit interrupt

The transmit interrupt is asserted when there are four or fewer valid entries in the transmit FIFO. The transmit interrupt is also generated when the SSP operation is disabled (SSPxCR1<SSE>=0).

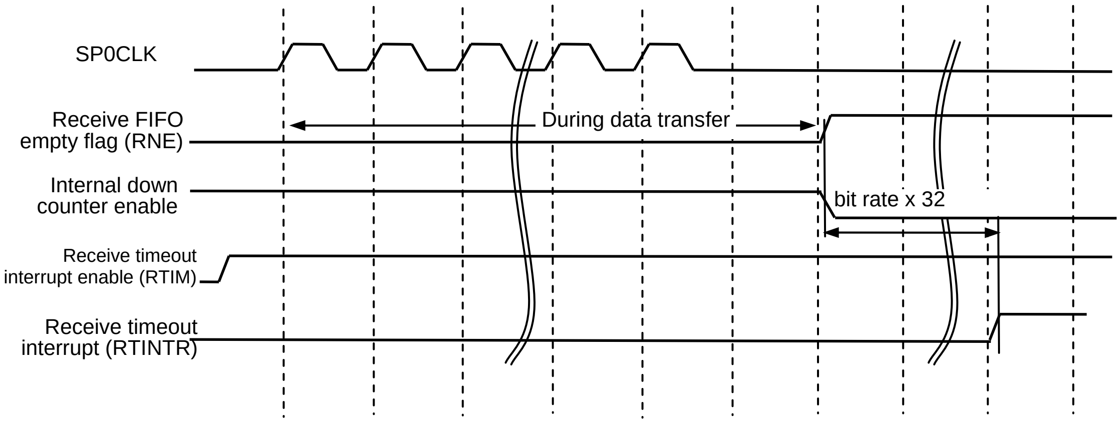
The first transmitted data can be written in the FIFO by using this interrupt.

(b) Receive interrupt

The receive interrupt is asserted when there are four or more valid entries in the receive FIFO.

(c) Timeout interrupt

The receive timeout interrupt is asserted when the receive FIFO is not empty and the SSP has remained idle for a fixed 32-bit period (bit rate). This mechanism ensures that the user is aware that data is still present in the receive FIFO and requires servicing. This operation occurs in both master and slave modes. When the timeout interrupt is generated, read all data from the receive FIFO. Even if all the data is not read, data can be transmitted/received if the receive FIFO has a free space and the number of data to be transmitted does not exceed the free space of the receive FIFO. When transfer starts, the timeout interrupt will be cleared. If data is transmitted/received when the receive FIFO has no free space, the timeout interrupt will not be cleared and an overrun interrupt will be generated.

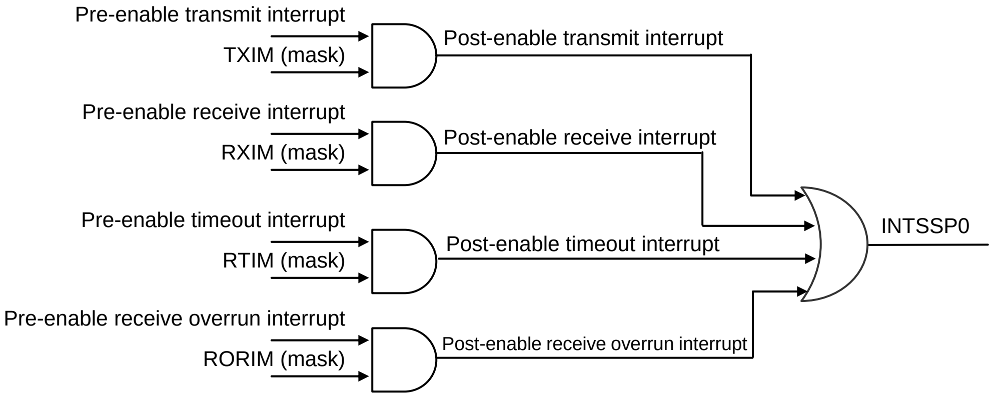

SP0CLK
Receive FIFO empty flag (RNE)
During data transfer
Internal down counter enable
bit rate x 32
Receive timeout interrupt enable (RTIM)
Receive timeout interrupt (RTINTR)

(d) Receive overrun interrupt

When the next data (9th data item) is received when the receive FIFO is already full, a receive overrun interrupt is generated immediately after transfer. The data received after the receive overrun interrupt is generated (including the 9th data item) will become invalid and be discarded. However, if data is read from the receive FIFO while the 9th data item is being received (before the interrupt is generated), the 9th received data will be written in the receive FIFO as valid data. To perform transfer properly when the receive overrun interrupt has been generated, write "1" to the receive overrun interrupt clear register, and then read all data from the receive FIFO. Even if all the data is not read, data can be transmitted/received if the receive FIFO has free space and the number of data to be transmitted does not exceed the free space of the receive FIFO. Note that if the receive FIFO is not read (provided that the receive FIFO is not empty) within a certain 32-bit period (bit rate) after the receive overrun interrupt is cleared, a timeout interrupt will be generated.

(e) Combined interrupt

The above four interrupts combine individual masked sources into a single interrupt. When any of the above interrupts is asserted, the integrated interrupt INTSSP0 is asserted.


Pre-enable transmit interrupt
TXIM (mask)
Post-enable transmit interrupt
Pre-enable receive interrupt
RXIM (mask)
Post-enable receive interrupt
Pre-enable timeout interrupt
RTIM (mask)
Post-enable timeout interrupt
Pre-enable receive overrun interrupt
RORIM (mask)
Post-enable receive overrun interrupt
INTSSP0

(5) DMA interface

The SSP provides an interface to connect to a DMA controller.

3.2.3 SSP Operation

(1) Initial settings for SSP

Settings for the SSP communication protocol must be made with the SSP disabled.

Control registers SSP0CR0 and SSP0CR1 need to configure this SSP as a master or slave operating under one of the following protocols. In addition, make the settings related to the communication speed in the prescale registers SSP0CPSR and SSP0CR0<SCR>.

This SSP supports the following protocols:

- SPI, SSI, Microwire

(2) SSP enable

The transfer operation starts when the operation is enabled with the transmitted data written in the transmit FIFO, or when transmitted data is written in the transmit FIFO with the operation enabled.

However, if the transmit FIFO contains only 4 or fewer entries when the operation is enabled, a transmit interrupt will be generated. This interrupt can be used to write the initial data.

Note) When the SSP is in the SPI slave mode and the FSS pin is not used, be sure to transmit data of 1 byte or more in the FIFO before enabling the operation. If the operation is enabled with the transmit FIFO empty, the transfer data will not be output correctly.

(3) Clock ratios

When setting a frequency for PCLK, the following conditions must be met.

[In master mode]

f_{SP0CLK} (maximum) => f_{PCLK} / 8

f_{SP0CLK} (minimum) => f_{PCLK} / (254 x 256)

[In slave mode]

f_{SP0CLK} (maximum) => f_{PCLK} / 12

f_{SP0CLK} (minimum) => f_{PCLK} / (254 x 256)

(4) Frame format

Each frame format is between 4 and 16 bits wide depending on the size of data programmed, and is transmitted starting from the MSB.

- Serial clock (SP0CLK)

Signals remain LOW in the SSI and Microwire formats and as Inactive in the SPI format while the SSP is in the idle state. In addition, data is output at the set bit rate only during data transmission.

- Serial frame (SP0FSS)

In the SPI and Microwire frame formats, signals are set to LOW Active and always asserted to LOW during frame transmission.

In the SSI frame format, signals are asserted only during 1 bit rate before each frame transmission. In this frame format, output data is transmitted at the rising edge of SP0CLK and the input data is received at its falling edge.

- Notes on the Microwire

The Microwire format uses a special master/slave messaging method, which operates in half-duplex mode. In this mode, when a frame begins, an 8-bit control message is transmitted to the slave. During this transmit, no incoming data is received by the SSP. After the message has been transmitted, the slave decodes it, and after waiting one serial clock after the last bit of the 8-bit control message has been sent, it responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

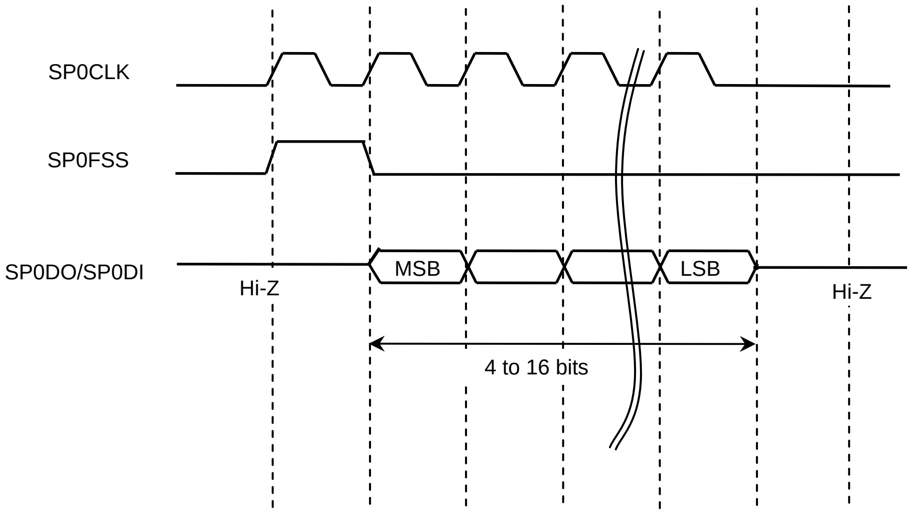
The details of each frame format are described below:

(a) SSI frame format

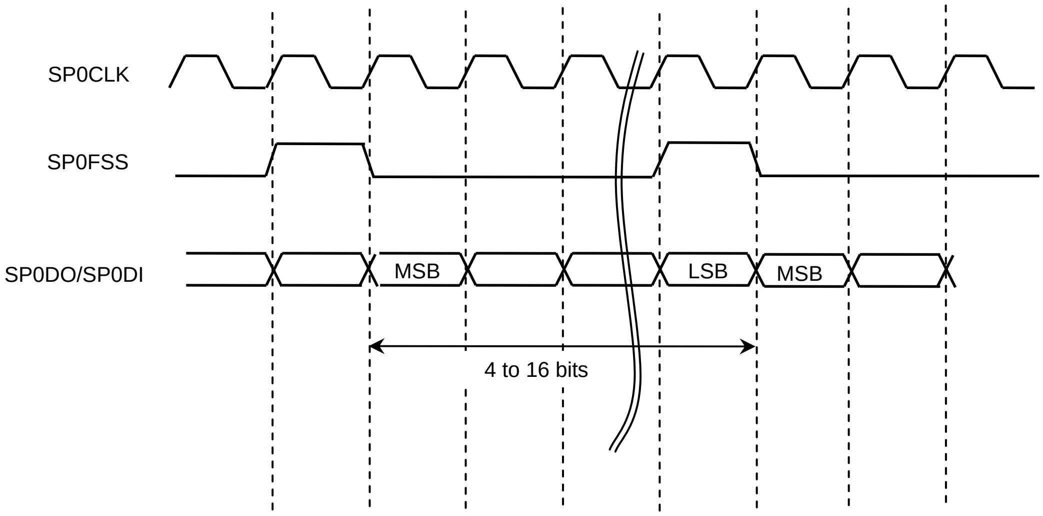
In this mode, the SSP is in idle state, SP0CLK and SP0FSS are forcedly set to LOW, and the transmit data line SP0DO becomes Hi-Z. When data is written in the transmit FIFO, the master outputs High pulses of 1 SP0CLK to the SP0FSS line. The transmitted data will be transferred from the transmit FIFO to the transmit serial shift register. Data of 4 to 16 bits will be output from the SP0DO pin at the next rising edge of SP0CLK.

Likewise, the received data will be input starting from the MSB to the SP0DI pin at the falling edge of SP0CLK. The received data will be transferred from the serial shift register into the receive FIFO at the rising edge of SP0CLK after its LSB data is latched.

SSI frame format (transmission/reception during single transfer)



SSI frame format (transmission/reception during continuous transfer)



(b) SPI

The SPI interface has 4 lines. SP0FSS is used for slave selection. One of the main features of the SPI format is that the <SPO> and <SPH> bits in the SSP0CR0 control register can be used to set the SP0CLK operation timing.

SSP0CR0<SPO>

SSP0CR0<SPO> is used to set the level at which SP0CLK in idle state is held.

<SPO>=1: Sets SP0CLK in High state

<SPO>=0: Sets SP0CLK in Low state

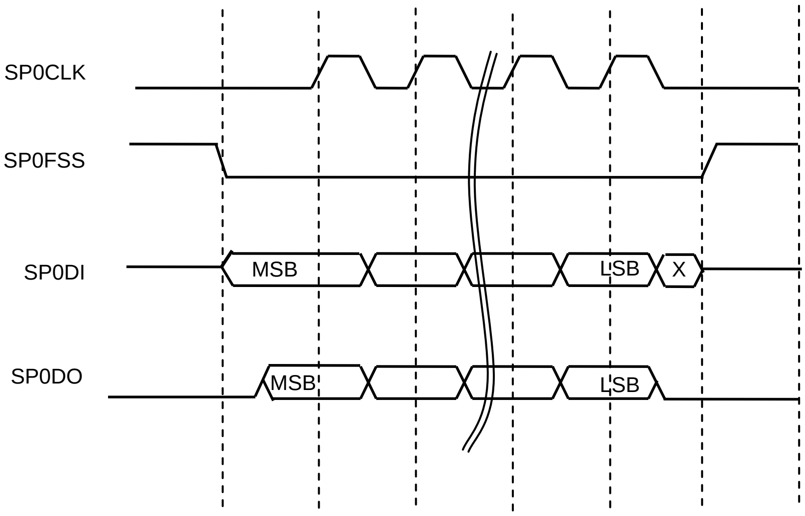
SSP0CR0<SPH>

SSP0CR0<SPH> is used to select the clock edge at which data is latched.

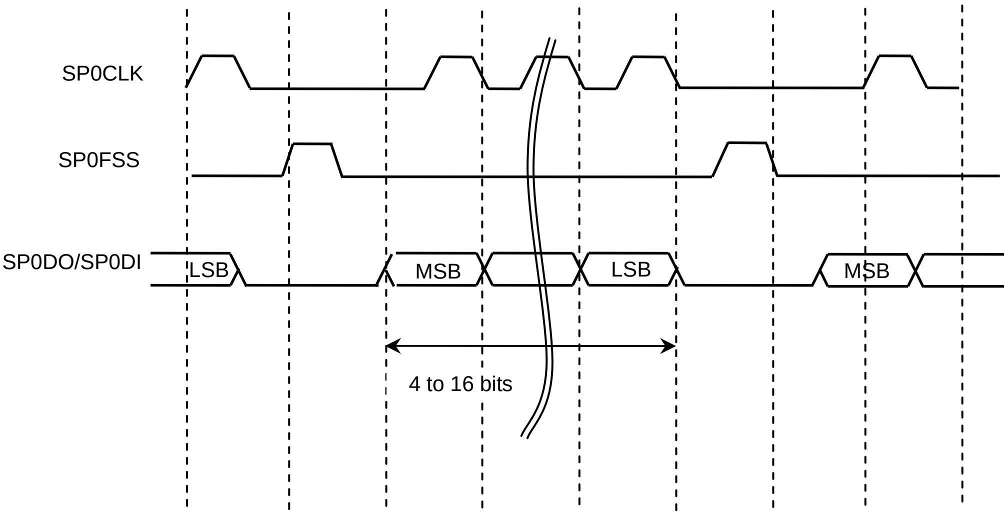
SSP0CR0<SPH>=0: Captures data at the 1st clock edge.

SSP0CR0<SPH>=1: Captures data at the 2nd clock edge.

SPI frame format (single transfer, <SPO>=0 & <SPH>=0)



SPI frame format (continuous transfer, <SPO>=0 & <SPH>=0)



With this setting, during the idle period:

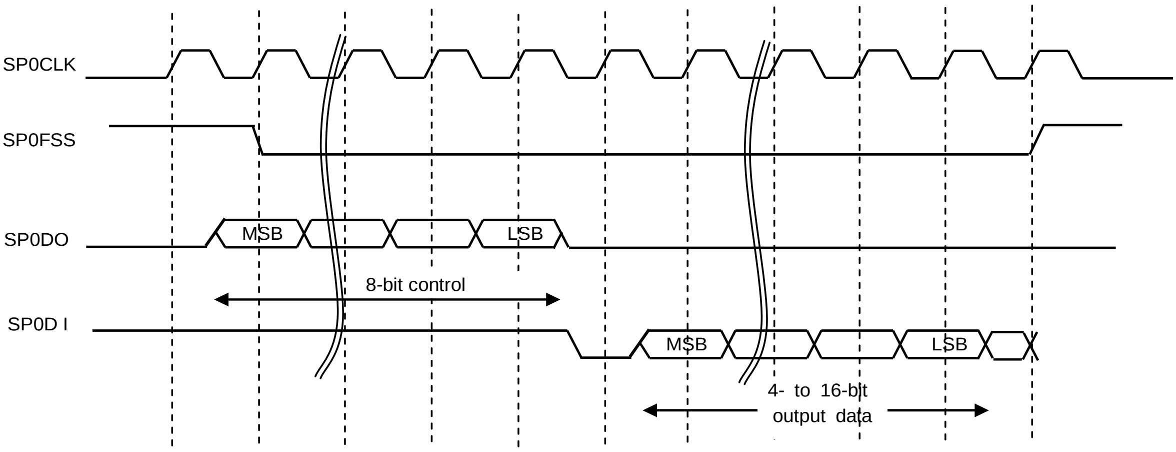
- The SP0CLK signal is forcedly set to LOW.
- SP0FSS is forcedly set to HIGH.
- The transmit data line SP0DO is set to LOW.

If the SSP is enabled and valid data exists in the transmit FIFO, the SP0FSS master signal driven by LOW notifies of the start of transmission. This enables the slave data in the SP0DI input line of the master.

When a half of the SP0CLK period has passed, valid master data is transferred to the SP0DO pin. Both the master data and slave data are now set. When another half of SP0CLK has passed, the SP0CLK master clock pin becomes HIGH. After that, the data is captured at the rising edge of the SP0CLK signal and transmitted at its falling edge. In the single word transfer, the SP0FSS line will return to the idle HIGH state when all the bits of that data word have been transferred, and then 1 cycle of SP0CLK has passed after the last bit was captured. However, for continuous transfer, the SP0FSS signal must be pulsed at HIGH between individual data word transfers. This is because change is not enabled when the slave selection pin freezes data in its peripheral register and the <SPH> bit is logical 0. Therefore, to enable writing of serial peripheral data, the master device must drive the SP0FSS pin of the slave device between individual data transfers. When the continuous transfer is complete, the SP0FSS pin will return to the idle state when one cycle of SP0CLK has passed after the last bit is captured.

(c) Microwire frame format

Microwire frame format (single transfer)



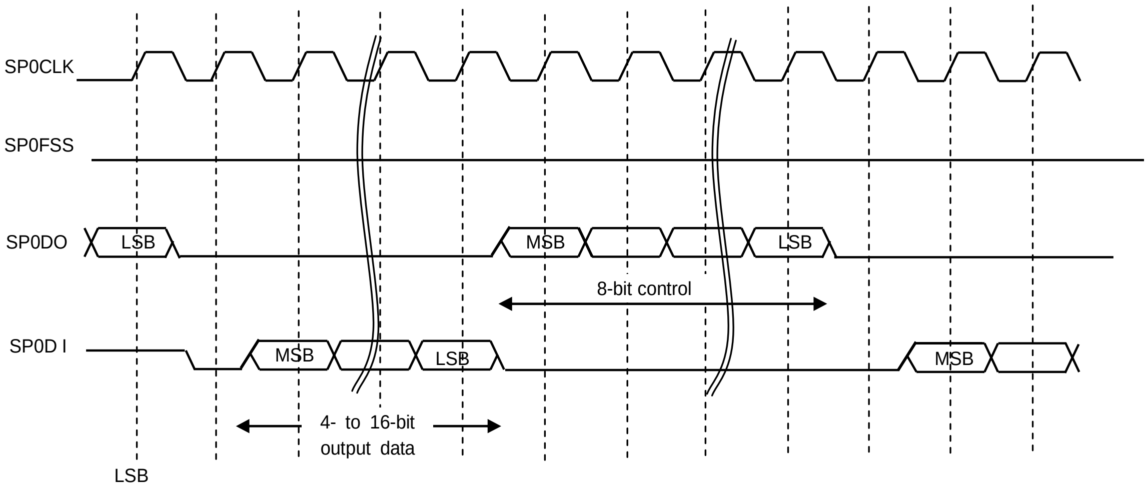
Though the Microwire format is similar to the SPI format, it uses the master/slave message transmission method for half-duplex communications. Each serial transmission is started by an 8-bit control word, which is sent to the off-chip slave device. During this transmission, the SSP does not receive input data. After the message has been transmitted, the off-chip slave decodes it, and after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the requested data. The returned data can be 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits. With this configuration, during the idle period:

- The SP0CLK signal is forcedly set to LOW.
- SP0FSS is forcedly set to HIGH.
- The transmit data line SP0DO is set to LOW.

A transmission is triggered by writing a control byte to the transmit FIFO. The falling edge of SP0FSS causes the value stored in the bottom entry of the transmit FIFO to be transferred to the serial shift register for the transmit logic, and the MSB of the 8-bit control frame to be shifted out onto the SP0DO pin. SP0FSS remains LOW and the SP0D1 pin remains tristated during this transmission. The off-chip serial slave device latches each control bit into its serial shifter on the rising edge of each SP0CLK. After the last bit is latched by the slave device, the control byte is decoded during a one clock wait-state, and the slave responds by transmitting data back to the SSP. Each bit is driven onto SP0DI line on the falling edge of SP0CLK. The SSP in turn latches each bit on the rising edge of SP0CLK. At the end of the frame, for single transfers, the SP0FSS signal is pulled HIGH one clock period after the last bit has been latched in the receive serial shifter, which causes the data to be transferred to the receive FIFO.

Note) The off-chip slave device can tristate the receive line either on the falling edge of SP0CLK after the LSB has been latched by the receive shifter, or when the SP0FSS pin goes HIGH.

Microwire frame format (continuous transfer)



For continuous transfers, data transmission begins and ends in the same manner as a single transfer. However, the SP0FSS line is continuously asserted (held LOW) and transmission of data occurs back to back. The control byte of the next frame follows directly after the LSB of the received data from the current frame. Each of the received values is transferred from the receive shifter on the falling edge of SP0CLK, after the LSB of the frame has been latched into the SSP.

Note) [Example of connection]

The SSP does not support dynamic switching between the master and slave in the system. Each sample SSP is configured and connected as either a master or slave.

(5) DMA interface

The DMA operation of the SSP is controlled through the DMA control register, SP0DMACR.

When there are more data than the watermark level (half of the FIFO) in the receive FIFO, the receive DMA request is asserted.

When the amount of data left in the receive FIFO is less than the watermark level (half of the FIFO), the transmit DMA request is asserted.

To clear the transmit/receive DMA request, an input pin for the transmit/receive DMA request clear signals, which are asserted by the DMA controller, is provided.

Set the DMA burst length to 4 words.

* For the remaining three characters, the SSP does not assert the burst request.

Each request signal remains asserted until the relevant DMA clear signal is asserted. After the request clear signal is deasserted, a request signal can become active again, depending on the conditions described above. All request signals are deasserted if the SSP is disabled or the DMA enable signal is cleared.

The following table shows the trigger points for DMABREQ, for both the transmit and receive FIFOs.

	Burst length	
Watermark level	Transmit (number of empty locations)	Receive (number of filled locations)
1/2	4	4

(Sample program) Sample DMAC settings for SSP0

		[DMA transfer data of 8 words]
(DMACConfiguration) ←	0x00000001	; AHB M1/M2 little endian, DMA enable
(DMACIntTCClear) ←	0x000000FF	; Terminal count int clear
(DMACIntErrClr) ←	0x000000FF	; Error int clear
(1) Memory to Peripheral		
(DMACCxSrcAddr) ←	0xXXXXXXXX	; DMACCxSrcAddr : RAM
(DMACCxDestAddr) ←	0x4001D008	; DMACCxDestAddr : SSP0DR
(DMACCxControl) ←	0x84492008	; Chanel Control
(DMACCxConfiguration) ←	0x0000C9D1	; Config Mem to Pre
(2) Peripheral to Memory		
(DMACCxSrcAddr) ←	0x4001D008	; DMACCxSrcAddr : SSP0DR
(DMACCxDestAddr) ←	0xXXXXXXXX	; DMACCxDestAddr : RAM
(DMACCxControl) ←	0x88489008	; Chanel Control
(DMACCxConfiguration) ←	0x0000D1D1	; Config Pre to Mem

3.2.4 Explanation of the Register

The following lists the SFRs:

- SSP0

base address = 0x4001_D000

Register Name	Address (base+)	Description
SSP0CR0	0x0000	Control register 0
SSP0CR1	0x0004	Control register 1
SSP0DR	0x0008	Receive FIFO (read) and transmit FIFO data register (write)
SSP0SR	0x000C	Status register
SSP0CPSR	0x0010	Clock prescale register
SSP0IMSC	0x0014	Interrupt enable/disable register
SSP0RIS	0x0018	Pre-enable interrupt status register
SSP0MIS	0x001C	Post-enable interrupt status register
SSP0ICR	0x0020	Interrupt clear register
SSP0DMACR	0x0024	DMA control register
−	0x0028 ~ 0xFFC	Reserved

- SSP1

base address = 0x4001_E000

Register Name	Address (base+)	Description
SSP1CR0	0x0000	Control register 0
SSP1CR1	0x0004	Control register 1
SSP1DR	0x0008	Receive FIFO (read) and transmit FIFO data register (write)
SSP1SR	0x000C	Status register
SSP1CPSR	0x0010	Clock prescale register
SSP1IMSC	0x0014	Interrupt enable/disable register
SSP1RIS	0x0018	Pre-enable interrupt status register
SSP1MIS	0x001C	Post-enable interrupt status register
SSP1ICR	0x0020	Interrupt clear register
SSP1DMACR	0x0024	DMA control register
−	0x0028 ~ 0xFFC	Reserved

- SSP2

base address = 0x4001_F000

Register Name	Address (base+)	Description
SSP2CR0	0x0000	Control register 0
SSP2CR1	0x0004	Control register 1
SSP2DR	0x0008	Receive FIFO (read) and transmit FIFO data register (write)
SSP2SR	0x000C	Status register
SSP2CPSR	0x0010	Clock prescale register
SSP2IMSC	0x0014	Interrupt enable/disable register
SSP2RIS	0x0018	Pre-enable interrupt status register
SSP2MIS	0x001C	Post-enable interrupt status register
SSP2ICR	0x0020	Interrupt clear register
SSP2DMACR	0x0024	DMA control register
–	0x0028 ~ 0xFFC	Reserved

- SSP3

base address = 0x4002_0000

Register Name	Address (base+)	Description
SSP3CR0	0x0000	Control register 0
SSP3CR1	0x0004	Control register 1
SSP3DR	0x0008	Receive FIFO (read) and transmit FIFO data register (write)
SSP3SR	0x000C	Status register
SSP3CPSR	0x0010	Clock prescale register
SSP3IMSC	0x0014	Interrupt enable/disable register
SSP3RIS	0x0018	Pre-enable interrupt status register
SSP3MIS	0x001C	Post-enable interrupt status register
SSP3ICR	0x0020	Interrupt clear register
SSP3DMACR	0x0024	DMA control register
–	0x0028 ~ 0xFFC	Reserved

1. SSP0CR0 (SSP0 control register 0)

Address = (0x4001_D000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:8]	SCR	R/W	0y0	For serial clock rate setting Parameter: [Refer to Explanation]) 0x00 ~ 0xFF
[7]	SPH	R/W	0y0	SPCLK phase (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[6]	SPO	R/W	0y0	SPCLK polarity (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[5:4]	FRF	R/W	0y00	Frame format: 0y00: Motorola SPI frame format 0y01: TI synchronous Serial frame format 0y10: National Microwire Frame format 0y11: Reserved, undefined operation
[3:0]	DSS	R/W	0y0000	Data size select: 0y0000: Reserved, undefined operation 0y0001: Reserved, undefined operation 0y0010: Reserved, undefined operation 0y0011: 4-bit data 0y0100: 5-bit data 0y0101: 6-bit data 0y0110: 7-bit data 0y0111: 8-bit data 0y1000: 9-bit data 0y1001: 10-bit data 0y1010: 11-bit data 0y1011: 12-bit data 0y1100: 13-bit data 0y1101: 14-bit data 0y1110: 15-bit data 0y1111: 16-bit data

2. SSP1CR0 (SSP1 control register 0)

Address = (0x4001_E000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:8]	SCR	R/W	0y0	For serial clock rate setting Parameter: 0x00~0xFF
[7]	SPH	R/W	0y0	SPCLK phase (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[6]	SPO	R/W	0y0	SPCLK polarity (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[5:4]	FRF	R/W	0y00	Frame format: 0y00: Motorola SPI frame format 0y01: TI synchronous Serial frame format 0y10: National Microwire Frame format 0y11: Reserved, undefined operation
[3:0]	DSS	R/W	0y0000	Data size select: 0y0000: Reserved, undefined operation 0y0001: Reserved, undefined operation 0y0010: Reserved, undefined operation 0y0011: 4-bit data 0y0100: 5-bit data 0y0101: 6-bit data 0y0110: 7-bit data 0y0111: 8-bit data 0y1000: 9-bit data 0y1001: 10-bit data 0y1010: 11-bit data 0y1011: 12-bit data 0y1100: 13-bit data 0y1101: 14-bit data 0y1110: 15-bit data 0y1111: 16-bit data

3. SSP2CR0 (SSP2 control register 0)

Address = (0x4001_F000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:8]	SCR	R/W	0y0	For serial clock rate setting Parameter: 0x00~0xFF
[7]	SPH	R/W	0y0	SPCLK phase (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[6]	SPO	R/W	0y0	SPCLK polarity (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[5:4]	FRF	R/W	0y00	Frame format: 0y00: Motorola SPI frame format 0y01: TI synchronous Serial frame format 0y10: National Microwire Frame format 0y11: Reserved, undefined operation
[3:0]	DSS	R/W	0y0000	Data size select: 0y0000: Reserved, undefined operation 0y0001: Reserved, undefined operation 0y0010: Reserved, undefined operation 0y0011: 4-bit data 0y0100: 5-bit data 0y0101: 6-bit data 0y0110: 7-bit data 0y0111: 8-bit data 0y1000: 9-bit data 0y1001: 10-bit data 0y1010: 11-bit data 0y1011: 12-bit data 0y1100: 13-bit data 0y1101: 14-bit data 0y1110: 15-bit data 0y1111: 16-bit data

4. SSP3CR0 (SSP3 control register 0)

Address = (0x4002_0000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:8]	SCR	R/W	0y0	For serial clock rate setting Parameter: 0x00~0xFF
[7]	SPH	R/W	0y0	SPCLK phase (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[6]	SPO	R/W	0y0	SPCLK polarity (applicable to Motorola SPI frame format only, Refer to [Motorola SPI frame format])
[5:4]	FRF	R/W	0y00	Frame format: 0y00: Motorola SPI frame format 0y01: TI synchronous Serial frame format 0y10: National Microwire Frame format 0y11: Reserved, undefined operation
[3:0]	DSS	R/W	0y0000	Data size select: 0y0000: Reserved, undefined operation 0y0001: Reserved, undefined operation 0y0010: Reserved, undefined operation 0y0011: 4-bit data 0y0100: 5-bit data 0y0101: 6-bit data 0y0110: 7-bit data 0y0111: 8-bit data 0y1000: 9-bit data 0y1001: 10-bit data 0y1010: 11-bit data 0y1011: 12-bit data 0y1100: 13-bit data 0y1101: 14-bit data 0y1110: 15-bit data 0y1111: 16-bit data

[Explanation]

a. <SCR>

Used to generate the SPP transmit bit rate and receive bit rate.

This bit rate can be obtained by the following equation:
Bit rate = f_{PCLK} / (CPSDVSR × (1+SCR))
CPSDVSR is an even number between 2 to 254, which is programmed by the SSPxCPSR register, and SCR takes a value between 0 to 255.

5. SSP0CR1 (SSP0 control register 1)

Address = (0x4001_D000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	SOD	R/W	0y0	Slave mode SP0DO output control: 0y0: Enable 0y1: Disable
[2]	MS	R/W	0y0	Master/slave mode select: 0y0: Device configured as a master 0y1: Device configured as a slave
[1]	SSE	R/W	0y0	SSP0 enable: 0y0: Disable 0y1: Enable
[0]	LBM	R/W	0y0	Loop back mode: 0y0: Normal serial port operation enabled 0y1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

6. SSP1CR1 (SSP1 control register 1)

Address = (0x4001_E000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	SOD	R/W	0y0	Slave mode SP0DO output control: 0y0: Enable 0y1: Disable
[2]	MS	R/W	0y0	Master/slave mode select: 0y0: Device configured as a master 0y1: Device configured as a slave
[1]	SSE	R/W	0y0	SSP1 enable: 0y0: Disable 0y1: Enable
[0]	LBM	R/W	0y0	Loop back mode: 0y0: Normal serial port operation enabled 0y1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

7. SSP2CR1 (SSP2 control register 1)

Address = (0x4001_F000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	SOD	R/W	0y0	Slave mode SP0DO output control: 0y0: Enable 0y1: Disable
[2]	MS	R/W	0y0	Master/slave mode select: 0y0: Device configured as a master 0y1: Device configured as a slave
[1]	SSE	R/W	0y0	SSP2 enable: 0y0: Disable 0y1: Enable
[0]	LBM	R/W	0y0	Loop back mode: 0y0: Normal serial port operation enabled 0y1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

8. SSP3CR1 (SSP3 control register 1)

Address = (0x4002_0000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	SOD	R/W	0y0	Slave mode SP0DO output control: 0y0: Enable 0y1: Disable
[2]	MS	R/W	0y0	Master/slave mode select: 0y0: Device configured as a master 0y1: Device configured as a slave
[1]	SSE	R/W	0y0	SSP3 enable: 0y0: Disable 0y1: Enable
[0]	LBM	R/W	0y0	Loop back mode: 0y0: Normal serial port operation enabled 0y1: Output of transmit serial shifter is connected to input of receive serial shifter internally.

[Explanation]

a. <SOD>

Slave mode output disable. This bit is relevant only in the slave mode (<MS>=1).

9. SSP0DR (SSP0 data register)

Address = (0x4001_D000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	DATA	R/W	0x0000	Transmit/receive FIFO data: 0x00 ~ 0xFF

10. SSP1DR (SSP1 data register)

Address = (0x4001_E000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	DATA	R/W	0x0000	Transmit/receive FIFO data: 0x00 ~ 0xFF

11. SSP2DR (SSP2 data register)

Address = (0x4001_F000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	DATA	R/W	0x0000	Transmit/receive FIFO data: 0x00 ~ 0xFF

12. SSP3DR (SSP3 data register)

Address = (0x4002_0000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read undefined. Write as zero.
[15:0]	DATA	R/W	0x0000	Transmit/receive FIFO data: 0x00 ~ 0xFF

[Explanation]

a. <DATA>

Read: Receive FIFO

Write: Transmit FIFO

You must right-justify data when the SSP is programmed for a data size that is less than 16 bits. Unused bits at the top are ignored by transmit logic. The receive logic automatically right-justifies.

13. SSP0SR (SSP0 status register)

Address = (0x4001_D000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read undefined. Write as zero.
[4]	BSY	RO	0y0	Busy flag 0y0: Idle 0y1: Busy
[3]	RFF	RO	0y0	Receive FIFO full: 0y0: Receive FIFO is not full 0y1: Receive FIFO is full
[2]	RNE	RO	0y0	Receive FIFO empty flag 0y0: Receive FIFO is empty 0y1: Receive FIFO is not empty
[1]	TNF	RO	0y1	Transmit FIFO full flag: 0y0: Transmit FIFO is full 0y1: Transmit FIFO is not full
[0]	TFE	RO	0y1	Transmit FIFO empty flag: 0y0: Transmit FIFO is not empty 0y1: Transmit FIFO is empty

14. SSP1SR (SSP1 status register)

Address = (0x4001_E000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read undefined. Write as zero.
[4]	BSY	RO	0y0	Busy flag 0y0: Idle 0y1: Busy
[3]	RFF	RO	0y0	Receive FIFO full: 0y0: Receive FIFO is not full 0y1: Receive FIFO is full
[2]	RNE	RO	0y0	Receive FIFO empty flag 0y0: Receive FIFO is empty 0y1: Receive FIFO is not empty
[1]	TNF	RO	0y1	Transmit FIFO full flag: 0y0: Transmit FIFO is full 0y1: Transmit FIFO is not full
[0]	TFE	RO	0y1	Transmit FIFO empty flag: 0y0: Transmit FIFO is not empty 0y1: Transmit FIFO is empty

15. SSP2SR (SSP2 status register)

Address = (0x4001_F000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read undefined. Write as zero.
[4]	BSY	RO	0y0	Busy flag 0y0: Idle 0y1: Busy
[3]	RFF	RO	0y0	Receive FIFO full: 0y0: Receive FIFO is not full 0y1: Receive FIFO is full
[2]	RNE	RO	0y0	Receive FIFO empty flag 0y0: Receive FIFO is empty 0y1: Receive FIFO is not empty
[1]	TNF	RO	0y1	Transmit FIFO full flag: 0y0: Transmit FIFO is full 0y1: Transmit FIFO is not full
[0]	TFE	RO	0y1	Transmit FIFO empty flag: 0y0: Transmit FIFO is not empty 0y1: Transmit FIFO is empty

16. SSP3SR (SSP3 status register)

Address = (0x4002_0000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read undefined. Write as zero.
[4]	BSY	RO	0y0	Busy flag 0y0: Idle 0y1: Busy
[3]	RFF	RO	0y0	Receive FIFO full: 0y0: Receive FIFO is not full 0y1: Receive FIFO is full
[2]	RNE	RO	0y0	Receive FIFO empty flag 0y0: Receive FIFO is empty 0y1: Receive FIFO is not empty
[1]	TNF	RO	0y1	Transmit FIFO full flag: 0y0: Transmit FIFO is full 0y1: Transmit FIFO is not full
[0]	TFE	RO	0y1	Transmit FIFO empty flag: 0y0: Transmit FIFO is not empty 0y1: Transmit FIFO is empty

[Explanation]

a. <BSY>

BSY="1" indicates that the SSP is currently transmitting and/or receiving a frame or the transmit FIFO is not empty.

17. SSP0CPSR (SSP0 clock prescale register)

Address = (0x4001_D000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	CPSDVSR	R/W	0x0000	Clock prescale divider: Set an even number from 2 to 254.

18. SSP1CPSR (SSP1 clock prescale register)

Address = (0x4001_E000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	CPSDVSR	R/W	0x0000	Clock prescale divider: Set an even number from 2 to 254.

19. SSP2CPSR (SSP2 clock prescale register)

Address = (0x4001_F000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	CPSDVSR	R/W	0x0000	Clock prescale divider: Set an even number from 2 to 254.

20. SSP3CPSR (SSP3 clock prescale register)

Address = (0x4002_0000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read undefined. Write as zero.
[7:0]	CPSDVSR	R/W	0x0000	Clock prescale divider: Set an even number from 2 to 254.

[Explanation]

a. <CPSDVSR>

Clock prescale divider. Must be an even number from 2 to 254, depending on the frequency of PCLK. The least significant bit always returns zero on reads.

21. SSP0IMSC (SSP0 interrupt enable/disable register)

Address = (0x4001_D000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXIM	R/W	0y0	Transmit FIFO interrupt enable: 0y0: Disable 0y1: Enable
[2]	RXIM	R/W	0y0	Receive FIFO interrupt enable: 0y0: Disable 0y1: Enable
[1]	RTIM	R/W	0y0	Receive timeout interrupt enable: 0y0: Disable 0y1: Enable
[0]	RORIM	R/W	0y0	Receive overrun interrupt enable: 0y0: Disable 0y1: Enable

22. SSP1IMSC (SSP1 interrupt enable/disable register)

Address = (0x4001_E000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXIM	R/W	0y0	Transmit FIFO interrupt enable: 0y0: Disable 0y1: Enable
[2]	RXIM	R/W	0y0	Receive FIFO interrupt enable: 0y0: Disable 0y1: Enable
[1]	RTIM	R/W	0y0	Receive timeout interrupt enable: 0y0: Disable 0y1: Enable
[0]	RORIM	R/W	0y0	Receive overrun interrupt enable: 0y0: Disable 0y1: Enable

23. SSP2IMSC (SSP2 interrupt enable/disable register)

Address = (0x4001_F000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXIM	R/W	0y0	Transmit FIFO interrupt enable: 0y0: Disable 0y1: Enable
[2]	RXIM	R/W	0y0	Receive FIFO interrupt enable: 0y0: Disable 0y1: Enable
[1]	RTIM	R/W	0y0	Receive timeout interrupt enable: 0y0: Disable 0y1: Enable
[0]	RORIM	R/W	0y0	Receive overrun interrupt enable: 0y0: Disable 0y1: Enable

24. SSP3IMSC (SSP3 interrupt enable/disable register)

Address = (0x4002_0000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXIM	R/W	0y0	Transmit FIFO interrupt enable: 0y0: Disable 0y1: Enable
[2]	RXIM	R/W	0y0	Receive FIFO interrupt enable: 0y0: Disable 0y1: Enable
[1]	RTIM	R/W	0y0	Receive timeout interrupt enable: 0y0: Disable 0y1: Enable
[0]	RORIM	R/W	0y0	Receive overrun interrupt enable: 0y0: Disable 0y1: Enable

[Explanation]

a. <TXIM>

Enables/disables transmit interrupt

b. <RXIM>

Enables/disables receive interrupt

c. <RTIM>

Enables/disables timeout interrupt

d. <RORIM>

Enables/disables receive overrun interrupt

25. SSP0RIS (SSP0 pre-enable interrupt status register)

Address = (0x4001_D000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXRIS	RO	0y1	Pre-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXRIS	RO	0y0	Pre-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTRIS	RO	0y0	Pre-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORRIS	RO	0y0	Pre-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

26. SSP1RIS (SSP1 pre-enable interrupt status register)

Address = (0x4001_E000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXRIS	RO	0y1	Pre-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXRIS	RO	0y0	Pre-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTRIS	RO	0y0	Pre-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORRIS	RO	0y0	Pre-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

27. SSP2RIS (SSP2 pre-enable interrupt status register)

Address = (0x4001_F000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXRIS	RO	0y1	Pre-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXRIS	RO	0y0	Pre-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTRIS	RO	0y0	Pre-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORRIS	RO	0y0	Pre-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

28. SSP3RIS (SSP3 pre-enable interrupt status register)

Address = (0x4002_0000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXRIS	RO	0y1	Pre-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXRIS	RO	0y0	Pre-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTRIS	RO	0y0	Pre-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORRIS	RO	0y0	Pre-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

29. SSP0MIS (SSP0 post-enable interrupt status register)

Address = (0x4001_D000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXMIS	RO	0y0	Post-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXMIS	RO	0y0	Post-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTMIS	RO	0y0	Post-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORMIS	RO	0y0	Post-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

30. SSP1MIS (SSP1 post-enable interrupt status register)

Address = (0x4001_E000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXMIS	RO	0y0	Post-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXMIS	RO	0y0	Post-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTMIS	RO	0y0	Post-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORMIS	RO	0y0	Post-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

31. SSP2MIS (SSP2 post-enable interrupt status register)

Address = (04001_F000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXMIS	RO	0y0	Post-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXMIS	RO	0y0	Post-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTMIS	RO	0y0	Post-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORMIS	RO	0y0	Post-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

32. SSP3MIS (SSP3 post-enable interrupt status register)

Address = (0x4002_0000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read undefined. Write as zero.
[3]	TXMIS	RO	0y0	Post-enable transmit interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[2]	RXMIS	RO	0y0	Post-enable receive interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[1]	RTMIS	RO	0y0	Post-enable receive timeout interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present
[0]	RORMIS	RO	0y0	Post-enable receive overrun interrupt flag: 0y0: Interrupt not present 0y1: Interrupt present

33. SSP0ICR (SSP0 interrupt clear register)

Address = (0x4001_D000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	RTIC	WO	Undefined	Clear the receive timeout interrupt flag: 0y0: Do nothing 0y1: Clear
[0]	RORIC	WO	Undefined	Clear the receive overrun interrupt flag: 0y0: Do nothing 0y1: Clear

34. SSP1ICR (SSP1 interrupt clear register)

Address = (0x4001_E000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	RTIC	WO	Undefined	Clear the receive timeout interrupt flag: 0y0: Do nothing 0y1: Clear
[0]	RORIC	WO	Undefined	Clear the receive overrun interrupt flag: 0y0: Do nothing 0y1: Clear

35. SSP2ICR (SSP2 interrupt clear register)

Address = (0x4001_F000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	RTIC	WO	Undefined	Clear the receive timeout interrupt flag: 0y0: Do nothing 0y1: Clear
[0]	RORIC	WO	Undefined	Clear the receive overrun interrupt flag: 0y0: Do nothing 0y1: Clear

36. SSP3ICR (SSP3 interrupt clear register)

Address = (0x4002_0000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	RTIC	WO	Undefined	Clear the receive timeout interrupt flag: 0y0: Do nothing 0y1: Clear
[0]	RORIC	WO	Undefined	Clear the receive overrun interrupt flag: 0y0: Do nothing 0y1: Clear

37. SSP0DMACR (SSP0DMA control register)

Address = (0x4001_D000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA control: 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA control: 0y0: Disable 0y1: Enable

38. SSP1DMACR (SSP1DMA control register)

Address = (0x4001_E000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA control: 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA control: 0y0: Disable 0y1: Enable

39. SSP2DMACR (SSP2DMA control register)

Address = (0x4001_F000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA control: 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA control: 0y0: Disable 0y1: Enable

40. SSP3DMACR (SSP3DMA control register)

Address = (0x4002_0000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA control: 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FIFO DMA control: 0y0: Disable 0y1: Enable

- Notes related to specifications

(1) When correct data reception is disturbed due to clock phase shift during reception: After disabling SSP, clearing all data in the receive FIFO and then enabling SSP again will restore the correct reception status.

Example: How to restore a receive data error

```
(SSP0CR1) ←          ((SSP0CR1)&(0xFFFFFFFD))       ; Set "0" to SSP0CR1<SSE>.
                                                    Sync serial port disable

(GPIOAFR1) ←         ((GPIOAFR1)&(0xFFFFFFF0))      ; Set "0" to GPIOAFR1.
                                                    Port A SSP0 function disable

while((SSP0SR)&0x00000004)!=0x00000000){            ; Read (RNE="0")SSP0DR until the receive FIFO
Reg ← (SSP0DR)}                                     becomes empty.

(GPIOAFR1) ←         ((GPIOAFR1)|(0x0000000F))      ; Set "1" to GPIOAFR1.
                                                     Port A SSP0 function enable

(SSP0CR1) ←          ((SSP0CR1)|(0x00000002))       ; Set "1" to SSP0CR1<SSE>.
                                                    Sync serial port enable
```

3.16 USB Host Controller

The USB Host Controller (USBHC), which conforms to the USB2.0 Specification Revision 2.0, the Enhanced Host Controller Interface (EHCI) Specification Revision 1.0, and the Open Host Controller Interface (OHCI) For USB release 1.0a, is capable of USB transfer at 480 Mbps (high speed) and 12 Mbps (full speed). Note that the following optional functions are not supported.

1. OHCI LEGACY SUPPORT
2. EHCI 64-bit Addressing
3. EHCI Debug Port

This manual describes USB PHY as the scope of this chapter.

3.16.1 Abbreviations and Terms

Terminology	Description
OHCI	Open Host Controller Interface
EHCI	Enhanced Host Controller Interface
PHY	Physical layer to generate and receive mostly differential signals of USB communication
HCD	Host Controller Driver. Software to control this USBHC.
Park mode	The USBHC, which executes a HS bulk/control transfer descriptor by reading it from memory, is the function of executing this same descriptor for multiple times continuously. In this product, executing three continuous times is set by default.
Out transfer	Data is transferred from the USBHC to USB devices. The USBHC asserts a bus request to read transfer data from system memory and stores it in the packet buffer. This transfer data is transmitted to USB devices.
In transfer	Data is transferred from a USB device to the USBHC. Data received from a USB device is stored in the packet buffer. The USBHC asserts the bus request to write the data in system memory.

3.16.2 System Overview

The main features of the USBHC are as follows:

(1) Can connect to 480 Mbps (high-speed) and 12 Mbps (full-speed) USB devices.
Low-speed is not supported.

(2) Supports control, bulk, interrupt, and isochronous transfers.

(3) The connection interface for the ARM core is the AHB. Conforms to AMBA Specification revision 2.0. For USB communication, the USBHC serves as the bus master to directly access to the built-in SRAM.

(4) Uses the SRAM4 area for the descriptors and communication areas used in USB communication. Refer to 3.3, "Memory Map" and 3.5.4, "Bus Configuration."

(5) 45-Ω termination resistors, 1.5-kΩ pull-up resistors, and 15-kΩ pull-down resistors are built into the PHY.

3.16.3 System Configuration

Figure 3-16-1 shows the block diagram of the USBHC. The USBHC consists of the EHCI Host Controller block which handles high-speed communication, and the OHCI Host Controller block which handles full-speed communication.

For the external I/F, a USB transceiver (USB-use, PHY2.0) is used.

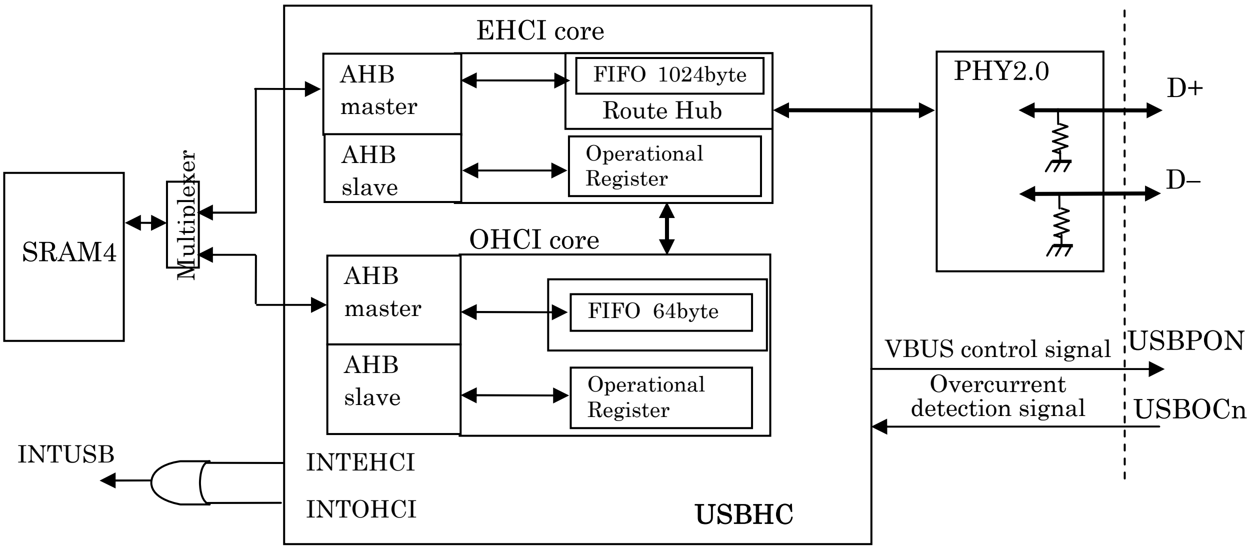


Figure 3.16-1 Block Diagram of USBHC

3.16.4 Interrupts

In the USBHC, each of the OHCI and EHCI has one interrupt output each to the ARM core. This signal is integrated into one to be input to the NVIC. The following shows the interrupt sources:

(1) OHCI interrupt sources

- Scheduling Overrun
- HcDoneHead Write back
- Start of Frame
- Rsume Detect
- Unrecoverable Error
- Frame Number Overflow
- Root Hub Status Change
- Ownership Change

Meeting interrupt conditions causes the USBHC to set the corresponding bit in the internally provided HcInterruptStatus register. When this bit is set, if the MasterInterruptEnable (MIE) bit in the HcInterruptEnable register is enabled and this bit is enabled in the HcInterruptEnable register, an OHCI interrupt occurs.

The USBHC driver software can clear this bit by writing a '1' in the corresponding bit in the HcInterruptStatus register to deassert the interrupt request. Since the hardware does not clear the bit automatically, clearing it is always needed by software.

(2) EHCI interrupt sources

- Interrupt on Async Advance
- Host System Error
- Frame List Rollover
- Port Change Interrupt
- USB Error Interrupt
- USB Interrupt

Meeting interrupt conditions causes the USBHC to set the corresponding bit in the internally provided USBSTS register. When this bit is set, if the corresponding bit is enabled in the USBINTR register, an EHCI interrupt occurs. The USBHC driver software can clear this bit by writing a '1' in the corresponding bit in the USBSTS register to deassert the interrupt request. Since the hardware does not clear the bit automatically, clearing it is always needed by software.

3.16.5 Reset

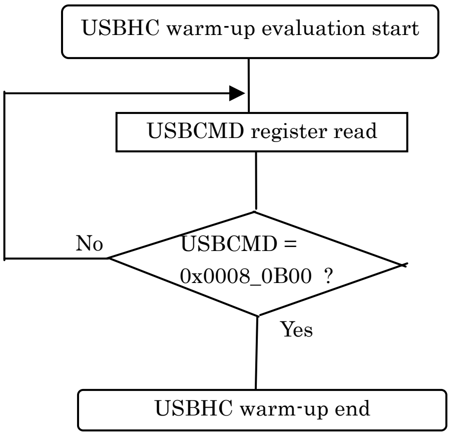
The USBHC is initialized by system reset or software reset.

3.16.5.1 System Reset

System reset is caused by the external reset terminal or by watchdog timer reset followed by SYSRESETREQ to the CPU.

- All registers are initialized.
- The USBHC outputs a reset signal onto the USB bus (DP=DM=0).
- The status of the USB transitions to the USBRESET status.
- The USBHC is disabled for list processing and SOF token generation.
- The FrameNumber of the HcFmNumber register does not increase.
- To stabilize the PLL clock in the PHY, approximately 1 ms of warming-up period is needed, during which the USBHC and the PHY are kept reset. Therefore, do not access registers during this period.

After the warming-up period has ended, evaluation is available by reading the register as shown below:



This is achieved by USBCMD reading all zeros during a reset and reading 0x0008_0B00 after the reset ends.

3.16.5.2 Software Reset

(1) OHCI

Software reset is caused by writing a '1' in the HostControllerReset bit in the HcCommandStatus register.

- All registers of the OHCI are initialized.
  However, the RemoteWakeupConnected bit and InterruptRouting bit in the HcControl register are not initialized.
- The USBHC outputs a reset signal onto the USB bus (DP=DM=0).
- The status of the USBHC transitions to USBSUSPEND
  (The FunctionalState bit in the HcControl register is set to 0x03 to transition to the USBSUSPEND state).

(2) EHCI

Software reset is caused by writing a '1' in the HCRESET bit in the USBCMD register.

- The route hub register is initialized with the same effect as when system reset is made. However, the USBHCREG register is not initialized.
- The internal pipelines, timers, counters, state machines, and other values are set to initial values.
- The HCRESET bit in the USBCMD register is cleared to 0 when reset is completed. This is used to evaluate the end of software reset. Note that software writing this bit to 0 cannot cancel reset processing.
- When the HC Halted bit in the USBSTS register is 0, if the HCRESET bit in the USBCMD register is set to 1, an undefined operation results.

3.16.6 Bus Power Control

The USBHC has the control signals of the external VBUS power supply IC and controls it using the USBPON pin.

First, the port control register sets the functions of the pins. After this, in the case of full-speed connection, setting 1 in the LPSC bit in the HcRhStatus register in the OHCI register will output High from the USBPON pin. This product takes GlobalPower control and therefore does not use the PPS bit in the HcRhPortStatus register.

In the case of high-speed connection, the PortPower bit in the PORTSC register in the EHCI register is set to 1.

For detection of overcurrent, the USBOCn pin is used. When the USBHC detects Low from this pin, the OCI bit in HcRhStatus in the OHCI register is set to 1 automatically in the case of full-speed connection. Then, the USBPON pin turns off automatically

(To use it as a USBOCn pin, set the function of the pin by using the port control register).

This product takes GlobalPower control, and therefore the POCI bit in the HcRhPortStatus register is not used (does not change).

In the case of high-speed connection, the Over-current Active bit in the PORTSC register in the EHCI register is set to 1. This product has no auto power-off function by overcurrent detection. Control it with software.

3.16.7 AHB Burst Operation and Transfer Start Condition

After setting the operational register using an AHB slave interface, the USBHC serves as the bus master to make a burst access to system memory for USB transfer.

3.16.7.1 EHCI

<OUT transfer>

Relevant registers

Register name	Description
USBHCREG03	Sets whether to perform an AHB burst transfer continuously or divide it.
USBHCREG01	Sets the division size of AHB burst transfer. It includes the AHB read threshold value and the AHB write threshold value.
	Sets the data transfer start threshold value for a USB device in OUT transfer.

In the default setting (USBHCREG03[0]=0), a single AHB burst transfer reads data from system memory up to the packet buffer size (1024 bytes) irrespective of the transfer size. The settings of the USBHCREG01 register have no effect.

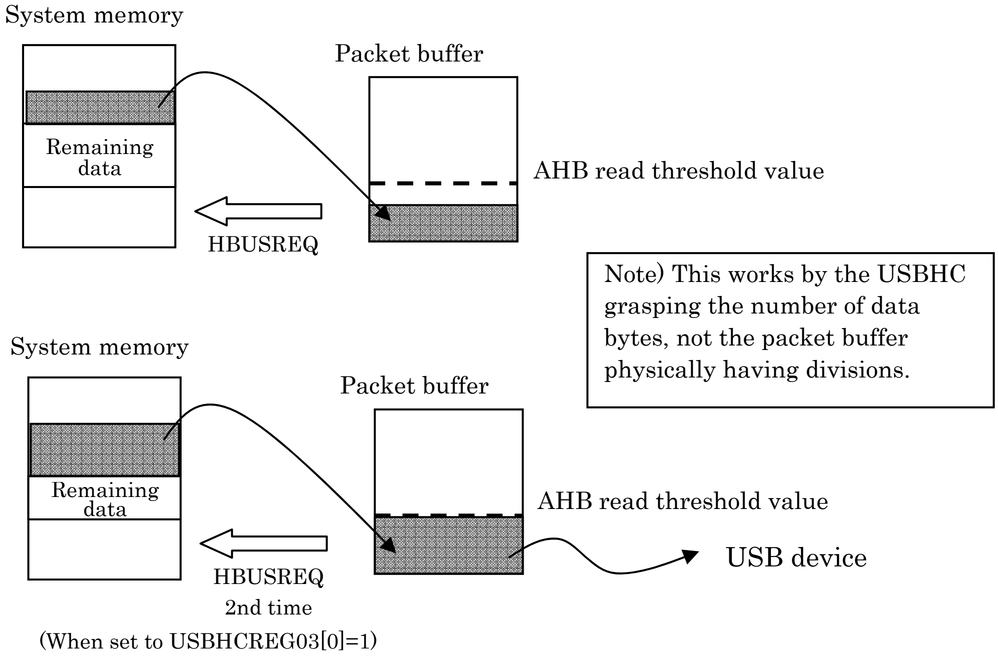
However, if USBHCREG03[0]=1 is set, AHB burst transfer is disconnected at each AHB read threshold value (HBUSREQ is disconnected for a time). To set an AHB read threshold value, set the value in the USBHCREG01 register.
When transmitting data of 1024 bytes and when the USBHCREG01 register is set to 256 bytes, burst transfer is divided at each 256-byte burst transfer. That is, at the point when a read of 256 bytes has been completed, the HBUSREQ signal is deasserted once. Because of this, four burst transfers are performed to transfer 1024 bytes of data. In this product, the USBHCREG01 register is set to 256 bytes by default.

In this product, all AHB burst transfers take undefined length burst transfer.

For more information on the USBHCREG register, refer to item D in Section 3.16.9.

The following describes the condition of transfer start to a USB device. When the data read from system memory exceeds the AHB read threshold value set in the USBHCREG01 register, the packet buffer starts transfer to a USB device. This start condition is always enabled irrespective of the setting of the USBHCREG03 register.

The USBHC grasps the data transfer size by deciphering the descriptor in which detailed information on USB transfer is described. By grasping the number of bytes being transferred, control of the packet buffer is maintained.



Note) This works by the USBHC grasping the number of data bytes, not the packet buffer physically having divisions.

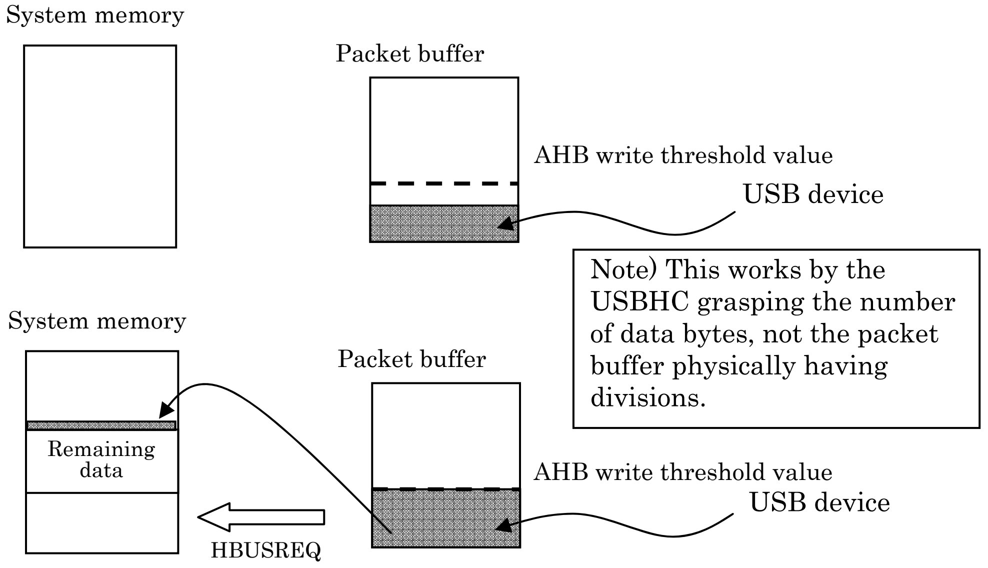
<IN transfer>

In the default setting, when the packet buffer has received data that amounts to 256 bytes, AHB burst transfer is started. When transfer data is smaller than 256 bytes, AHB burst transfer is started after all data has been stored in the packet buffer.

This "256 bytes" is called the AHB write threshold value. You can change the AHB write threshold value in the USBHCREG01 register. When the receive data remaining after AHB burst transfer is started exceeds the AHB write threshold value, the burst transfer is disconnected (HBUSREQ is deasserted) every time it occurs.

In the case of data of 1024 bytes, for example, four burst transfers are performed. Note that the setting of the USBHCREG03 register has no effect on this function.

For more information on the USBHCREG register, refer to item D in Section 3.16.9.



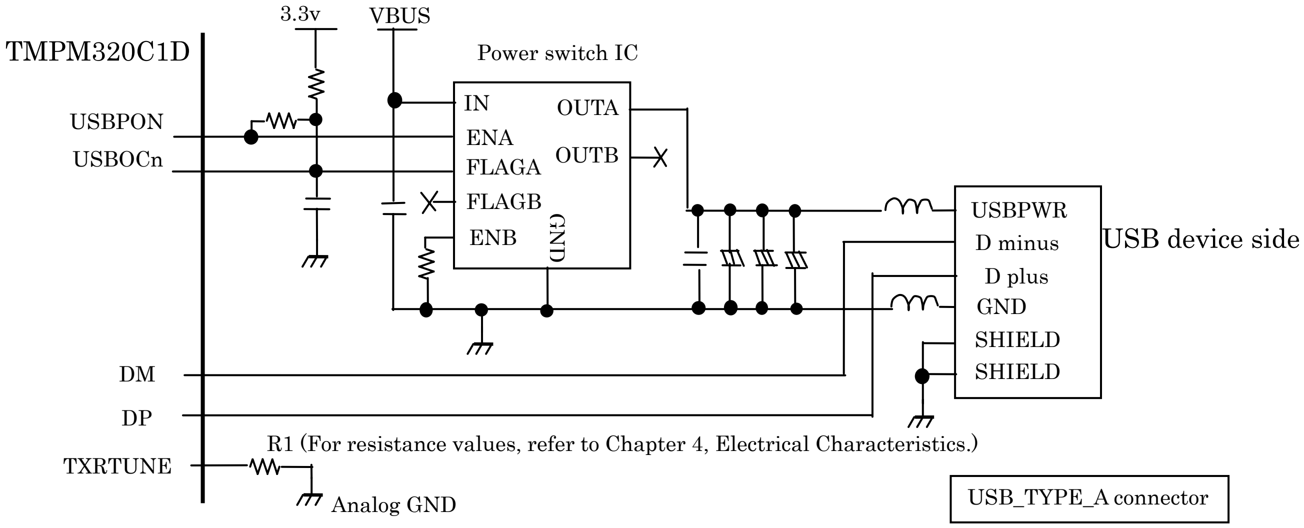
Note) This works by the USBHC grasping the number of data bytes, not the packet buffer physically having divisions.

3.16.7.2 OHCI

As the USB I/F, the OHCI has a FIFO of 64 bytes. As the AHB I/F, there are FIFOs of 16 bytes for read and write each. Because of this, AHB burst transfer takes undefined length burst transfer of up to 4 beats.

There is no setting for threshold value.

3.16.8 Connection Circuit Diagram for Reference



Note) In your PCB, design it so that it satisfies the USB2.0 Standards with reference to the High Speed USB platform design guidelines provided by the USB I/F, and the USB PCB layout guidelines from Application Notes.

3.16.9 Registers

Only the slave interface function and word access (32 bits) are supported.
HTRANS (AHB transfer type) supports only IDLE, BUSY, and NONSEQUENTIAL.

3.16.9.1 OHCI registers

Register map base address = 0x4000_3000

xxx3	xxx2	xxx1	xxx0	Address (base+)
HcRevision				0x0000
HcControl				0x0004
HcCommandStatus				0x0008
HcInterruptStatus				0x000C
HcInterruptEnable				0x0010
HcInterruptDisable				0x0014
HcHCCA				0x0018
HcPeriodCurrentED				0x001C
HcControlHeadED				0x0020
HcControlCurrentED				0x0024
HcBulkHeadED				0x0028
HcBulkCurrentED				0x002C
HcDoneHead				0x0030
HcFmInterval				0x0034
HcFmRemaining				0x0038
HcFmNumber				0x003C
HcPeriodStart				0x0040
HcLSThreshold				0x0044
HcRhDescriptorA				0x0048
HcRhDescripterB				0x004C
HcRhStatus				0x0050
HcRhPortStatus				0x0054
–				0x0058
–				0x005C
–				0x0080
–				0x0084
–				0x0088
–				0x008C
–				0x00C0
–				0x00C4
–				0x00C8
–				0x00CC

Note 1) The addresses shown in the list are the ones mapped on TMPM320C1D.

Note 2) In Open HCI Specification Release 1.0a, the FrameRemaining(FR) bit and FrameRemainingToggle(FRT) bit in the HcFmRemaining register, and the FrameNumber(FN) bit in the HcFmNumber register are read only in the host control driver (HCD). However, the USB 1.1 OHCI host control core enables the HCD to write to these registers for debugging purposes. If the HCD writes in these registers, undefined settings result. Do not write in these bits with the HCD.

Note 3) The following sections of 3.13.6.1HcRevision Register to 3.13.6.22HcRhPortStatus Register are for reference use. For more information on each register in accordance with the OHCI, refer to the specifications of Open HCI Specification Release 1.0a.

Terminology is complemented below for the following register lists.

HCD: HostControllerDriver. HCD refers to the software driver for the Host Controller USB. Read/Write is the access right from the viewpoint of software.

HC: HostController. HC refers to the Host Controller as hardware. When Read/Write is R, the controller circuit itself only sees values; when R/W, the controller circuit itself may make write updates depending on processing irrespective of the software. Example: HcRhStatus register, OCI bit Software cannot write the overcurrent status, whereas the Host Controller circuit automatically updates the status to 1 by overcurrent detection signal input.

HcRevision

Address = (0x4000_3000) + 0x0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved															
Read/Write (HCD)																
Read/Write (HC)																
After reset																
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved								Rev							
Read/Write (HCD)									R							
Read/Write (HC)									R							
After reset									0	0	0	1	0	0	0	0

Bit	Mnemonic	Field name	Description
31:8		Reserved	
7:0	REV	Revision	This read-only field contains the BCD representation of the version of the HCI specification that is implemented by this HC. For example, a value 0x11 corresponds to version 1.1. All of the HC implementations that are compliant with this specification will have a value of 0x10.

HcControl

The HcControl Register defines the operating modes for the Host Controller. Most of the fields in this register are modified only by the Host Controller Driver, except HostControllerFunctionState and RemoteWakeupConnected.

Address = (0x4000_3000) + 0x0004

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved															
Read/Write (HCD)																
Read/Write (HC)																
After reset																

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved					RWE	RWC	IR	HCFS		BLE	CLE	IE	PLE	CBSR	
Read/Write (HCD)						R/W										
Read/Write (HC)						R	R/W	R	R/W		R	R	R	R	R	
After reset						0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31:11		Reserved	
10	RWE	RemoteWakeup Enable	This bit is used by HCD to enable or disable the remote wakeup feature upon the detection of an upstream resume signal. When this bit is set and the ResumeDetect bit in the HcInterruptStatus is set, a remote wakeup is signaled to the host system. Setting this bit has no effect on the generation of hardware interrupts.
9	RWC	RemoteWakeup Connected	This bit indicates whether HC supports remote wakeup signaling. If remote wakeup is supported and used by the system, it is the responsibility of system firmware to set this bit during the POST (Power on Self Test) period. HC clears the bit upon a hardware reset but does not alter it upon software reset.
8	IR	Interrupt Routing	This bit determines the routing of interrupts generated by events registered in the HcInterruptStatus register. If clear, all interrupts are routed to the normal host bus interrupt mechanism. If set, interrupts are routed to the System Management Interrupt. HCD clears this bit upon a hardware reset, but it does not alter this bit upon a software reset. HCD uses this bit as a tag to indicate the ownership of HC.
7:6	HCFS	HostController FunctionalState ForUSB	00: USBRESET 01: USBRESUME 10: USBOPERATIONAL 11: USBSUSPEND A transition to USBOPERATIONAL from another state causes SOF packet generation to begin 1 ms later. HCD determines whether HC has begun sending SOFs by reading the StartofFrame field of the HcInterrupt register. This field can be changed by HC only when in the USBSUSPEND state. HC moves from the USBSUSPEND state to the USBRESUME state after detecting the resume signaling from a downstream port. HC enters the USBSUSPEND state after a software reset, whereas it enters the USBRESET state after a hardware reset. The latter also resets the root hub and asserts subsequent reset signaling to downstream ports.

5	BLE	BulkListEnable	Setting this bit enables the processing of the Bulk list in the next Frame. If cleared by HCD, processing of the Bulk list does not occur after the next SOF. HC always checks this bit whenever it determines to process the list. When disabled, HCD can modify the list. If the HCBulkCurrentED register is pointing to an ED to be removed, HCD must advance the pointer by updating HCBulkCurrentED before re-enabling processing of the list.
4	CLE	ControlList Enable	Setting this bit enables the processing of the Control list in the net Frame. If cleared by HCD, processing of the Control list does not occur after the next SOF. HC always checks this bit whenever it determines to process the list. When disabled, HCD can modify the list. If the HcControlCurrentED register is pointing to an ED to be removed, HCD must advance the pointer by updating HcControlCurrentED before re-enabling processing of the list.
3	IE	Isochronous Enable	This bit is used by HCD to enable/disable processing of isochronous EDs. While processing the Periodic Transfer List in a Frame, HC checks the status of this bit when it finds an Isochronous ED(F=1). If set (enabled), HC continues processing the EDs. If cleared (disabled), HC halts processing of the Periodic Transfer List and begins processing the Bulk and Control lists. Setting this bit is guaranteed to take effect in the next Frame. * This product has restrictions for isochronous transfer.
2	PLE	PeriodicList Enable	Setting this bit enables the processing of the Periodic Transfer List in the next Frame. If cleared by HCD, processing of the Periodic Transfer List does not occur after the next SOF. HC must check this bit before it starts processing the list.
1:0	CBSR	ControlBulk ServiceRatio	This bit specifies the service ratio between Control and Bulk EDs. Before processing any of the Asynchronous Transfer Lists, HC must compare the ratio specified with its internal count on how many nonempty Control EDs have been processed, in determining whether to continue serving another Control ED or switching to Bulk EDs. The internal count will be retained even when crossing the frame boundary. In case of reset, HCD is responsible for restoring this value again. CBSR No. of Control EDs over Bulk EDs served 00 1:1 01 2:1 10 3:1 11 4:1

HcCommandStatus

The HcCommandStatus register is used by the Host Controller to receive commands issued by the Host Controller Driver, as well as reflecting the current status of the Host Controller. To the Host Controller Driver, it appears to be a "write to set" register. The Host Controller ensures that bits written as '1' become set in the register while bits written as '0' remain unchanged in the register. The Host Controller Driver may issue multiple distinct commands to the Host Controller without concern for corrupting previously issued commands. The Host Controller Driver has normal read access to all bits.

The SchdulingOverrunCount field indicates the number of frames with which the Host Controller has detected the scheduling overrun error. This occurs when the Periodic Transfer List does not complete before EOF. When a scheduling overrun error is detected, the Host Controller increments the counter and sets the SchedulingOverrun field in the HcInterruptStatus register.

Address = (0x4000_3000) + 0x0008

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved														SOC	
Read/Write (HCD)															R	
Read/Write (HC)															R/W	
After reset															0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved												OCR	BLF	CLF	HCR
Read/Write (HCD)													R/W			
Read/Write (HC)													R/W			
After reset													0	0	0	0

Bit	Mnemonic	Field name	Description
31:18		Reserved	
16:17	SOC	Scheduling OverrunCount	These bits are incremented on each scheduling overrun error. It is initialized to 00b and wraps around at 11b. These bits will be incremented when a scheduling overrun error is detected even if the SchedulingOverrun field in the HcInterruptStatus register has already been set. This is used by HCD to monitor any persistent scheduling problems.
15:4		Reserved	
3	OCR	Ownership ChangeRequest	This bit is set by an OS HCD to request a change of control of the HC. When set, HC will set the OwnershipChange field in *the* HcInterruptStatus register. After the changeover, this bit is cleared and remains so until the next request from OS HCD.
2	BLF	BulkListFilled	This bit is used to indicate whether there are any TDs on the Bulk list. It is set by HCD whenever it adds a TD to an ED in the Bulk list. When HC begins to process the head of the Bulk list, it checks BLF. As long as BulkListFilled is 0, HC will not start processing the Bulk list. If BulkListFilled is 1, HC will start processing the Bulk list and will set BLF to 0. If HC finds a TD on the list, then HC will set BulkListFilled to 1 causing the Bulk list processing to continue. If no TD is found on the Bulk list, and if HCD does not set BulkListFilled, then BulkListFilled will still be 0 when HC completes processing the Bulk list and Bulk list processing will stop.

1	CLF	ControlListFilled	This bit is used to indicate whether there are any TDs on the Control list. It is set by HCD whenever it adds a TD to an ED in the Control list. When HC begins to process the head of the Control list, it checks CLF. As long as ControlListFilled is 0, HC will not start processing the Control list. If CLF is 1, HC will start processing the Control list and will set CLF to 0. If HC finds a TD on the list, then HC will set ControlListFilled to 1 causing the Control list processing to continue. If no TD is found on the Control list, and if the HCD does not set ControlListFilled, thenControlListFilled will still be 0 when HC completes processing the Control list and Control list processing will stop.
0	HCR	HostController Reset	This bit is set by HCD to initiate a software reset of HC. Regardless of the functional state of HC, it moves to the USBSUSPEND state in which most of the operational registers are reset except those stated otherwise; e.g., the InterruptRouting field in the HcControl register, and no Host bus accesses are allowed. This bit is cleared by HC upon the completion of the reset operation. The reset operation must be completed within 10 μs. This bit, when set, will not cause a reset to the Root Hub and no subsequent reset signaling will be asserted to its downstream ports.

HcInterruptStatus

The HcInterruptStatus register provides status on various events that cause hardware interrupts. When an event occurs, the Host Controller sets the corresponding bit in this register. When a bit becomes set, a hardware interrupt is generated if the interrupt is enabled in the HcInterruptEnable register and the MasterInterruptEnable bit is set. The Host Controller Driver can clear specific bits in this register by writing a '1' to bit positions to be cleared. The Host Controller Driver cannot set any of these bits. The Host Controller will never clear the bit.

Address = (0x4000_3000) + 0x000C

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved	OC	Reserved													
Read/Write (HCD)		R/W														
Read/Write (HC)		R/W														
After reset		0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved									RHSC	FNO	UE	RD	SF	WDH	SO
Read/Write (HCD)										R/W						
Read/Write (HC)										R/W						
After reset										0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31		Reserved	
30	OC	Ownership Change	This bit is set by HC when HCD sets the OwnershipChangeRequest field in the HcCommandStatus register. This event, when unmasked, will generate an System Management Interrupt (SMI) immediately. This bit is tied to 0b when the SMI pin is not implemented.
29:7		Reserved	
6	RHSC	RootHubStatus Change	This bit is set when the content of HcRhStatus or the content of any of HcRhPortStatus [NumberofDownstreamPort] has changed.
5	FNO	FrameNumber Overflow	This bit is set when the MSb of *the* HcFmNumber register (bit 15) changes value, from 0 to 1 or from 1 to 0, and after HccaFrameNumber has been updated.
4	UE	Unrecoverable Error	This bit is set when HC detects a system error not related to USB. HC will not proceed with any processing or signaling before the system error has been corrected. HCD clears this bit after HC has been reset.
3	RD	ResumeDetected	This bit is set when HC detects that a device on the USB is asserting resume signaling. It is the transition from no resume signaling to resume signaling that causes this bit to be set. This bit is not set when HCD sets the USBRESUME state.
2	SF	StartofFrame	This bit is set by HC at each start of a frame and after the update of HccaFrameNumber. HC also generates a SOF token at the same time.
1	WDH	WritebackDone Head	This bit is set immediately after HC has written HcDoneHead to HccaDoneHead. Further updates of the HccaDoneHead will not occur until this bit has been cleared. HCD only clears this bit after it has saved the content of HccaDoneHead.
0	SO	Scheduling Overrun	This bit is set when the USB schedule for the current Frame overruns and after the update of HccaFrameNumber. A scheduling overrun will also cause the SchedulingOverrunCount field in the HcCommandStatus register to be incremented.

HcInterruptEnable

Each enable bit in the HcInterruptEnable register corresponds to an associated interrupt bit in the HcInterruptStatus register. The HcInterruptEnable register is used to control which events generate a hardware interrupt. When a bit is set in the HcInterruptStatus register AND the corresponding bit in the HcInterruptEnable register is set AND the MasterInterruptEnable bit is set, then a hardware interrupt is requested on the host bus.

Writing a '1' to a bit in this register sets the corresponding bit, whereas writing a '0' to a bit in this register leaves the corresponding bit unchanged. On read, the current value of this register is returned.

Address = (0x4000_3000) + 0x0010

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	MIE	OC	Reserved													
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0														

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved									RHSC	FNO	UE	RD	SF	WDH	SO
Read/Write (HCD)										R/W						
Read/Write (HC)										R						
After reset										0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31	MIE	MasterInterrupt Enable	A '0' written to this field is ignored by HC. A '1' written to this field enables interrupt generation due to events specified in the other bits of this register**.** This is used by HCD as a Master Interrupt Enable.
30	OC	Ownership Change	0: Ignore 1: Enable interrupt generation due to OwnershipChange.
29:7		Reserved	
6	RHSC	RootHubStatus Change	0: Ignore 1: Enable interrupt generation due to RootHubStatusChange.
5	FNO	FrameNumber Overflow	0: Ignore 1: Enable interrupt generation due to FrameNumberOverflow.
4	UE	Unrecoverable Error	0: Ignore 1: Enable interrupt generation due to UnrecoverableError.
3	RD	ResumeDetected	0: Ignore 1: Enable interrupt generation due to ResumeDetected.
2	SF	StartofFrame	0: Ignore 1: Enable interrupt generation due to StartofFrame.
1	WDH	WritebackDone Head	0: Ignore 1: Enable interrupt generation due to HcDoneHeadWriteback.
0	SO	Scheduling Overrun	0: Ignore 1: Enable interrupt generation due to SchedulingOverrun.

HcInterruptDisable

Each disable bit in the HcInterruptDisable register corresponds to an associated interrupt bit in the HcInterruptStatus register. The HcInterruptDisable register is coupled with the HcInterruptEnable register. Thus, writing a '1' to a bit in this register clears the corresponding bit in the HcInterruptEnable register, whereas writing a '0' to a bit in this register leaves the corresponding bit in the HcInterruptEnable register unchanged. On read, the current value of the HcInterruptEnable register is returned.

Address = (0x4000_3000) + 0x0014

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	MIE	OC	Reserved													
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0														

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved									RHSC	FNO	UE	RD	SF	WDH	SO
Read/Write (HCD)										R/W						
Read/Write (HC)										R						
After reset										0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31	MIE	MasterInterrupt Enable	A '0' written to this field is ignored by HC. A '1' written to this field disables interrupt generation due to events specified in the other bits of this register. This field is set after a hardware or software reset.
30	OC	Ownership Change	0: Ignore 1: Disable interrupt generation due to OwnershipChange.
29:7		Reserved	
6	RHSC	RootHubStatus Change	0: Ignore 1: Disable interrupt generation due to RootHubStatusChange.
5	FNO	FrameNumber Overflow	0: Ignore 1: Disable interrupt generation due to FrameNumberOverflow.
4	UE	Unrecoverable Error	0: Ignore 1: Disable interrupt generation due to UnrecoverableError.
3	RD	ResumeDetected	0: Ignore 1: Disable interrupt generation due to ResumeDetected.
2	SF	StartofFrame	0: Ignore 1: Disable interrupt generation due to StartofFrame.
1	WDH	WritebackDone Head	0: Ignore 1: Disable interrupt generation due to HcDoneHeadWriteback.
0	SO	Scheduling Overrun	0: Ignore 1: Disable interrupt generation due to SchedulingOverrun.

HcHCCA

The HcHCCA register sets the physical address of the Host Controller Communication Area. The Host Controller Driver determines the alignment restrictions by writing all ones to HcHCCA and reading the content of HcHCCA. The alignment is evaluated by examining the number of zeros in the lower order bits. The minimum alignment is 256 bytes; therefore, bits 0 through 7 must always return '0' when read. This area is used to hold the control structures and the Interrupt table that are accessed by both the Host Controller and the Host Controller Driver.

Address = (0x4000_3000) + 0x0018

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	HCCA															
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	HCCA								Reserved							
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0								

Bit	Mnemonic	Field name	Description
31:8	HCCA	HostController Communication Area	This is the page address of the Host Controller Communication Area.
7:0		Reserved	

HcPeriodCurrentED

The HcPeriodCurrentED register sets the physical address of the current Isochronous or Interrupt Endpoint Descriptor.

Address = (0x4000_3000) + 0x001C

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	PCED															
Read/Write (HCD)	R															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	PCED												Reserved			
Read/Write (HCD)	R															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Description
31:4	PCED	PeriodCurrent ED	This is used by HC to point to the head of one of the Periodic Transfer Lists which will be processed in the current Frame. The contents of this register are updated by HC after a periodic transfer ED has been processed. HCD may read the content in determining which ED is currently being processed at the time of reading.
3:0		Reserved	

HcControlHeadED

The HcControlHeadED register sets the physical address of the first Endpoint Descriptor of the Control list.

Address = (0x4000_3000) + 0x0020

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	CHED															
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	CHED												Reserved			
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Description
31:4	CHED	ControlHeadED	HC traverses the Control list starting with the HcControlHeadEDpointer. The contents are loaded from HCCA during the initialization of HC.
3:0		Reserved	

HcControlCurrentED

The HcControlCurrentED register sets the physical address of the current Endpoint Descriptor of the Control list.

Address = (0x4000_3000) + 0x0024

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	CCED															
Read/Write (HCD)	R/W															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	CCED												Reserved			
Read/Write (HCD)	R/W															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Description
31:4	CCED	ControlCurrentED	This pointer is advanced to the next ED after serving the present one. HC will continue processing the list from where it left off in the last Frame. When it reaches the end of the Control list, HC checks the ControlListFilled field in the HcCommandStatus register. If set, it copies the contents of the HcControlHeadED register to the HcControlCurrentED register and clears the bit. If not set, HC executes nothing. HCD is allowed to modify this register only when the ControlListEnable field in the HcControl register is cleared. When set, HCD only reads the instantaneous value of this register. Initially, this field is set to zero to indicate the end of the Control list.
3:0		Reserved	

HcBulkHeadED

The HcBulkHeadED register sets the physical address of the first Endpoint Descriptor of the Bulk list.

Address = (0x4000_3000) + 0x0028

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	BHED															
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	BHED												Reserved			
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Description
31:4	BHED	BulkHeadED	HC traverses the Bulk list starting with the HCBulkHeadED pointer. The contents are loaded from HCCA during the initialization of HC.
3:0		Reserved	

HcBulkCurrentED

The HcBulkCurrentED register sets the physical address of the current endpoint of the Bulk list. As the Bulk list will be processed sequentially, the endpoints will be ordered according to their insertion to the list.

Address = (0x4000_3000) + 0x002C

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	BCED															
Read/Write (HCD)	R/W															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	BCED												Reserved			
Read/Write (HCD)	R/W															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Description
31:4	BCED	BulkCurrentED	This pointer is advanced to the next ED after the HC has served the present one. HC will continue processing the list from where it left off in the last Frame. When it reaches the end of the Bulk list, HC checks the BulklListFilled field in the HcCommandStatus register. If set, it copies the contents of the HcBulkHeadED register to the HcBulkCurrentED register and clears the bit. If not set, HC executes nothing. HCD is allowed to modify this register only when the BulkListEnable field in the HcControl register is cleared. When set, HCD only reads the instantaneous value of this register. Initially, this field is set to zero to indicate the end of the Bulk list.
3:0		Reserved	

HcDoneHead

The HcDoneHead register sets the physical address of the last completed Transfer Descriptor that was added to the Done queue. In normal operation, Host Controller Driver should not need to read this register as its contents are periodically written to the HCCA.

Address = (0x4000_3000) + 0x0030

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	DH															
Read/Write (HCD)	R															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	DH												Reserved			
Read/Write (HCD)	R															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Description
31:4	DH	DoneHead	When a TD is completed, HC writes the contents of HcDoneHead to the NextTD field of the TD. HC then overwrites the contents of HcDoneHead with the address of this TD. This field is set to zero whenever HC writes the contents of this register to HCCA. It also sets the WritebackDoneHead field in the HcInterruptStatus register.
3:0		Reserved	

HcFmInterval

The HcFmInterval register sets a 14-bit value which indicates the bit time interval in a Frame, (i.e., between two consecutive SOFs), and a 15-bit value indicating the Full Speed maximum packet size that the Host Controller can transmit or receive without causing scheduling overrun. The Host Controller Driver carries out minor adjustments on the FrameInterval by writing a new value over the present one at each SOF. This provides the programmability necessary for the Host Controller to synchronize with an external clocking resource and to adjust any unknown local clock offset.

Address = (0x4000_3000) + 0x0034

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	FIT	FSMPS														
Read/Write (HCD)	R/W	R/W														
Read/Write (HC)	R	R														
After reset	0	TBD														
	15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
bit Symbol	Reserved		FI													
Read/Write (HCD)			R/W													
Read/Write (HC)			R													
After reset			1	0	1	1	1	0	1	1	0	1	1	1	1	1

Bit	Mnemonic	Field name	Description
31	FIT	FrameInterval Toggle	HCD toggles this bit whenever it loads a new value to FrameInterval.
30:16	FSMPS	FSLargestData Packet	This field specifies a value which is loaded into the Largest Data Packet Counter at the beginning of each frame. The counter value represents the largest amount of data in bits which can be sent or received by the HC in a single transaction at any given time without causing scheduling overrun. The field value is calculated by the HCD.
15:14		Reserved	
13:0	FI	FrameInterval	This field specifies the interval between two consecutive SOFs in bit times. The nominal value is set to be 11,999. HCD must store the current value of this field before resetting HC. Setting the HostControlReset field in the HcCommandStatus register will cause the HC to reset this field to its nominal value. HCD can choose to restore the stored value upon the completion of the Reset sequence.

HcFmRemaining

The HcFmRemaining register is a 14-bit down counter showing the bit time remaining in the current Frame.

Address = (0x4000_3000) + 0x0038

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	FRT	Reserved														
Read/Write (HCD)	R															
Read/Write (HC)	R/W															
After reset	0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved		FR													
Read/Write (HCD)			R													
Read/Write (HC)			R/W													
After reset			0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31	FRT	FrameRemaining Toggle	This bit is loaded from the FrameIntervalToggle field in the HcFmInterbval register whenever FrameRemaining reaches 0. This bit is used by HCD for the synchronization between the FmInterval field and the FmRemaining field.
30:14		Reserved	
13:0	FR	FrameRemaining	This counter is decremented at each bit time. When it reaches zero, it is reset by loading the FrameInterval field value specified in the HcFmInterval register at the next bit time boundary. When entering the USBOPERATIONAL state, HC re-loads the contents with the FrameInterval field in the HcFmInterval register and uses the updated value from the next SOF.

HcFmNumber

The HcFmNumber register is a 16-bit counter. It provides a timing reference among events happening in the Host Controller and the Host Controller Driver. The Host Controller Driver uses the 16-bit value specified in this register and generates a 32-bit frame number without requiring frequent access to the register.

Address = (0x4000_3000) + 0x003C

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved															
Read/Write (HCD)																
Read/Write (HC)																
After reset																
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	FN															
Read/Write (HCD)	R															
Read/Write (HC)	R/W															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31:16		Reserved	
15:0	FN	FrameNumber	This field is incremented when the HcFmRemaining register is re-loaded. It will be rolled over to 0x0000 after 0xFFFF. When entering the USBOPERATIONAL state, this field will be incremented automatically. The contents will be written to HCCA after HC has incremented the FrameNumber at each frame boundary and sent an SOF but before HC reads the first ED in that Frame. After writing to HCCA, HC will set the StartofFrame field in the HcInterruptStatus register.

HcPeriodicStart

The HcPeriodicStart register sets a 14-bit programmable value which determines when is the earliest time that the HC should start processing the Periodic Transfer List.

Address = (0x4000_3000) + 0x0040

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved															
Read/Write (HCD)																
Read/Write (HC)																
After reset																

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved		PS													
Read/Write (HCD)			R/W													
Read/Write (HC)			R													
After reset			0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31:14		Reserved	
13:0	PS	PeriodicStart	After a hardware reset, this field is cleared. This is then set by HCD during the HC initialization. The value is calculated roughly as 10% off from the HcFmInterval register value. A typical value is 0x3E67. When the HcFmRemaining register reaches the value specified, processing of the Periodic Transfer Lists will have priority over Control/Bulk list processing. HC will therefore start processing the Interrupt List after completing the current Control or Bulk transaction that is in progress.

HcLSThreshold

The HcLSThreshold register contains an 11-bit value used by the Host Controller to determine whether to commit to the transfer of a maximum of 8-byte LS packet before EOF. Neither the Host Controller nor the Host Controller Driver are allowed to change this value.

Address = (0x4000_3000) + 0x0044

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved															
Read/Write (HCD)																
Read/Write (HC)																
After reset																
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved				LST											
Read/Write (HCD)					R/W											
Read/Write (HC)					R											
After reset					0	1	1	0	0	0	1	0	1	0	0	0

Bit	Mnemonic	Field name	Description
31:12		Reserved	
11:0	LST	LSThreshold	This field sets a value which is compared to the FrameRemaining field prior to initiating a Low Speed transaction. The transaction is started only if FrameRemaining this field. The value is calculated by HCD with the consideration of transmission and setup overhead.

HcRhDescriptorA

The HcRhDescriptorA register is the first register of two describing the characteristics of the Root Hub. Reset values are implementation-specific. The descriptor length, descriptor type, and hub descriptor current fields of the hub Class Descriptor are emulated by the HCD. All other fields are located in the HcRhDescriptorA and HcRhDescriptorB registers.

Address = (0x4000_3000) + 0x0048

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	POTPGT								Reserved							
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	1	0								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved			NOCP	OCPM	DT	NPS	PSM	NDP							
Read/Write (HCD)				R/W	R/W	R	R/W	R/W	R							
Read/Write (HC)				R	R	R	R	R	R							
After reset				0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	Mnemonic	Field name	Description
31:24	POTPGT	PowerOnTo PowerGoodTime	This byte specifies the duration that the HCD has to wait before accessing a powered-on port of the Root Hub. It is implementation-specific. The unit of time is 2 ms. The duration is calculated as POTPGT x 2 ms.
23:13		Reserved	
12	NOCP	NoOverCurrent Protection	This bit describes how the overcurrent status for the Root Hub ports are reported. When this bit is cleared, the OverCurrentProtectionMode field specifies global or per-port reporting. 0: Overcurrent status is reported collectively for all downstream ports. 1: No overcurrent protection supported.
11	OCPM	OverCurrent ProtectionMode	This bit describes how the overcurrent status for the Root Hub ports are reported. At reset, this field should reflect the same mode as PowerSwitchingMode. This field is valid only if the NoOverCurrent Protection field is cleared. 0: Overcurrent status is reported collectively for all downstream ports. 1: overcurrent status is reported on a per-port basis.
10	DT	DeviceType	This bit specifies that the Root Hub is not a compound device. The Root Hub is not permitted to be a compound device. This field should always read/write 0.
9	NPS	NoPower Switching	These bits are used to specify whether power switching is supported or ports are always powered on. It is implementation-specific. When this bit is cleared, the PowerSwitchingMode field specifies global or per-port switching. 0: Ports are power switched 1: Ports are always powered on when the HC is powered on
8	PSM	PowerSwitching Mode	This bit is used to specify how the power switching of the Root Hub ports is controlled. It is implementation-specific. This field is only valid if the NoPower Switching field is cleared. 0: All ports are powered on at the same time. 1: Each port is powered on individually. This mode allows port power to be controlled by either the global switch or per-port switching. If the PortPowerCurrentMask bit in the HcRhDescriptorB register is set, the port responds only to port power commands If the port mask is cleared, then the port is controlled only by the global power switch (Set/ClearGlobalPower).
17:0	NDP	Number DownstreamPorts	These bits specify the number of downstream ports supported only by the Root Hub. It is implementation-specific. The number of ports of this module is 1 and therefore 0x01 is read.

HcRhDescriptorB

The HcRhDescriptorB register is the second register of two describing the characteristics of the Root Hub.

These fields are written during initialization to correspond with the system implementation. Reset values are implementation-specific.

Address = (0x4000_3000) + 0x004C

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	PPCM															
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	DR															
Read/Write (HCD)	R/W															
Read/Write (HC)	R															
After reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Description
31:16	PPCM	PortPower ControlMask	Each bit indicates if a port is controlled by a global power control command when PowerSwitchingMode in the HcRhDescriptorA register is set. When set, the port's power state is only controlled by per-port power control (Set/ClearPortPower). When cleared, the port is controlled by the global power switch (Set/ClearGlobalPower). If the device is configured to global switching mode (PowerSwitchingMode=0), this field is not valid. bit0:Reserved bit1:Ganged-power mask on Port#1
15:0	DR	Device Removable	Each bit is dedicated to a port of the Root Hub. When set to 0, the attached device is removable with power on. When set to 1, the attached device is not removable with power on. bit0:Reserved bit1:Device attached to Port#1

HcRhStatus

The HcRhStatus register is divided into two parts. The lower word of a Dword represents the Hub Status field and the upper word represents the Hub Status Change field.

Reserved bits should always be written '0'.

Address = (0x4000_3000) + 0x0050

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	CRWE	Reserved													OCIC	LPSC
Read/Write (HCD)	W														R/W	R/W
Read/Write (HC)	R														R/W	R
After reset	–														0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	DRWE	Reserved													OCI	LPS
Read/Write (HCD)	R/W														R	R/W
Read/Write (HC)	R														R/W	R
After reset	0														0	0

Bit	Mnemonic	Field name	Description
31	CRWE	ClearRemote WakeupEnable	A write of a '1' clears DeviceRemoteWakeUpEbable. A write of a '0' has no effect.
30:18		Reserved	
17	OCIC	OverCurrent Indicator Change	This bit is set by hardware when a change has occurred to the OCI field of this register. The HCD clears this bit by writing a '1'. A write of a '0' has no effect.
16	LPSC	LocalPower StatusChange	(read)LocalPowerStatusChange The route hub does not support LocalPowerStatusChange. Thus, this bit is always read as 0. (write)SetGlobalPower In global power mode (PowerSwitchingMode=0), this bit is written to '1' to turn on power to all ports (clearPortPowerStatus). In per-port power mode, it sets PortPowerStatus only on ports whose PortPowerControlMaskbit is not set. A write of a '0' has no effect.
15	DRWE	DeviceRemote WakeupEnable	(read)DeviceRemoteWakeupEnable This bit enables a ConnectStatusChange bit as a resume event, causing a USBSUSPEND to USBRESUME state transition and setting the ResumeDetected interrupt. 0: ConnectStatusChange is not a remote wakeup event. 1: ConnectStatusChange is a remote wakeup event. (write) A write of a '1' sets DeviceRemoteWakeupEnable. A write of a '0' has no effect.
14:2		Reserved	
1	OCI	OverCurrent Indicator	This bit reports overcurrent conditions when the global reporting is implemented. When set, an overcurrent condition exists. When cleared, all power operations are normal. If per-port overcurrent protection is implemented, this bit is always '0'.

0	LPS	LocalPower Status	(read)LocalPowerStatus The route hub does not support LocalPowerStatus. Thus, this bit is always read as 0. (write)ClearGlobalPower In global power mode (PowerSwitchMode=0), this bit is written to '1' to turn off power to all ports (clear PortPowerStatus). In per-port power mode, it clears PortPowerStatus only on ports whose PortPowerControlMask bit is not set. A write of a '0' has no effect.

HcRhPortStatus

The HcRhPortStatus register is used to control and report port events on a per-port basis. NumberDownstreamPorts in the HcRhDescriptorA register represents the number of HcRhPortStatus registers that are implemented in hardware. The lower word is used to reflect the port status, whereas the upper word reflects the status change bits. Some status bits are implemented with special write behavior (see below). If a transaction (token through handshake) is in progress when a write to change port status occurs, the resulting port status change will be executed after the transaction completes. Reserved bits should always be written '0'.

Address = (0x4000_3000) + 0x0054

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved											PRSC	OCIC	PSSC	PESC	CSC
Read/Write (HCD)												R/W	R/W	R/W	R/W	R/W
Read/Write (HC)												R/W	R/W	R/W	R/W	R/W
After reset												0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved						LSDA	PPS	Reserved			PRS	POCI	PSS	PES	CCS
Read/Write (HCD)							R/W	R/W				R/W	R/W	R/W	R/W	R/W
Read/Write (HC)							R/W	R/W				R/W	R/W	R/W	R/W	R/W
After reset							X	0				0	0	0	0	0

Bit	Mnemonic	Field name	Description
31:21		Reserved	
20	PRSC	PortResetStatus Change	This bit is set at the end of the 10-ms port reset signal. The HCD writes 1 to clear this bit. A write of a '0' has no effect. 0: Port reset is not compete 1: Port reset is compete
19	OCIC	PortOverCurrent IndicatorChange	This bit is valid when overcurrent conditions are reported on a per-port basis. This bit is set when Root Hub changes the PortOverCurrentIndicator bit. The HCD writes '1' to clear this bit. A write of a '0' has no effect. 0: No change in PortOverCurrentIndicator 1: PortOverCurrentIndicator has changed
18	PSSC	PortSuspend StatusChange	This bit is set when the full resume sequence has been completed. This sequence includes the 20-s resume pulse, LS EOP, and 3-ms resynchronization delay. The HCD writes '1' to clear this bit. A write of a '0' has no effect. This bit is also cleared when ResetStatusChange is set. 0: Resume is not competed 1: Resume competed
17	PESC	PortEnable StatusChange	This bit is set when hardware events cause the PortEnableStatus bit to be cleared. Changes from HCD writes do not set this bit. The HCD writes '1' to clear this bit. A write of a '0' has no effect. 0: No change in PortEnableStatus 1: PortEnableStatus has changed

16	CSC	ConnectStatus Change	This bit is set whenever a connect or disconnect event occurs. The HCD writes '1' to clear this bit. A write of a '0' has no effect. If CurrentConnectStatus is cleared when a SetPortReset, SetPortEnable, or SetPortSuspend write occurs, this bit is set to force the driver to re-evaluate the connection status since these writes will not occur if the port is disconnected. 0: No change in CurrentConnectStatus 1: CurrentControlStatus has changed (Note) If the DeviceRemovable[NDP] bit is set, this bit is set only after a Root Hub reset to inform the system that the device is attached.
15:10		Reserved	
9	LSDA	LowSpeed DeviceAttached	(read)LowSpeedDeviceAttached This bit indicates the speed of the device attached to this port. When set, a Low Speed device is attached to this port. When cleared, a Full Speed device is attached to this port. This field is valid only when CurrentConnectStatus is set. 0: Full Speed device attached 1: Low Speed device attached (write)ClearPortPower The HCD clears the PortPowerStatus bit by writing a '1' to this bit. A write of a '0' has no effect.
8	PPS	PortPower Status	(read)PortPowerStatus This bit reflects the port's power status, regardless of the type of power switching implemented. This bit is cleared if an overcurrent condition is detected. HCD sets this bit by writing SetPortPower or SetGlobalPower. HCD clears this bit by writing ClearPortPower or ClearGlobalPower. Which power control switches are enabled is determined by PortSwitchingMode and PortPowerControlMask[NDP]. In global switching mode (PowerSwitchingMode=0), only Set/ClearGlobalPower controls this bit. In per-port power switching (PowerSwitchingMode=1), if the PortPowerControlMask[NDP] bit for the port is set, only Set/ClearPortPower commands are enabled. If the mask is not set, only Set/ClearGlobalPower commands are enabled. When port power is disabled, CurrentConnectStatus, PortEnableStatus, PortSuspendStatus, and PortResetStatus must be reset. 0: Port power is off 1: Port power is on (write)SetPortPower The HCD writes a '1' to set the PortPowerStatus bit. A write of a '0' has no effect. (Note) This bit always reads '1' if power switching is not supported.
7:5		Reserved	
4	PRS	PortReset Status	(read)PortResetStatus When this bit is reset by a write to SetPortReset, the port reset signal is asserted. When reset is completed, this bit is cleared when PortResetStatusChange is set. This bit cannot be set if CurrentConnectStatus is cleared. 0: Port reset signal is not active 1: Port reset signal is active (write)SetPortReset The HCD sets the port reset signaling by writing a '1' to this bit. A write of a '0' has no effect. If CurrentConnectStatus is cleared, this write does not set PortResetStatus, but instead sets CurrentConnectStatusChange. This informs the driver that it attempted to reset a disconnected port.

3	POCI	PortOverCurrent Indicator	(read)PortOverCurrentIndicator This bit is only valid when the Root Hub is configured in such a way that overcurrent conditions are reported on a per-port basis. If per-port overcurrent reporting is not supported, this bit is cleared to 0. If cleared, all power operations are normal for this port. If set, an overcurrent condition exists on this port. This bit always reflects the overcurrent input signal. 0: No overcurrent condition 1: Overcurrent condition detected (write)ClearSuspendStatus The HCD writes a '1' to initiate a resume. A write of a '0' has no effect. A resume is initiated only if PortSuspendStatus is set.
2	PSS	PortSuspend Status	(read)PortSuspendStatus This bit indicates that the port is suspended or in the resume sequence. It is set by a SetSuspendState write and cleared when PortSuspendStatusChange is set at the end of the resume interval. This bit is also cleared when CurrentConnectStatus is set. This bit is also cleared when PortResetStatusChange is set at the end of the port reset or when the HC is placed in the USBRESUME state. 0: Port is not suspended 1: Port is suspended (write)SetPortSuspend The HCD sets the PortSuspendStatus bit by writing a '1' to this bit. A write of a '0' has no effect. If CurrentConnectStatus is cleared, this write does not set PortSuspendStatus; instead it sets ConnectStatusChange. This informs the driver that it attempted to suspend a disconnected port.
1	PES	PortEnable Status	(read)PortEnableStatus This bit indicates whether the port is enabled or disabled. The Root Hub clears this bit when an overcurrent condition, disconnect event, switched-off power, or operational error such as babble is detected. This change also causes PortEnabledStatusChange to be set. HCD sets this bit by writing SetPortEnable and clears it by writing ClearPortEnable. This bit cannot be set when CurrentConnectStatus is cleared. This bit is also set at the completion of a port reset when ResetStatusChange is set or port suspend when SuspendStatusChange is set. 0: Port is disabled 1: Port is enabled (write)SetPortEnable The HCD sets PortEnableStatus by writing a '1'. A write of a '0' has no effect. If CurrentConnectStatus is cleared, this write does not set PortEnableStatus, but instead sets ConnectStatusChange. This informs the driver that it attempted to enable a disconnected port.
0	CCS	CurrentConnect Status	(read)CurrentConnectStatus This bit reflects the current state of the downstream port. 0: No device connected 1: Device connected (write)ClearPortEnable The HCD writes a '1' to this bit to clear the PortEnableStatus bit. A write of a '0' has no effect. CurrentConectStatus is not affected by any write. (Note) This bit always reads '1' when the attached device is nonremovable (Device Removable[NDP]).

3.16.9.2 EHCI registers

The attribute symbols are explained below:

RO: Read Only. You can make reads only. If you make a write, it has no effect.

WO: Write Only. You can make writes only. If you make a read, 0 is read.

R/W: Read/Write. A register with this attribute can be both read and written.

R/WC: Read/Write Clear. A register with this attribute can be both read and written. However, a write of a '1' clears (sets to 0) the corresponding bit. A write of a '0' has no effect.

A. Register map

Capability registers

base address = 0x4000_2000

Register name	Address (base+)	Attribute	Initial value
CAPLENGTH	0x0000	RO	0x10
Reserved	0x0001	RO	0x00
HCIVERSION	0x0002	RO	0x0100
HCSPARAMS	0x0004	RO	0x0000_1111
HCCPARAMS	0x0008	RO	0x0000_0016

EHCI Operational registers

base address = 0x4000_2000

Register name	Address (base+)	Attribute	Initial value
USBCMD	0x0010	RO,R/W,WO	0x00080B00 (Park mode support)
USBSTS	0x0014	RO,R/W,R/WC	0x00001000
USBINTR	0x0018	R/W	0x00000000
FRINDEX	0x001C	R/W	0x00000000
CTRLDSSEGMENT	0x0020	R/W	0x00000000
PERIODICLISTBASE	0x0024	R/W	0x00000000
ASYNCLISTADDR	0x0028	R/W	0x00000000
CONFIGFLAG	0x0050	R/W	0x00000000
PORTSC	0x0054	RO,R/W,R/WC	0x00002000

Host Controller registers

base address = 0x4000_2000

Register name	Address (base+)	Attribute	Initial value
USBHCREG01	0x0094	R/W.RO	0x00400040
USBHCREG03	0x009C	R/W,RO	0x00000000

B. Capability registers

These resisters specify the USBHC implementation, and the values are fixed.

The main specifications of this product are as follows:

Register name	Description
CAPLENGTH	Offset address of Operational register address
HCIVERSION	EHCI revision number
HCSPARAMS	Supports one USBHC of USB1.1. The number of ports is 1. Has the port power switch.
HCCPARAMS	The number of isochronous data micro-frames that the Host Controller can hold is 1. Supports the Park mode. Can change the frame list size.

C. Operational registers

USBCMD

This resister shows commands executed by the USBHC. A write in this register executes the corresponding command.

Address = (0x4000_2000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	–	–	Undefined	Read undefined. Write as zero.
[23:16]	ITC	R/W	0x08	Interrupt Threshold Control. Selects the maximum interval at which the USBHC will issue interrupts. The only valid setting values are shown below. If other values are selected, the results are undefined. Set value Maximum interrupt interval 0x00 (Reserved) 0x01 1 micro-frame 0x02 2 micro-frames 0x04 4 micro-frames 0x08 8 micro-frames (default: 1 ms) 0x10 16 micro-frames (2 ms) 0x20 32 micro-frames (4 ms) 0x40 64 micro-frames (8 ms)
[15:12]	–	–	Undefined	Read as zero. Write as zero.
[11]	ASPME	ROorR/W	0y1	Asynchronous Schedule Park Mode Enable. If the Asynchronous Park Capability bit in the HCCPARAMS register is set to 1, then this field defaults to '1' and is R/W. Otherwise it defaults to zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is set to 1, Park mode is enabled. When this bit is set to 0, Park mode is disabled.
[10]	–	–	Undefined	Read undefined. Write as zero.
[9:8]	ASPMC	ROorR/W	0x3	Asynchronous Schedule Park Mode Count. If the Asynchronous Park Capability bit in the HCCPRAMS register is set to 1, then this field defaults to 0x3 and is R/W. Otherwise it defaults to 0x0 and is RO. It contains a count of the number of successive transactions the USBHC is allowed to execute from a HS queue head on the Asynchronous Schedule. Valid values are 0x1 and 0x3. Software must not write a '0' to this bit when Park mode is enabled. Doing so will produce undefined results.
[7]	LHCR	R/W	0y0	Light Host Controller Reset. This bit allows the driver to reset the EHCI controller without affecting the state of the ports or the relationship to the OHCI controllers. For example, the PORTSC registers should not be reset to their default values and the CF bit setting should not go to zero (retaining port ownership relationships). A host software read of this bit as a '0' indicates the LHCR has completed and it is safe for host software to re-set the USBHC. A host software read of this bit as a '1' indicates that the LHCR has not yet completed.

[6]	IAAD	R/W	0y0	This bit is used as a doorbell to tell the USBHC to issue an interrupt the next time it advances Asynchronous Schedule. Software must write a '1' to this bit to use the doorbell. When the USBHC has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Async Advance Enable bit in the USBINTR register is set to 1, then the USBHC will assert an interrupt at the next interrupt source. The USBHC sets this bit to 0 after it has set the Interrupt on Async Advance status bit in the USBSTS register to 1. Software should not write a '1' to this bit when the Asynchronous Schedule is disabled. Doing so will produce undefined results.
[5]	ASE	R/W	0y0	Asynchronous Schedule Enable. This bit controls whether the USBHC skips processing the Asynchronous Schedule. 0: Does not execute the Asynchronous Schedule 1: Use the ASYNCLISTADDR register to access the Asynchronous Schedule
[4]	PSE	R/W	0y0	Periodic Schedule Enable. This bit controls whether the USBHC skips processing the Periodic Schedule. 0: Does not execute the Periodic Schedule 1: Use the PERIODICLISTBASE register to access the Periodic Schedule
[3:2]	FLS	R/WorRO	0y00	Frame List Size. This field is R/W only if the Programmable Frame List Flag in the HCCPRAMS register is set to 1. This field specifies the size of the frame list. This frame list size controls which bits in the Frame Index Register should be used for the Frame List Current Index register. 0y00 1024 elements(4096bytes) 0y01 512 elements(2048bytes) 0y10 256 elements(1024bytes) 0y11 Reserved
[1]	HCR	R/W	0y0	Host Controller Reset. This control bit is used by software to reset the USBHC. The effects of this on Root Hub registers are similar to a hardware reset. When software writes a '1' to this bit, the USBHC resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. This USB reset is not driven on ports. This bit is set to 0 by the USBHC when the reset process is complete. Software cannot terminate the reset process early by writing a '0' to this register. Software should not set this bit to 1 when HcHalted in the USBSTS register is set to 0. Doing so will produce undefined results.
[0]	RS	RW	0y0	Run/Stop. 1 Run, 0 Stop. When this bit is set to 1, the USBHC starts executing the schedule. The USBHC continues execution as long as this bit is set to 1. When this bit is set to 0, the USBHC completes the current and any actively pipelined transactions on the USB and then halts. The USBHC halts within 16 micro-frames after software clears the Run bit. The HcHalted bit in the status register indicates it when the USBHC has finished its transactions and has entered the stopped state. Software must not write a '1' to this bit if the USBHC is in the Halted state. Doing so will produce undefined results.

USBSTS

This register indicates pending interrupts and various states of the USBHC.

The status resulting from a transaction on the serial bus is not indicated in this register. Software clears a bit to 0 in this register by writing a '1' to it.

Address = (0x4000_2000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as zero. Write as zero.
[15]	ASS	RO	0y0	Asynchronous Schedule Status. The bit reports the current status of the Asynchronous Schedule. If this bit is set to 0, then the status of the Asynchronous Schedule is disabled. If set to 1, then the status of the Asynchronous Schedule is enabled. The USBHC is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either 1 = Enabled or 0 = Disabled.
[14]	PSS	RO	0y0	Periodic Schedule Status. The bit reports the current status of the Periodic Schedule. If this bit is set to 0, then the status of the Periodic Schedule is disabled. If set to1, then the status of the Periodic Schedule is enabled. The USBHC is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either 1 = Enabled or 0 = Disabled.
[13]	R	RO	0y0	Reclamation. This bit is used to detect an empty Asynchronous Schedule.
[12]	HCH	RO	0y1	HCHalted. This bit is a '0' whenever the Run/Stop bit is set to 1. The USBHC sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0.
[11:6]	–	–	Undefined	Read as zero. Write as zero.
[5]	IAA	R/WC	0y0	Interrupt on Async Advance. System software can force the USBHC to issue an interrupt the next time the USBHC advances the Asynchronous Schedule by writing a '1' to the Interrupt on Async Advance Doorbell bit in the USBCMD register. This status bit indicates the existence of that interrupt source.
[4]	HSE	R/WC	0y0	Host System Error. The USBHC sets this bit to 1 when a serious error occurs during a USBHC access involving the USBHC. When this error occurs, the USBHC clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs.
[3]	FLR	R/WC	0y0	Frame List Rollover. The USBHC sets this bit to 1 when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field in the USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the USBHC sets this bit every time FRINDEX[12] toggles.

[2]	PCD	R/WC	0y0	Port Change Detect. The USBHC sets this bit to 1 when the Port Owner bit is set to 0 or is transitioned from 0 to 1, or when a Force Port Resume bit transitions from 0 to 1 as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to 1 after system software has relinquished ownership of a connected port by writing a '1' to a port's Port Owner bit. This bit is loaded with the change bits of the PORTSC register.
[1]	USB ERRINT	R/WC	0y0	USB Error Interrupt. The USBHC sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.
[0]	USBINT	R/WC	0y0	USB Interrupt The USBHC sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set. The USBHC also sets this bit to 1 when a short packet is detected (actual number of bytes received was less than the expected number of bytes).

USBINTR

This register reports Enabled or Disabled for the interrupt function. When a bit is set and the corresponding interrupt is active, an interrupt is generated. Interrupt sources that are disabled in this register still appear in the USBSTS register to allow the software to poll for events. Each interrupt enable bit indicates whether it is dependent on the interrupt occurrence mechanism.

Address = (0x4000_2000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as zero. Write as zero.
[5]	IAAE	R/W	0y0	Interrupt on Async Advance Enable. When this bit is a '1' and the Interrupt on Async Advance bit in the USBSTS register is a '1', the USBHC will issue an interrupt at the next interrupt occurrence timing. This interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.
[4]	HSEE	R/W	0y0	Host System Error Enable. When this bit is a '1' and the Host System Error Status bit in the USBSTS register is a '1', the USBHC will issue an interrupt. This interrupt is acknowledged by software clearing the Host System Error bit.
[3]	FLRE	R/W	0y0	Frame List Rollover Enable. When this bit is a '1' and the Frame List Rollover bit in the USBSTS register is '1', the USBHC will issue an interrupt. This interrupt is acknowledged by software clearing the Frame List Rollover bit.
[2]	PCIE	R/W	0y0	Port Change Interrupt Enable. When this bit is a '1' and the Port Change Detect bit in the USBSTS register is '1', the USBHC will issue an interrupt. This interrupt is acknowledged by software clearing the Port Change Detect bit.
[1]	USB EIE	R/W	0y0	USB Error Interrupt Enable. When this bit is a '1' and the USBERRINT bit in the USBSTS register is '1', the USBHC will issue an interrupt. This interrupt is acknowledged by software clearing the USBERRINT bit.
[0]	USBIE	R/W	0y0	USB Interrupt Enable. When this bit is a '1' and the USBINT bit in the USBSTS register is '1', the USBHC will issue an interrupt. This interrupt is acknowledged by software clearing the USBINT bit.

FRINDEX

This register is used by the USBHC to index into the periodic frame list. The register updates at every 125 µs (once each micro-frame). Bits [N:3] are used to select a particular entry in the Periodic Frame List being executed. The number of bits used for the index depends on the size of the frame list as set by system software in the FrameListSize field in the USBCMD register. This register must be written as a DWord. Byte writing produces undefined results. This register cannot be written if the USBHC is in the Halted state as indicated by the HCHalted bit. A write to this register while the Run/Stop bit is set to 1 produces undefined results. Writing to this register also affects the SOF value.

Address = (0x4000_2000) + 0x001C

Bit	Bit Symbol	Type	Reset Value	Description
[31:14]	–	–	Undefined	Read as zero. Write as zero.
[13:0]	FI	R/W	0x0000	FrameIndex. The value in this register increments at the end of each micro-frame. Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times before moving to the next index. The following illustrates values of *N* based on the value of the FrameListSize field in the USBCMD register. USBCMD[FrameListSize] Number Elements N 0y00 (1024) 12 0y01 (512) 11 0y10 (256) 10 0y11 Reserved The SOF frame number value for the SOF token is derived from this register. The value of FRINDEX must be 125 µs (1 micro-frame) ahead of the SOF token value. The SOF value may be implemented as an 11-bit shadow register. For this discussion, this shadow register is named SOFV. SOFV updates every 8 micro-frames (1 ms) An example implementation to achieve this behavior is to increment SOFV each time the FRINDEX[2:0] increments from 0 to 1. Software must use the value of FRINDEX derived from the current micro-frame number, both for HS isochronous scheduling purposes and to provide the get micro-frame number function required for client drivers. Therefore, the value of FRINDEX and the value of SOFV must be kept consistent if a chip is reset or software writes to FRINDEX. Writing to FRINDEX must also write-through FRINDEX[13:3] to SOFV[10:0]. In order to keep the update as simple as possible, software should never write a FRINDEX value where the three least significant bits are "111" or "000".

CTRLDSSEGMENT

Address = (0x4000_2000) + 0x0020

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	R/W	0x00000000	This 32-bit register must correspond to the address bits [63:32] for all EHCI data structures. If the Addressing Capability field in the HCCPARAMS register is 0, then this register is not used. Software cannot write to it and a read from this register will return zeros. If the 64-bit Addressing Capability field in the HCCPARAMS register is 1, then this register is used with the link pointers to construct 64-bit addresses to EHCI control data structures. This register is concatenated with the link pointer from either PERIODICLISTBASE and ASYNCLISTADDR, or any control data structure link field to construct a 64-bit address. This register allows software to locate all control data structures within the same 4 GB memory segment.

PERIODICLISTBASE

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. If the USBHC is in 64-bit mode, then the most significant 32 bits of every control data structure address come from the CTRLDSSEGMENT register. System software loads this register prior to starting the schedule execution by the USBHC. The memory structure referenced by this physical memory pointer is 4-Kbyte aligned. The contents of this register are combined with FRINDEX to enable the USBHC to step through the Periodic Frame List in sequence.

Address = (0x4000_2000) + 0x0024

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	BA	R/W	Undefined	Base Address(Low). These bits correspond to memory address signals [31:12], respectively.
[11:0]	–	–	Undefined	Read undefined. Write as zero.

ASYNCLISTADDR

This register contains the address of the next asynchronous queue head to be executed. If the USBHC is in 64-bit mode, then the most significant 32 bits of every control data structure address comes from the CTRLDSSEGMENT register. Bits [4:0] of this register cannot be modified by system software and will always return 0 when read. The memory structure referenced by this physical memory pointer is 32-byte (cache line) aligned.

Address = (0x4000_2000) + 0x0028

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	LPL	R/W	Undefined	Link Pointer Low. These bits correspond to memory address signals [31:5], respectively. This field may only reference a QueueHead.
[4:0]	–	–	Undefined	Read undefined. Write as zero.

CONFIGFLAG

This register is initialized only by hardware reset.

Address = (0x4000_2000) + 0x0050

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as zero. Write as zero.
[0]	CF	R/W	0y0	Configure Flag. The default is 0. Software sets this bit as the last action in its process of configuring the USBHC. This bit controls the default port-routing control logic. 0y0: Classic host controller for default routes 0y1: EHCI controller for default routes

PORTSC

The USBHC implements one port register. The initial condition of the port after reset is either No device connected or Port disabled. Software must not change the state of the port until after power supply is started by setting PortPower to 1.

The host is required to have power supply stable to the port within 20 ms.

Note 1) When a device is attached, the port status transitions to the connected state and software can process this as with any status change notification.

Note 2) If a port is being used as the Debug Port, then the port may report device connected, and enabled when the Configured Flag is 0.

Address = (0x4000_2000) + 0x0054

Bit	Bit Symbol	Type	Reset Value	Description
[31:23]	–	–	Undefined	Read as zero. Write as zero.
[22]	WKOC_E	R/W	0y0	Wake on Over-current Enable. The default is 0. Writing this bit to 1 enables the port to be sensitive to over-current conditions as wake-up sources. This bit is '0' if PortPower is set to 0.
[21]	WKDSCNNT_E	R/W	0y0	Wake on Disconnect Enable. The default is 0. Writing this bit to 1 enables the port to be sensitive to device disconnects as wake-up sources. This bit is '0' if PortPower is set to 0.
[20]	WKCNNT_E	R/W	0y0	Wake on Connect Enable. The default is 0. Writing this bit to 1 enables the port to be sensitive to device connects as wake-up sources. This bit is '0' if PortPower is set to 0.
[19:16]	PTC	R/W	0y0000	Port Test Control. When this field is 0, the port is not operating in a test mode. A non-zero value indicates that it is operating in test mode and the specific test mode is indicated by specific values shown below: Bits TestMode 0y0000: Test mode not enabled 0y0001: Test J_STATE 0y0010: Test K_STATE 0y0011: Test SE0_NAK 0y0100: Test Packet 0y0101: Test FORCE_ENABLE

[15:14]	PIC	R/W	0y00	Port Indicator Control. Writing to these bits has no effect if P_INDICATOR in the HCSPARAMS register is a '0'. If P_INDICATOR is 1, then the bit encodings are: Bit value Meaning 0y00: Port Indicators are off 0y01: Amber 0y10: Green 0y11: Undefined This field is '0' if Port Power is set to 0.
[13]	PO	R/W	0y1	Port Owner. This bit unconditionally goes to a 0 when configured in the CONFIGFLAG register makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever configured is 0. Software uses this bit to release ownership of the port selected by the USBHC (this relates to the event that the attached device is not an HS device). Software writes 1 to this bit when the attached device is not a HS device. A 1 in this bit means that a companion USBHC owns and controls the port.
[12]	PP	R/W	0y0	Port Power. The function of this bit depends on the value of the Port Power Control bit in the HCPARAMS register. PPC PP 0y1 0y1/0 U USBHC has port power control switches. This bit represents the current setting of the switch (0 = off, 1 = on). When power is not available on a port (PP=0), the port is nonfunctional and will not report attaches, detaches, etc. When an overcurrent condition is detected on a powered port and PPC is 1, the PP bit in each affected port may be transitioned by the USBHC from 1 to 0 (removing power from the port).
[11:10]	LS	RO	0y00	Line State. These bits reflect the current logical levels of the D+ and D- signal lines. These bits are used for detection of LS USB devices prior to the port reset and operation enable sequence. This field is valid only when the port enable bit is 0 and the current connect status bit is set to 1. Bit State Description 0y00 SE0: Not LS device, perform EHCI reset 0y10 J-state: Not LS device, perform EHCI reset 0y01 K-state: LS device, release ownership of port 0y11 Undefined: Not LS, perform EHCI reset This value of this field is undefined if PP=0.
[9]	–	–	Undefined	Read as zero. Write as zero.
[8]	PR	R/W	0y0	Port Reset. 1 = Port is being reset. 0 = Port is not being reset. The default is 0. When software writes 1 to this bit from 0, the bus reset sequence as defined in theUSB2.0 is started. Software writes 0 to this bit to terminate the bus reset sequence. Software must keep this bit at 1 long enough to ensure that the reset sequence, as specified in the USB2.0, completes. Note: When software writes this bit to 1, it must also write 0 to the PortEnable bit. Note that when software writes 0 to this bit, there may be a delay before the bit status changes to 0. The bit status will not read as 0 until after the reset has completed. If the port is in HS mode after reset is complete, the USBHC will automatically enable this port (PortEnable=1). A USBHC must terminate the reset and stabilize the state of the port within 2 ms of software transitioning this bit from 1 to 0. For example: If the port detects that the attached device is HS during reset, then the USBHC must have the port in the enabled state within 2 ms of software writing this bit to 0. The HCHalted bit in the USBSTS register should be 0 before software attempts to use this bit. This field is 0 if PortPower=0.

[7]	S	R/W	0y0	Suspend. 1 = Port in suspend status. 0 = Port not in suspend status. The default is 0. PortEnabled Suspend Port Status 0 X Disable 1 0 Enable 1 1 Suspend When in suspend status, propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend status, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. A write of a '0' to this bit is ignored by the USBHC. The USBHC will unconditionally set this bit to 0 when: • Software sets the Force Port Resume bit to 0 from 1. • Software sets the Port Reset bit to 1 from 0. If host software sets this bit to 1 when the port is not enabled (Port Enable = 0), the results are undefined. This field is 0 if Port Power = 0.
[6]	FPR	R/W	0y0	Force Port Rsume. 1= Resume detected/driven on port. 0 = No resume (K-state) detected/driven on port. The default is 0. This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is suspended (*Suspend* and *Enabled* bits are 1) and software transitions this bit to 1, then the effects on the bus are undefined. Software sets this bit to 1 to drive resume signaling. The USBHC sets this bit to 1 if a J-to-K transition is detected while the port is in the Suspend status. When this bit Is set because a J-to-K transition is detected, the Port Change Detect bit in the USBSTS register is also set to 1. If software sets this bit to 1, the USBHC will not set the Port Change Detect bit. Note that when the EHCI controller owns the port, the resume sequence follows the USB2.0 Specification. The resume signaling (FS 'K') is driven on the port as long as this bit remains 1. Software must appropriately time the Resume and set this bit to 0 when the appropriate amount of time has elapsed. Writing 0 causes the port to return to HS mode (HS idle). This bit will remain 1 until the port has switched to the HS idle. The USBHC must complete this transition within 2 ms of software setting this bit to 0.
[5]	OC	R/WC	0y0	Over-current Change. The default is 0. 1 = Overcurrent condition detected. Software clears this bit by writing 1 to this bit.
[4]	OA	RO	0y0	Over-current Active. The default is 0. 1 = This port currently has an overcurrent condition. 0 = This port does not have an overcurrent condition. This bit will automatically transition to 0 after the overcurrent condition is removed.

[3]	PE/DC	R/WC	0y0	Port Enable/Disable Change. 1 = Port enable/disable status has changed. 0 = No change. The default is 0. For the root hub, this bit gets set to 1 only when a port is disabled due to the appropriate conditions existing at the EOF2 point. Software clears this bit by writing 1 to it. This field is 0 if Port Power = 0.
[2]	PE/D	R/W	0y0	Port Enabled/Disabled. 1 = Enable. 0 = Disable. The default is 0. Ports can only be set to 1 by the USBHC as a part of the reset and enable. Software cannot enable a port by writing 1 to this field. The USBHC will only set this bit when the reset sequence determines that the attached device is an HS device. Ports can be disabled by either a fault condition (disconnect event or other error) or by software. Note that the bit status does not change until the port status actually changes. There may be a delay in enabling or disabling a port due to other USBHC events and bus events. When the port is disabled, propagation of data is blocked on this port, except for reset. This field is 0 if Port Power = 0.
[1]	CSC	R/WC	0y0	Connect Status Change. 1 = Change in Current Connect Status. 0 = No change. The default is 0. Indicates a change has occurred in the port's Current Connect Status. The USBHC sets this bit for all changes to the port device connect status, even if software has not cleared an existing connect status change. For example, the status may change twice before software has cleared the changed condition. Hub hardware will be "setting" an already-set bit. Software sets this bit to 0 by writing 1 to it. This field is 0 if Port Power = 0.
[0]	CCS	RO	0y0	Current Connect Status. 1 = Device is present on port. 0 = No device is present. The default is 0. This value reflects the current status of the port, and may not correspond concurrently to the event that caused the Connect Status Change bit to be set to 1. This field is 0 if Port Power = 0.

3.16.9.3 Frame Length Adjustment Registers

Frame Length Adjustment

Address = (0x4000_5000) + 0x0304

Bit	Bit Symbol	Type	Reset Value	Description
[31:26]	–	–	Undefined	Read as zero. Write as zero.
[25:20]	FLTV	R/W	0x20	Frame Length Timing Value. This value is used to adjust any offset from the clock source that drives the SOF generation counter. The default value is 0y20, which gives a SOF cycle time of 60000 bits. Frame Length FLADJ Value (HS bit times) 59488 0y00 59504 0y01 59520 0y02 … 59984 0y1F 60000 0y20 (default) … 60480 0y3E 60496 0y3F Note) This value can only be changed when the HcHalted bit in the USBSTS register is 1. Changing the value of this register in other situations produces undefined results.
[19:16]	–	R/W	0y0101	To change the FLTV, be sure to set this bit to 0y0101.
[15:0]	–	–	Undefined	Read as zero. Write as zero.

D. USBHCREG registers

This section describes the function registers unique to this product. These registers relate to the EHCI.

USBHCREG01

This can change the AHB burst transfer threshold value for the packet buffer and the threshold value of OUT transfer start. For transfer models, refer to Section 3.16.7. The default value is set to 256 bytes.

[31:16] =AHB read threshold value (for OUT transfer)

The AHB read threshold value and the OUT transfer start threshold value use a shared value.

[15:0] = AHB write threshold value (for IN transfer)

Threshold values for when the setting is changed are shown below:

(Setting examples)

0x01000100	1024-byte threshold value
0x00800080	512-byte threshold value
0x00400040	256-byte threshold value (default)
0x00200020	128-byte threshold value

USBHCREG03

This can change the setting of maximum number of burst transfers executed by the AHB master of the USBHC. In this product, this register acts on only OUT transfer.

(1) AHB burst transfer in OUT transfer (reads transmission data from system memory.)

When the USBHCREG03 register is set to 0x00000001: Executes AHB burst transfer based on the AHB read threshold value set in this register. For transfer models, refer to Section 3.16.7.

(2) AHB burst transfer in IN transfer (Writes data received from a USB device to system memory.)

Executes AHB burst transfer based on the AHB write threshold value set in this register.
For transfer models, refer to Section 3.16.7. The setting of the USBHCREG03 register has no effect in IN transfer.

3.16.10 HostSystem Error, Unrecoverable Error

3.16.10.1 Error Conditions, and Operation and Recovery when the Error Occurs (EHCI)

This section describes the occurrence conditions of Host System Error (USBSTS[4]) at the EHCI, and the operation and recovery when it occurs.

<Conditions>

A Host System Error occurs when any of the following conditions is met when the EHCI has ownership and when host software operates as the AHB master:

(1) When the "sys_interrupt_i" signal is asserted

(2) When an ERROR(0y01) response is received with the "ahb_hresp_i" signal when data is being transferred

<Operation after error>

The EHCI core operates as follows when the above conditions are detected:

- Sets the Run/Stop(USBCMD[0]) bit to 0.

- The following bits in the USBSTS register will be set:
  Sets the HostSystemError(USBSTS[4]) bit to 1.
  Sets the HCHalted(USBSTS[12]) bit to 1.
- When the HostSystemErrorEnable(USBINTR[4]) bit is enabled (set to 1), the EHCI core will issue the hardware interrupt signal (ehci_interrupt_o) (no interrupt delay occurs).
  After a HostSystemError occurs and then the interrupt is detected, software needs to recover the Host Controller with the following procedure:

<Recovery>

After a HostSystemError occurs and then the interrupt is detected, software needs to recover the Host Controller with the following procedure:

(1) Set the HCReset[1] bit in the USBCMD register to 1 to reset the EHCI operational register. After the reset process has completed, the Host Controller will clear the HCReset bit.

(2) After confirming that the HCReset bit has been cleared to 0, software will re-set the EHCI operational register.

3.16.10.2 Unrecoverable Error Conditions, and Operation and Recovery when the Error Occurs (OHCI)

This section describes the occurrence conditions of Unrecoverable Error (UE:HCInterrupt Status[4]) at the OHCI, and the operation and recovery when it occurs.

<Conditions>

A Host System Error occurs when any of the following conditions is met when the OHCI has ownership and when host software operates as the AHB master:

(1) When the "sys_interrupt_i" signal is asserted

(2) When an ERROR(0y01) response is received with the "ahb_hresp_i" signal

3.16.11 UnderRun/OverRun Conditions

3.16.11.1 OHCI

Since the size of the data-transfer FIFO provided in the OHCI is 64 bytes, UnderRun/OverRun may occur only at FS isochronous where data of up to 1023 bytes can be transmitted or received.

- UnderRun conditions at isochronous OUT transfer
  When the read transfer latency on the AHB bus is 10.6 μs or more, UnderRun may occur. Since the maximum burst transfer size of the OHCI core is 16 bytes, if the next 16 bytes of data will not be written before the FIFO is emptied after data of 16 bytes has accumulated in the FIFO, a buffer underrun occurs. Make the system guaranteed to be able to read the next data within 10.6 μs.

- OverRun conditions at isochronous IN transfer
  When the write transfer latency on the AHB bus is 10.6 μs or more, OverRun may occur. After data of 64 bytes has been accumulated in the FIFO, if the data in the FIFO is not read out onto the AHB bus, a buffer overrun occurs. Make the system guaranteed to be able to read data out to the AHB bus within 10.6 μs. The maximum burst transfer size is 16 bytes.

As for FS Bulk Out, FS Interrupt Out, and FS Control Out, OUT transfer is started only after all transfer data has been read into the FIFO. Thus, UnderRun never occurs.
As for FS Bulk In, FS Interrupt In, and FS Control In, there is the physical capacity to be able to write all transfer data sent from devices into the FIFO and thus OverRun never occurs.

3.16.11.2 EHCI

The conditions of possible UnderRun and OverRun occurrence are as follows:

- OUT transfer: 256 bytes < Transfer data size
  This byte count is called the AHB read and write threshold value.

Note) You cannot set a threshold value exceeding the packet buffer capacity.

The following explains this using typical two examples:

Condition 1: Packet buffer of 1024 bytes, Threshold value of 256 bytes (Default)
Condition 2: Packet buffer of 1024 bytes, Threshold value of 1024 bytes

- Possibility of UnderRun occurrence

× : UnderRun may occur
○ : UnderRun will not occur

	Condition 1	Condition 2
Isochronous OUT:	×	○
Interrupt OUT:	×	○
Bulk OUT:	×	○
Control OUT:	○	○

In Split-* transfer, "○" applies in any condition.

Since data up to 256 bytes, which is equivalent to the data amount defined for the threshold value, has not been accumulated in the internal FIFO, UnderRun will occur if the next data cannot be read before this 256-byte data has been output to the USB bus. Make the system guaranteed to be able to read next data within 4.2667 µs.

- Possibility of OverRun occurrence

In any transfer, no OverRun occurs. This is because the packet buffer size of this product is 1024 bytes whereas the maximum transfer size of isochronous transfer is 1024 bytes.

3.16.12 Test Modes

Test modes to facilitate compliance tests are implemented. This section describes the methods defined in the EHCI.

3.16.12.1 Procedures

1. Connect the USBHC with the fixture's SQ HOST pin, and so on (*).

(*) Fixture's connection destinations vary depending on the Test Mode.

Test Mode	Pin name
Test J_STATE	SQ Host
Test K_STATE	
Test SE0_NAK	
Test Packet	
Test FORCE_ENABLE	Device Signal Quality

2. To execute HCRESRT, write the following value in the USBCMD register.
   The USBHC clears the HCRESET bit after HCRESET has been completed.

   USBCMD: 0x00000002

3. To set the EHCI to be the owner, write the following value in the CONFIGFLAG register.

   CONFIGFLAG: 0x00000001

4. To turn on the port power of the EHCI, write the following value in the PORTSC register.

   PORTSC: 0x00001000

5. To transition to the Test Mode, write the following value in the PORTSC register.

   [19:16] Port Test Control= 0x4 (For Test Packet)

There are the following Test Mode types:

Port Test Control[19:16]	**Test Mode**
0x1	Test J_STATE
0x2	Test K_STATE
0x3	Test SE0_NAK
0x4	Test Packet
0x5	Test FORCE_ENABLE

6. This step is required only when the register is set to Test FORCE_ENABLE.
   Write the following value in the USBCMD register.

   [0]Run/Stop = 0y1

7. Perform each measurement.

3.16.12.2 How to Change the Setting

To change the Test Mode setting, you need to exit the Test Mode once. In other words, you can not dynamically change Test Modes.

(Example)
When Test Mode is set to 0x1 (Test J), even a write of 0x2 (Test K) to the register cannot execute Test K.

3.16.12.3 How to Exit

To execute HCRESRT, write the following value in the USBCMD register.
The USBHC clears the HCRESET bit after HCRESET has been completed.

USBCMD: 0x00000002

3.17 SD Host Controller

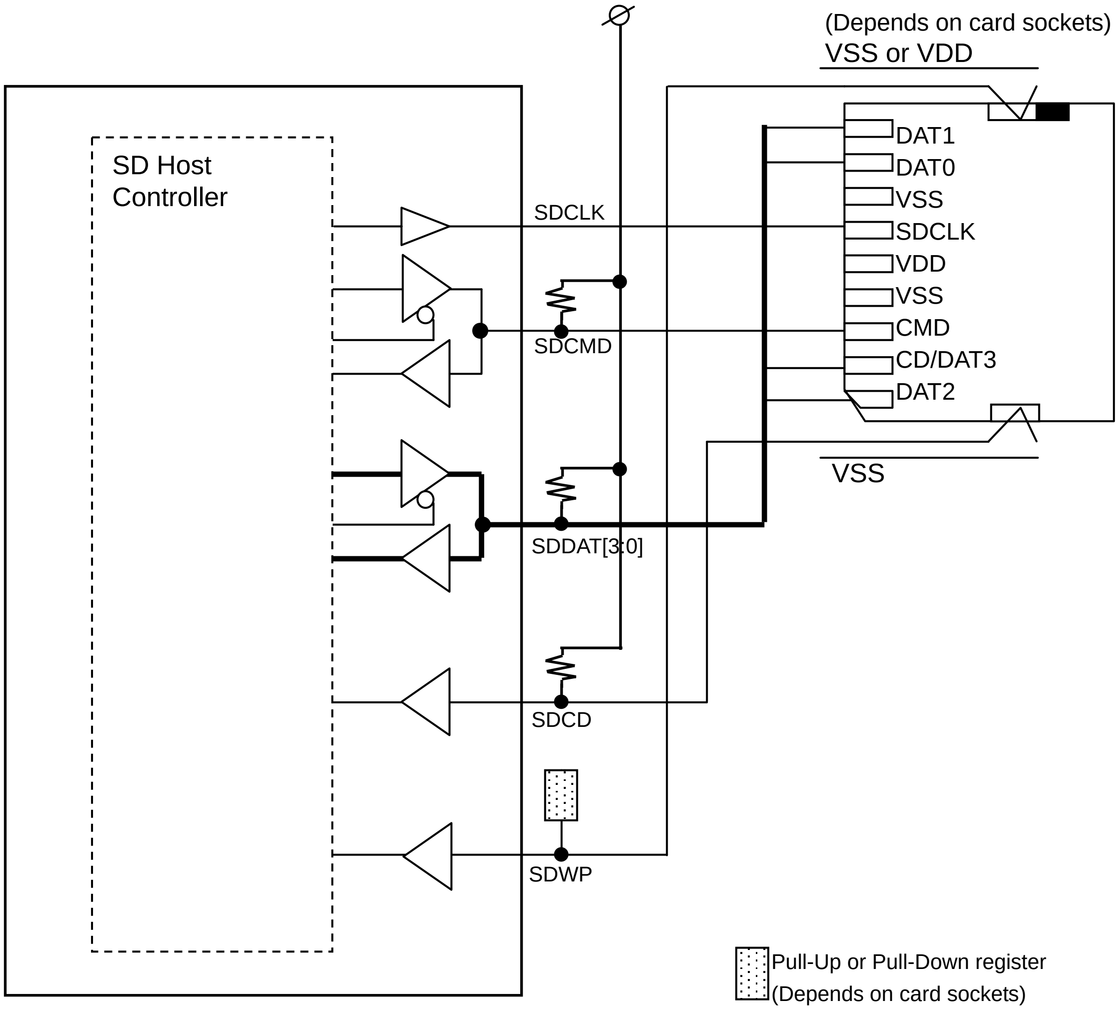
3.17.1 Function Overview

The following describes the functions and features of the SD host controller:

1) Data control on a frame basis
2) Can set transfer data length (sector) from 1 byte to 512 bytes (Set to 512 bytes for multi read).
3) Can set the number of sectors in multi read and write.
4) Transmit/receive error check: CRC7 (for command line), CRC16 (for data line)
5) Sync type: Bit sync by SDCLK
6) Can set the SDCLK frequency up to system clock (HCLK) 1/4 to 1/1024.
7) SD memory card interface: Command (1 bit), data (1/4 bit)
8) SDIO interfaces are not supported.
9) Number of support ports: 1 card
10) 512 bytes × 2 data buffers: 256 words × 16 bits × 2
11) Card detection function (SDCD or SDDAT3)
12) Data write protect function support
13) Detecting a status error state
    SD buffer underflow/overflow
    Various types of timeouts (response, etc.)
    END bit, CRC, CMD error
14) Various types of response frame formats can be recognized by register setting.

Note) The support for the access to the SD host controller including the registers is for 16-bit bus width only. You cannot make 32-bit access.

3.17.2 Example of SD Memory Card System



Note)

For card detection by SDDAT3, the lines need to be pulled down.

Pull-down resistors of 300 kΩ or more are recommended.

For more information, refer to the SD Memory Card Application Notes issued by SDA.

About the License

To use this IP, you must first conclude a contract on the SD HOST/ANCILLARY PRODUCT LICENSE AGREEMENT with SD-3C, LLC, a limited liability company in Delaware in the US, and the SD Card Association, a nonprofit company in California in the US. If no contract is concluded, you cannot use this function.
(http://www.sdcard.org/)

To use the CPRM technology of this IP, you must first conclude a contract on the 4C CPRM/CPRM LICENSE AGREEMENT or the CPRM for SD-BINDING LICENSE AGREEMENT with 4CEntity, LLC, a limited liability company in Delaware in the US. If no contract is concluded, you cannot use this function.
(http://www.4centity.com/)

* For the detailed specifications of this circuit, you need to conclude a separate confidentiality agreement with us.
For more information, please contact our sales representative.

3.18 Analog/Digital Converter

A 10-bit serial conversion analog/digital converter (AD converter) with four channels of analog input is built in.

Figure 3.18.1 shows the block diagram of the AD converter. (The four channels of analog input pins (AIN0-AIN3) are dedicated pins.)

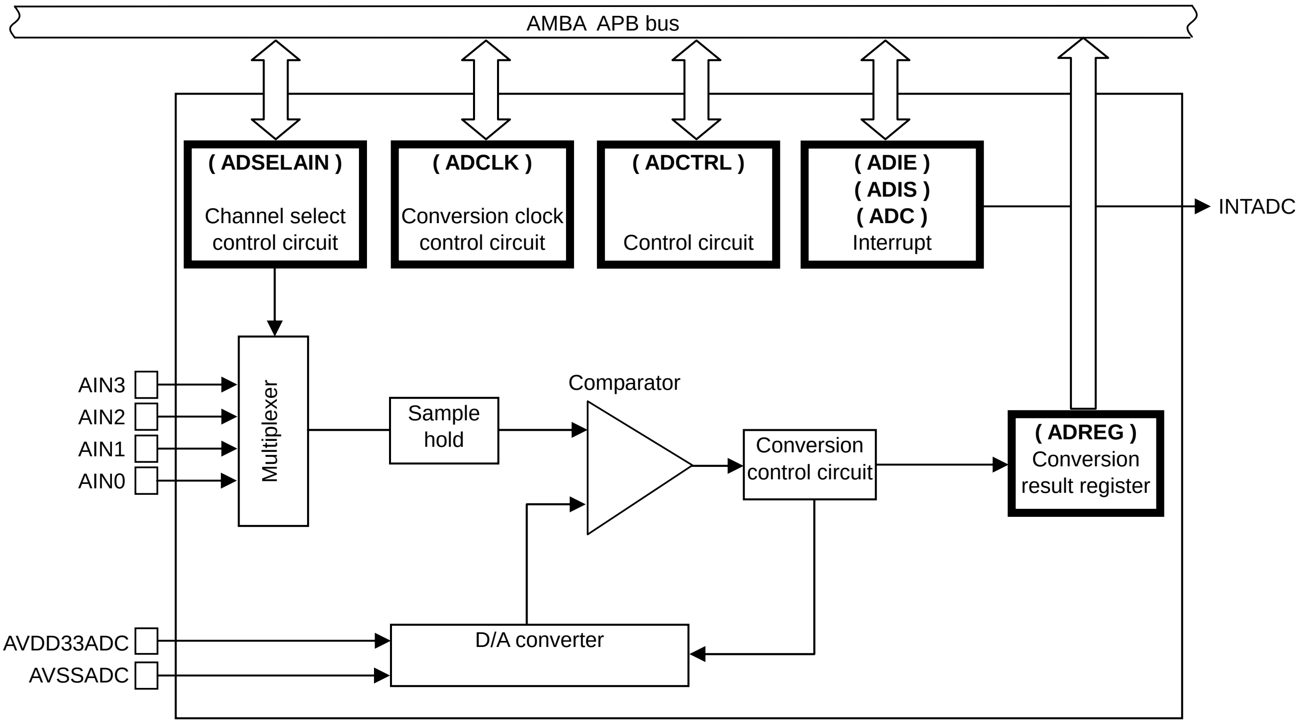


Figure 3.18.1 AD converter block diagram

3.18.1 Registers

The following lists the AD converter related registers: Adding the base address gives the address of this product.

base address = 0x4000_7000

Register Name	Address (base+)	Description
ADCTRL	0x0000	A/D control register
ADSELAIN	0x0004	A/D channel select register
ADREG	0x0008	A/D conversion result register
ADCLK	0x000C	A/D conversion clock setting register
ADIE	0x0010	A/D interrupt enable register
ADIS	0x0014	A/D interrupt status register
ADIC	0x0018	A/D interrupt clear register

Note) Notes for registers

R/W: Read/Write possible

RO: Readable/Write not reflected

WO: Writable/"0" is read when read

1. ADCTRL (AD Control Register)

base address = (0x4000_7000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read undefined. Write as zero.
[2:1]	ADRST[1:0]	R/W	0y00	Resets the ADC software by writing 10 -> 01. Initializes all except the (ADCLK) register.
[0]	ADS	R/W	0y0	A/D conversion start 0: Stop or conversion ended 1: Conversion started and being converting

R/W: Read/Write RO: Read Only WO: Write Only

Note 1) ADS is cleared to 0 when conversion ends.

Note 2) A write to ADS when AD conversion is being executed is invalid.

[Explanation]

a. <ADRST[1:0]>

Resets the ADC software by writing 10 -> 01. Initializes all registers except the A/D conversion clock setting register.

b. <ADS>

AD conversion is started up by software by setting ADCTRL<ADS> to "1."

However, starting AD conversion immediately after resetting the software will not start the conversion. Start conversion at least after the PCLK1.

Note that making a start at the same time with AD conversion end will not start conversion.

(WR)

Selects A/D conversion start.

0: Don't care

1: Conversion start

(RD)

Shows the state of AD conversion status.

0: Stop or conversion ended

1: Being converting

2. ADSELAIN (AD Channel Select Register)

base address = (0x4000_7000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1:0]	SELAIN[1:0]	R/W	0y00	Analog input channel select 00: AIN0, 01: AIN1, 10: AIN2, 11: AIN3

R/W: Read/Write RO: Read Only WO: Write Only

Note 1) Select the input channel before starting AD conversion.
Note 2) A WR to SELAIN when AD conversion is being executed is invalid.

[Explanation]

a. <SELAIN[1:0]>

Selects the analog input channel.

00: AIN0

01: AIN

10: AIN2

11: AIN3

3. ADREG (AD Conversion Result Register)

base address = (0x4000_7000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	0x0000	Read as zero.
[15:10]	–	–	0y000000	Read as zero.
[9]	ADR9	RO	0y0	AD conversion result bit 9
[8]	ADR8	RO	0y0	AD conversion result bit 8
[7]	ADR7	RO	0y0	AD conversion result bit 7
[6]	ADR6	RO	0y0	AD conversion result bit 6
[5]	ADR5	RO	0y0	AD conversion result bit 5
[4]	ADR4	RO	0y0	AD conversion result bit 4
[3]	ADR3	RO	0y0	AD conversion result bit 3
[2]	ADR2	RO	0y0	AD conversion result bit 2
[1]	ADR1	RO	0y0	AD conversion result bit 1
[0]	ADR0	RO	0y0	AD conversion result bit 0

R/W: Read/Write RO: Read Only WO: Write Only

[Explanation]

a. <ADR9-0>

AD conversion result bits 9 to 0

4. ADCLK (AD Conversion Clock Setting Register)

base address = (0x4000_7000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read undefined. Write as zero.
[2:0]	ADCLK[1:0]	R/W	0y00	AD prescaler output select AD conversion 1-clock period = 000: PCLK/16 001: PCLK/32 010: PCLK/64 011: PCLK/128 100: PCLK/1 101-111: Reserved (Setting not available)

R/W: Read/Write RO: ReadOnly WO: WriteOnly

Note 1) A WR to ADCLK when AD conversion is being executed is invalid.

[Explanation]

a. <ADCLK[2:0]>

Selects the AD prescaler output.

AD conversion 1-clock period =

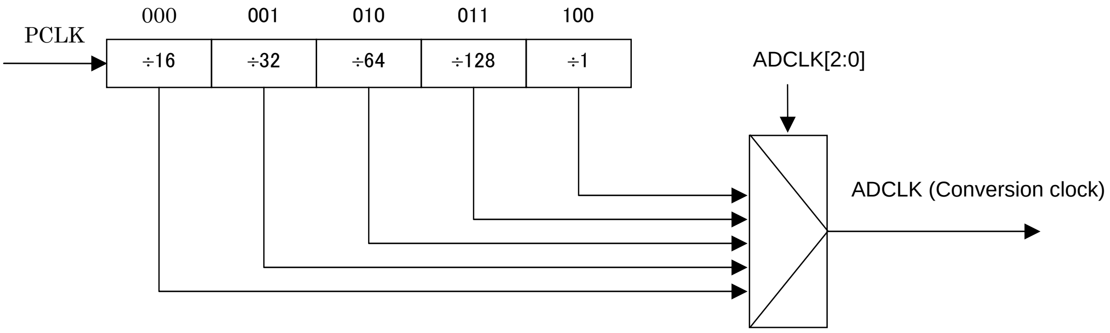
000: PCLK/16

001: PCLK/32

010: PCLK/64

011: PCLK/128

100: PCLK/1



PCLK	<ADCLK2:0>	ADCLK	AD conversion time
144 MHz	000	9 MHz	2.55 μsec
	001	4.5 MHzz	4.11 μsec
	010	2.25 Mhz	7.22 μsec
	011	1.125 MHz	13.44 μsec
12 MHz	100	12 MHz	2.08 μsec

Note) Do not make settings other than the above.

AD conversion time can be approximately determined with the following equation (Value determined from AD conversion start):

Conversion time = 14 × (1/ADCLK) + 160 × (1/PCLK) (Conversion setup time)

Note) ADCLK supports 1 MHz to 12 MHz.

5. ADIE (AD Interrupt Enable Register)

base address = (0x4000_7000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	NIE	R/W	0y0	AD conversion interrupt enable 0: Disable 1: Enable

R/W: Read/Write RO: Read Only WO: Write Only

[Explanation]

a. <NIE>

Controls the AD conversion interrupt.

0 : Disable

1 : Enable

6. ADIS (AD Interrupt Status Register)

base address = (0x4000_7000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	NIS	RO	0y0	Status of before masking an AD conversion interrupt 0: No interrupt 1: Interrupt occurred

R/W: Read/Write RO: Read Only WO: Write Only

[Explanation]

a. <NIS>

Shows the status before masking an AD conversion interrupt.

0: No interrupt

1: Interrupt occurred

7. ADIC (AD Interrupt Clear Register)

base address = (0x4000_7000) + 0x0018

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	NIC	WO	0y0	AD conversion interrupt clear 0: – 1: Clear

R/W: Read/Write RO: Read Only WO: Write Only

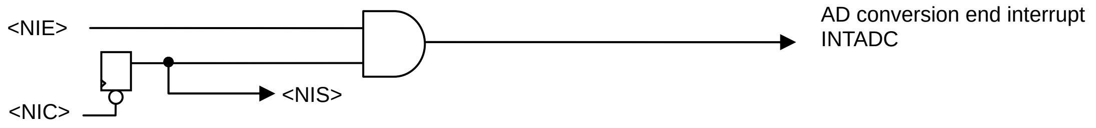
[Explanation]

a. <NIC>

Controls the AD conversion interrupt.

0: –

1: Clear



3.18.2 Functional Description

Selecting Analog Input Channels

Select one channel from the analog input pins AIN0 to AIN3 by setting ADSELAIN<SELAIN[1:0]>.

AD Conversion Start

AD conversion is started up by software by setting ADCTRL<ADS> to "1."

Actual conversion starts after "AD setup time 160 × (1/PCLK)" has elapsed.

AD Conversion Underway, and AD Conversion End Interrupt

After AD conversion is started, ADCTRL<ADS> is set to "1." When a specified AD conversion ends, ADCTRL<ADS> is cleared to "0" and an AD conversion end interrupt occurs.

Note that a WR to <ADS>, <SELAIN[1:0]>, and <ADCLK[2:0]> when conversion is being executed is invalid.

AD conversion time

AD conversion time corresponds to "Setup time 160 × (1/PCLK)" and "Sampling clock 14 × (1/ADCLK) clocks." The sampling clock is 1/16, 1/32, 1/64, 1/128, or 1/1 of the PCLK depending on <ADCLK[2:0]>.

Storage and Reading of AD Conversion Results

Conversion results are stored in the AD conversion result register (ADREG).

Data Polling

To process AD conversion results by using data polling without using interrupts, perform polling on ADCTRL<ADS>. After confirming that ADCTRL<ADS> is cleared to "0," read the AD conversion result register.

Forcible Stop of AD Conversion

To cancel a conversion when it is being executed, use ADCTRL<ADRST[1:0]>.

At that time, all registers except the (ADCLK) register are initialized.

Software reset is cleared after 34 × (1/PCLK).

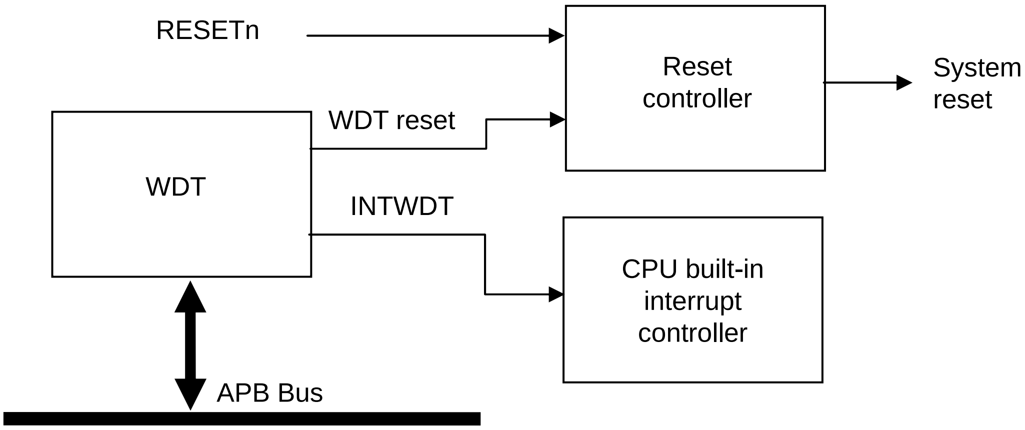
3.21 Watchdog Timer (Runaway Detection Timer WDT)

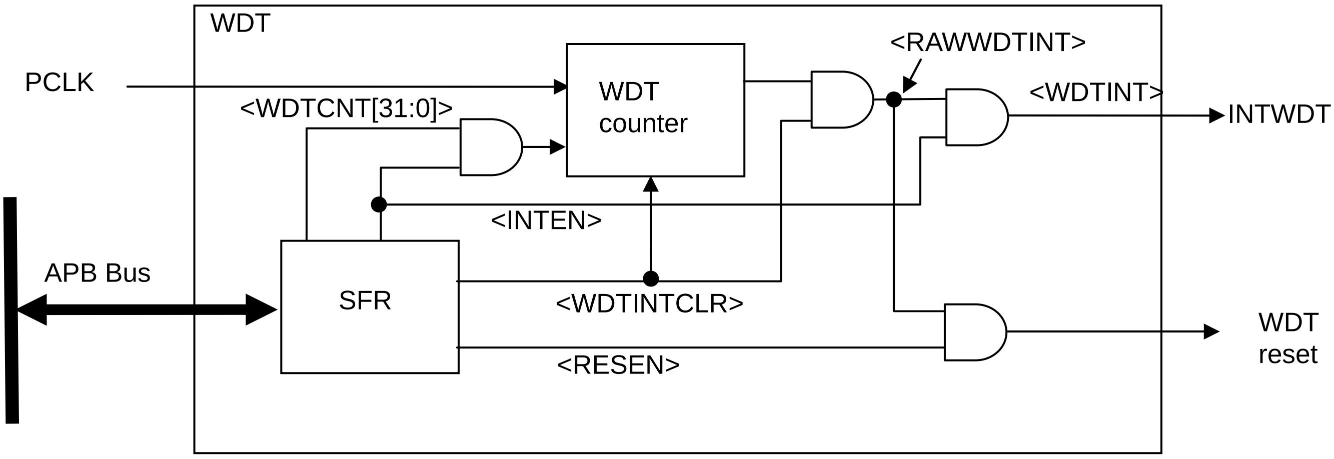
A watchdog timer for detecting a runaway is built in.

The watchdog timer (WDT) serves the purpose of detecting a CPU malfunction (runaway) started due to causes such as noise and then restoring it to the normal condition. When the watchdog timer detects a runway, it generates an interrupt to report it to the interrupt controller (NVIC) built into the CPU (the interrupt source signal to the interrupt controller is INTWDT).

In addition, connecting this watchdog timer OUT to the internal reset can perform a reset operation forcibly.

- 3.21.1 Block Diagram





3.21.11 Register Functions

The built-in registers and their functions are listed below.

base address = 0x4000_6000

Register Name	Address (base+)	Description
WdogLoad	0x0000	Watchdog load register
WdogValue	0x0004	The current value for the watchdog counter
WdogControl	0x0008	Watchdog control register
WdogIntClr	0x000C	Clears the watchdog interrupt
WdogRIS	0x0010	Watchdog raw interrupt status
WdogMIS	0x0014	Watchdog masked interrupt status
WdogLock	0x0C00	Watchdog Lock register

1. WdogLoad (Watchdog load register)

Address = (0x4000_6000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	WDTCNT	R/W	0xFFFFFFFF	Set value of WDT detection counter 0x00000001 to 0xFFFFFFFF

[Explanation]

a. <WDTCNT>

This sets the value of the WDT 32-bit counter (the clock of the WDT detection counter is PCLK).

After WdogControl<INTEN>, which is described later, is enabled, the set value of WdogLoad<WDTCNT> is loaded to the internal decrement counter.

The range of values that can be set for the counter is 0x00000001 to 0xFFFFFFFF. (Zero cannot be set.)

When this bit is read, the set value will be read.

2. WdogValue (The current value for the watchdog counter)

Address = (0x4000_6000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	CWDTCNT	RO	0xFFFFFFFF	Current WDT detection counter value

[Explanation]

a. <CWDTCNT>

This can read the current value of the watchdog counter.

3. WdogControl (Watchdog control register)

Address = (0x4000_6000) + 0x0008

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read undefined. Write as zero.
[1]	RESEN	R/W	0y0	WDT reset output enable 0y0: Disable 0y1: Enable
[0]	INTEN	R/W	0y0	WDT counter and interrupt enable 0y0: Disable 0y1: Enable

[Explanation]

a. <RESEN>

This bit controls the enable of WDT reset output.

The time that elapses before reset clear is approximately 10 μs (f_{OSCH} = 12 MHz, after 120 beats).

b. <INTEN>

0y1: This enables the WDT counter and reloads the counter’s set value from the WdogLoad register so that the counter starts decrementing. It also enables interrupts.

4. WdogIntClr (Clears the watchdog interrupt)

Address = (0x4000_6000) + 0x000C

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	WDTINTCLR	WO	Undefined	WDT interrupt clear (Arbitrary value)

[Explanation]

a. <WDTINTCLR>

Writing an arbitrary value in this register will clear the WDT interrupt. It also loads the WdogLoad register’s set value to the counter.

5. WdogRIS (Watchdog raw interrupt status)

Address = (0x4000_6000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined.
[0]	RAWWDTINT	RO	0y0	Interrupt status before the enable gate 0y0: No interrupt 0y1: Interrupt occurred

[Explanation]

a. <RAWWDTINT>

This shows the interrupt status from the WDT counter. It ANDs this value with the interrupt enable signal (WdogControl<INTEN>) to generate the interrupt after enabling (WdogMIS<WDTINT>).

6. WdogMIS (Watchdog masked interrupt status)

Address = (0x4000_6000) + 0x0014

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined.
[0]	WDTINT	RO	0y0	Interrupt status after the enable gate 0y0: No interrupt 0y1: Interrupt occurred

[Explanation]

a. <WDTINT>

This shows the interrupt status from the WDT counter. It reads the AND value obtained with WdogRIS<RAWWDTINT> and WdogControl<INTEN>.

7. WdogLock (Watchdog Lock register)

Address = (0x4000_6000) + 0x0C00

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	REGWEN	WO	undefined	Write enable to other WDT registers 0x1ACCE551: Enable Others: Disable (Enable by default)

Address = (0x4000_6000) + 0x0C00

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	Reserved	RO	undefined	Read undefined.
[0]	REGWENST	RO	0y0	Status of write disable to other WDT registers 0y0: Enable (Not locked) 0y1: Disable (Locked)

[Explanation]

a. <REGWEN>

To avoid WDT registers being rewritten due to causes including a program runway, you can disable the writes to the other WDT registers except this register.

The writing of data other than 0x1ACCE551 will disable the writes to the other WDT registers except this register.

The writing of 0x1ACCE551 will enable the writes to the other WDT registers except this register.

b. <REGWENST>

This shows the status of disabling or locking writing to other WDT registers.

3.20 I2S (The Inter-IC Sound BUS)

This function is a serial audio interface that conforms to the I2S bus standards.

It contains a 2-channel (stereo) voice Receiver and a 2-channel (stereo) voice Transmitter, which are independent of each other. To use this function, I2S supply clocks must be set from the external clock controller. For more information, refer to CG_FSCTRL and CG_I2SFSCTRL in Section 3.5.

The main features of this function are as follows:

Note) Frequency errors as shown in Tables 3.20.8 and 3.20.9 occur when this function is used for audio outputs.

a. About input data formats
- Supports front alignment and rear alignment in the I2S format.
- Supports each of the data lengths of 16, 18, 20, and 24 bits.
- Supports each of the numbers of slots of 32, 48, and 64.

b. About output data formats
- Supports front alignment and rear alignment in the I2S format.
- Supports each of the data lengths of 16, 18, 20, and 24 bits.
- Supports each of the numbers of slots of 32, 48, and 64.

c. Contains a bus I/F (slave interface) for the global bus.
- Supports two data transfer modes: ordinary transfer mode and 16-bit data transfer mode.

d. Each of the Receiver and the Transmitter has a FIFO of 8 words (for 4 fs) × 2 banks (A and B).

e. Provides 2 systems (1 input and 1 output) of DMA interface for the external DMAC.

f. Provides interrupt signals for error interrupts.

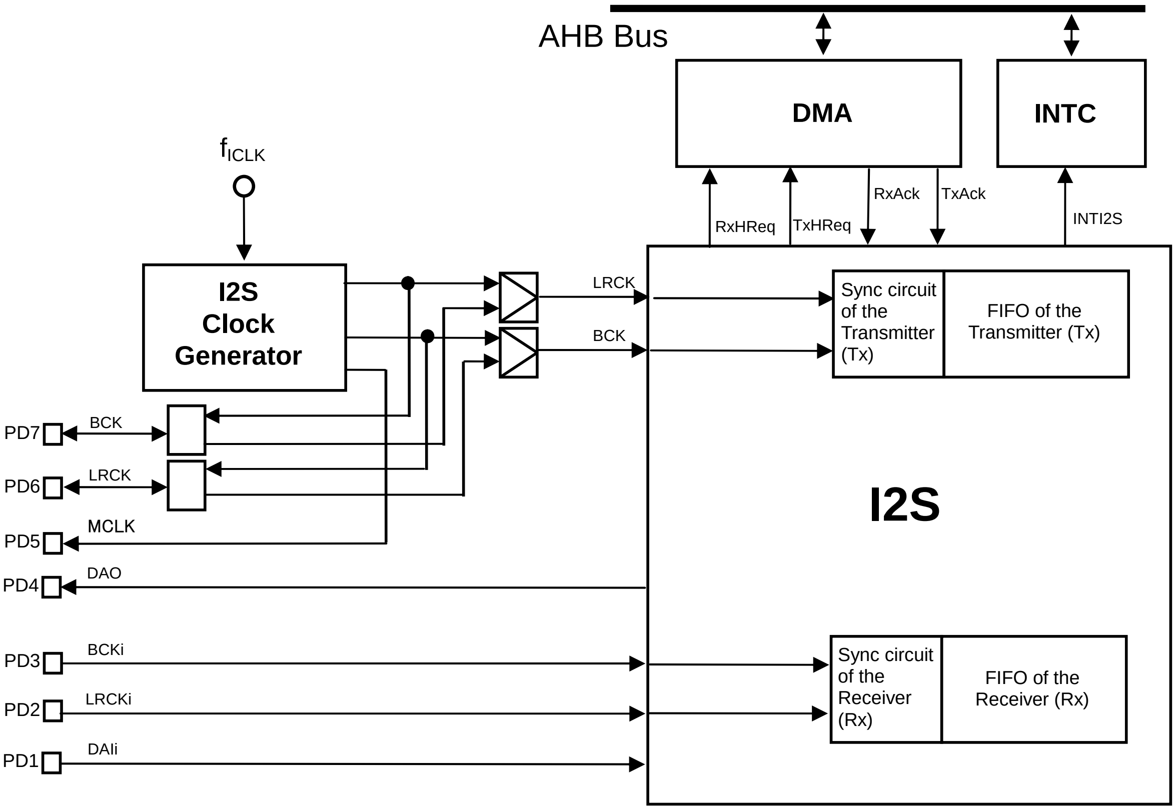


Figure 3.20-1 Block Diagram

3.20.1 External Connection

This section describes the method of connecting this function to the external I2S interface.

1. Data input (receive)

   Connect this function as shown below when it is used for data input:

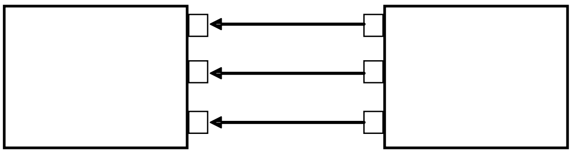


Figure 3.20-2 Example of External Connection (Data Receiver)

2. Data output (transmit)

   The figure below illustrates examples of connection of this function when it is used for data output. Use of MCLK and master/slave selection of LRCK and BCK are options. Set the master/slave selection using the built-in I2S_CG_CNT register (I2S Clock Generator Control Register).

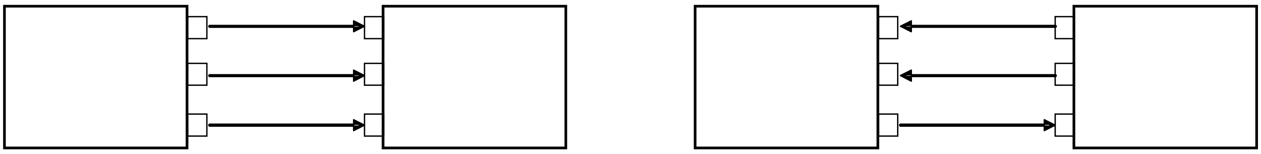


Figure 3.20-3 Example of External Connection (Data Transmitter)

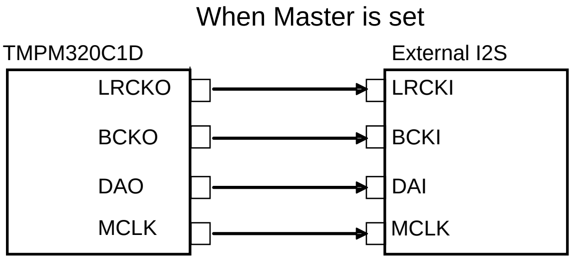


Figure 3.20-4 Example of External Connection When MCLK is Used (Data Transmitter)

3.20.2 Registers and their Functions

The following lists show the built-in registers and their functions:

Base address = 0x4001_6000

Register Name	Address (base+)	Description
I2S_IN_CONT	0x0000	I2S Input Control Register
I2S_IN_FIFOState	0x0010	I2S Input FIFO State Register
I2S_IN_FIFOData	0x0100	I2S Input FIFO Data Access Register
Reserved	0x0104 - 0x013C	-
I2S_OUT_CONT	0x0200	I2S Output Control Register
I2S_OUT_FIFOState	0x0210	I2S Output FIFO State Register
I2S_OUT_FIFOData	0x0300	I2S Output FIFO Data Access Register
Reserved	0x0304 - 0x033C	-
I2S_CG_CNT	0x03FC	I2S Clock Generator Control Register

1. I2S_IN_CONT (I2S Input Control Register)

Address = (0x4001_6000) + 0x0000

Bit	Bit Symbol	Type	Reset Value	Description
[31:25]	–	–	Undefined	Read undefined. Write as zero.
[24]	IOPEN	R/W	0y0	Operation Enable of the Receiver 0y0: Disable 0y1: Enable
[23:17]	–	–	Undefined	Read undefined. Write as zero.
[16]	IHREQ	R/W	0y0	Hardware Request (to DMAC) of the Receiver 0y0: de-assert / clear 0y1: assert / set
[15:9]	–	–	Undefined	Read undefined. Write as zero.
[8]	ITMODE	R/W	0y0	16-bit data transfer mode of the Receiver 0y0: Normal transfer mode 0y1: 16-bit data transfer mode
[7]	–	–	Undefined	Read undefined. Write as zero.
[6]	Reserved	R/W	0y0	Read undefined. Write as zero.
[5:4]	ISLT	R/W	0y11	Number of slots of the Receiver 0y00: 32 Slot 0y01: 48 Slot 0y10: 64 Slot 0y11: 64 Slot
[3:2]	IFMT	R/W	0y11	Data format of the Receiver 0y00: Front-aligned 0y01: Rear-aligned 0y10: I2S 0y11: I2S
[1:0]	ILNG	R/W	0y11	Data length of the Receiver 0y00: 16 bit 0y01: 18 bit 0y10: 20 bit 0y11: 24 bit

(Note) Do not access registers other than IOPEN during I2S operation (IOPEN = 1).

(Explanation)

a. < IOPEN >

This bit sets the operation mode of the I2S Receiver. The default value after resetting is Disable (Operation stops).

b. < IHREQ >

This bit monitors the Hardware Request status of the I2S Receiver. The default value after resetting is de-assert. The request can be cleared anytime by directly writing 0 to this bit.

(Note) Hardware Request is asserted if you write 1 to this bit during normal operation.

c. < ITMODE >

This bit selects the 16-bit data transfer mode of the I2S Receiver. The default value after resetting is normal transfer mode (32 bits).

d. < ISLT >

These bits set the number of slots of the I2S Receiver. The default value after resetting is 64 slots.

e. < IFMT >

These bits set the data format of the I2S Receiver. The default value after resetting is the I2S format.

f. < ILNG >

These bits set the data length of the I2S Receiver. The default value after resetting is 24 bits.

2. I2S_IN_FIFOState (I2S Input FIFO State Register)

Address = (0x4001_6000) + 0x0010

Bit	Bit Symbol	Type	Reset Value	Description
[31:25]	–	–	Undefined	Read undefined. Write as zero.
[24]	IFIFOBANK	RO	0y0	Access side of the FIFO block of the Receiver 0y0: Bank A 0y1: Bank B
[23]	–	–	Undefined	Read undefined. Write as zero.
[22:20]	IPOINT_BUS	RO	0y000	FIFO bus pointer (read) of the Receiver FIFO pointer being read by the bus (0y000 – 0y111)
[19]	–	–	Undefined	Read undefined. Write as zero.
[18:16]	IPOINT_I2S	RO	0y000	FIFO I2S pointer (write) of the Receiver FIFO pointer being written by I2S (0y000 – 0y111)
[15:13]	–	–	Undefined	Read undefined. Write as zero.
[12]	IFIFOINT	R/W	0y0	FIFO Interrupt Enable of the Receiver 0y0: Disable interrupts 0y1: Enable interrupts
[11:9]	–	–	Undefined	Read undefined. Write as zero.
[8]	IFIFOCLR	R/W	0y0	FIFO Interrupt Clear of the Receiver 0y0: Invalid 0y1: Clear the overflow, underflow, or error bit
[7:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	IFIFOOVR	RO	0y0	FIFO Overflow of the Receiver 0y0: No overflow has occurred. 0y1: Overflow occurred.
[1]	IFIFOUDR	RO	0y0	FIFO Underflow of the Receiver 0y0: No underflow has occurred. 0y1: Underflow occurred.
[0]	IFIFOERR	RO	0y0	FIFO Error of the Receiver 0y0: No error has occurred. 0y1: Error occurred.

(Explanation)

a. <IFIFOBANK>

This bit indicates the access bank of the FIFO of the I2S Receiver. The default value after resetting is bank A.

b. <IPOINT_BUS>

These bits indicate the FIFO bus pointer of the I2S Receiver. The default value after resetting is 0y000.

c. <IPOINT_I2S>

These bits indicate the FIFO I2S pointer of the I2S Receiver. The default value after resetting is 0y000.

d. <IFIFOINT>

This bit outputs an interrupt signal by FIFO Interrupt Enable of the I2S Receiver. The default value after resetting is Interrupt Disabled.

e. <IFIFOCLR>

This bit clears FIFO status of the I2S Receiver. The default value after resetting is Invalid. The IFIFOOVR/IFIFOUDR IFIFOERR bit can be cleared by writing 1 to this bit. After that, this bit automatically returns to 0. An interrupt can be cleared by writing 1 to this bit when the interrupt is being output in the FIFO Interrupt Enabled status.

f. <IFIFOOVR>

This bit indicates that the required amount of data (8 words) has not yet been read from the bus on FIFO bank switching in the I2S Receiver. The default value after resetting indicates that no overflow has occurred.

g. <IFIFOUDR>

This bit indicates that the required amount of data (8 words) or more has been read from the bus on FIFO bank switching in the I2S Receiver. The default value after resetting indicates that no underflow has occurred.

h. <IFIFOERR>

This bit indicates that either overflow or underflow has occurred in a FIFO of the I2S Receiver. The default value after resetting indicates that neither has occurred.

3. I2S_IN_FIFOData (I2S Input FIFO Data Access Register)

Address = (0x4001_6000) + 0x0100

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	IFIFO_DATA	RO	0x00000000	FIFO Data Read of the Receiver Normal transfer mode Upper 24 bits [31:8]: Audio data Lower 8 bits [7:0]: Fixed to 0x00 16-bit data transfer mode Upper 16 bits [31:16]: R-ch audio data Lower 16 bits [15:0]: L-ch audio data

(Explanation)

a. <IFIFO_DATA>

These bits indicate the data to be read into the FIFO of the I2S Receiver. The default value after resetting is 0x00000000. The data is MSB-aligned and the empty space becomes 0.

4. I2S_OUT_CONT (I2S Output Control Register)

Address = (0x4001_6000) + 0x0200

Bit	Bit Symbol	Type	Reset Value	Description
[31:25]	–	–	Undefined	Read undefined. Write as zero.
[24]	OOPEN	R/W	0y0	Operation Enable of the Transmitter 0y0: Disable 0y1: Enable
[23:17]	–	–	Undefined	Read undefined. Write as zero.
[16]	OHREQ	R/W	0y0	Hardware Request (to DMAC) of the Transmitter 0y0: de-assert / clear 0y1: assert / set
[15:9]	–	–	Undefined	Read undefined. Write as zero.
[8]	OTMODE	R/W	0y0	16-bit data transfer mode of the Transmitter 0y0: Normal transfer mode 0y1: 16-bit data transfer mode
[7]	–	–	Undefined	Read undefined. Write as zero.
[6]	Reserved	R/W	0y0	Read undefined. Write as zero.
[5:4]	OSLT	R/W	0y11	Number of slots of the Transmitter 0y00: 32 Slot 0y01: 48 Slot 0y10: 64 Slot 0y11: 64 Slot
[3:2]	OFMT	R/W	0y11	Data format of the Transmitter 0y00: Front-aligned 0y01: Rear-aligned 0y10: I2S 0y11: I2S
[1:0]	OLNG	R/W	0y11	Data length of the Transmitter 0y00: 16 bit 0y01: 18 bit 0y10: 20 bit 0y11: 24 bit

(Note) Do not access registers other than OOPEN during I2S operation (OOPEN = 1).

(Explanation)

a. < OOPEN >

This bit sets the operation mode of the I2S Transmitter. The default value after resetting is Disable (Operation stops).

b. < OHREQ >

This bit monitors the Hardware Request status of the I2S Transmitter. The default value after resetting is de-assert. The request can be cleared anytime by directly writing 0 to this bit.

(Note) Hardware Request is asserted if you write 1 to this bit during normal operation.

c. < OTMODE >

This bit selects the 16-bit data transfer mode of the I2S Transmitter. The default value after resetting is normal transfer mode (32 bits).

d. < OSLT >

These bits set the number of slots of the I2S Transmitter. The default value after resetting is 64 slots.

e. < OFMT >

These bits set the data format of the I2S Transmitter. The default value after resetting is the I2S format.

f. < OLNG >

These bits set the data length of the I2S Transmitter. The default value after resetting is 24 bits.

5. I2S_OUT_FIFOState (I2S Output FIFO State Register)

Address = (0x4001_6000) + 0x0210

Bit	Bit Symbol	Type	Reset Value	Description
[31:25]	–	–	Undefined	Read undefined. Write as zero.
[24]	OFIFOBANK	RO	0y0	Access side of the FIFO block of the Transmitter 0y0: Bank A 0y1: Bank B
[23]	–	–	Undefined	Read undefined. Write as zero.
[22:20]	OPOINT_BUS	RO	0y000	FIFO bus pointer (write) of the Transmitter FIFO pointer being written by the bus (0y000 – 0y111)
[19]	–	–	Undefined	Read undefined. Write as zero.
[18:16]	OPOINT_I2S	RO	0y000	FIFO I2S pointer (read) of the Transmitter FIFO pointer being read by I2S (0y000 – 0y111)
[15:13]	–	–	Undefined	Read undefined. Write as zero.
[12]	OFIFOINT	R/W	0y0	FIFO Interrupt Enable of the Transmitter 0y0: Disable interrupts 0y1: Enable interrupts
[11:9]	–	–	Undefined	Read undefined. Write as zero.
[8]	OFIFOCLR	R/W	0y0	FIFO Interrupt Clear of the Transmitter 0y0: Invalid 0y1: Clear the overflow, underflow, or error bit
[7:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	OFIFOOVR	RO	0y0	FIFO Overflow of the Transmitter 0y0: No overflow has occurred. 0y1: Overflow occurred.
[1]	OFIFOUDR	RO	0y0	FIFO Underflow of the Transmitter 0y0: No underflow has occurred. 0y1: Underflow occurred.
[0]	OFIFOERR	RO	0y0	FIFO Error of the Transmitter 0y0: No error has occurred. 0y1: Error occurred.

(Explanation)

a. <OFIFOBANK>

This bit indicates the access bank of the FIFO of the I2S Transmitter. The default value after resetting is bank A.

b. <OPOINT_BUS>

These bits indicate the FIFO bus pointer of the I2S Transmitter. The default value after resetting is 0y000.

c. <OPOINT_I2S>

These bits indicate the FIFO I2S pointer of the I2S Transmitter. The default value after resetting is 0y000.

d. <OFIFOINT>

This bit outputs an interrupt signal by FIFO Interrupt Enable of the I2S Transmitter. The default value after resetting is Interrupt Disabled.

e. <OFIFOCLR>

This bit clears the FIFO status of the I2S Transmitter. The default value after resetting is Invalid. The OFIFOOVR/OFIFOUDR OFIFOERR bit can be cleared by writing 1 to this bit. After that, this bit automatically returns to 0. An interrupt can be cleared by writing 1 to this bit when the interrupt is being output in the FIFO Interrupt Enabled status.

f. <OFIFOOVR>

This bit indicates that more than the required amount of data (8 words) has been written from the bus on FIFO bank switching in the I2S Transmitter. The default value after resetting indicates that no overflow has occurred.

g. <OFIFOUDR>

This bit indicates that the required amount of data (8 words) has not yet been written from the bus on FIFO bank switching in the I2S Transmitter. The default value after resetting indicates that no underflow has occurred.

h. <OFIFOERR>

This bit indicates that either overflow or underflow has occurred in a FIFO of the I2S Transmitter. The default value after resetting indicates that neither has occurred.

6. I2S_OUT_FIFOData (I2S Output FIFO Data Access Register)

Address = (0x4001_6000) + 0x0300

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	OFIFO_DATA	WO	undefined	FIFO Data Write of the Transmitter Normal transfer mode Upper 24 bits [31:8]: Audio data Lower 8 bits [7:0]: Fixed to 0x00 16-bit data transfer mode Upper 16 bits [31:16]: R-ch audio data Lower 16 bits [15:0]: L-ch audio data

(Explanation)

a. <OFIFO_DATA>

These bits indicate the data to be read into the FIFO of the I2S Receiver. The default value after resetting is undefined. The data must be MSB-aligned with 0 written to idle bits.

7. I2S_CG_CNT (I2S Clock Generator Control Register)

Address = (0x4001_6000) + 0x03FC

Bit	Bit Symbol	Type	Reset Value	Description
[31:13]	–	–	Undefined	Read undefined. Write as zero.
[12]	LRCKOSEL	R/W	0y0	LRCK Clock Selection of the Transmitter 0y0: Select the clock internally generated by CG. 0y1: Select the external input clock.
[11]	BCKOSEL	R/W	0y0	BCK Clock Selection of the Transmitter 0y0: Select the clock internally generated by CG. 0y1: Select the external input clock.
[10]	–	–	Undefined	Read undefined. Write as zero.
[9]	LRCKIPOL	R/W	0y0	LRCK Clock Polarity Selection of the Receiver 0y0: Forward 0y1: Inverted
[8]	–	–	Undefined	Read undefined. Write as zero.
[7]	LRCKOPOL	R/W	0y0	LRCK Clock Polarity Selection of the Transmitter 0y0: Forward 0y1: Inverted
[6:3]	–	–	Undefined	Read undefined. Write as zero.
[2]	–	–	Undefined	Read undefined. Write as zero.
[1]	–	–	Undefined	Read undefined. Write as zero.
[0]	–	–	Undefined	Read undefined. Write as zero.

(Explanation)

a. <LRCKOSEL>

This bit selects the LRCK clock of the I2S Transmitter. The default value after resetting selects the clock internally generated by CG.

b. <BCKOSEL>

This bit selects the BCK clock of the I2S Transmitter. The default value after resetting selects the clock internally generated by CG.

c. <LRCKIPOL>

This bit selects the LRCK clock polarity of the I2S Receiver. The default value after resetting selects forward.

d. <LRCKOPOL>

This bit selects the LRCK clock polarity of the I2S Receiver. The default value after resetting selects forward.

3.20.3 I2S Block of the Input (Receive) System

The I2S block inputs the serially input digital data (SDI) in synchronization with the rising edge of the Bit Clock (BCK) signal and selects the L-channel data or the R-channel data by the Word Select (LRCK) signal to output the data as 24-bit long parallel signals to the FIFO block. Note that the BCK signal and the LRCK signal are only input to the I2S block. The master/slave configuration should be supported by the external timing generator block.

Any of the data lengths of 16, 18, 20, and 24 bits can be used. The data length can be selected by setting the ILNG (Input Data Length) bits of the I2S_IN_CONT register (I2S Input Control Register).

Any of the numbers of slots of 32, 48, and 64 can be used. The number of slots can be selected by setting the ISLT (Input Slot) bits of the I2S_IN_CONT register (I2S Input Control Register).

Any of front-aligned, rear-aligned, and I2S can be used as the input mode. The input mode can be selected by setting the IFMT (Input Format) bits of the I2S_IN_CONT register (I2S Input Control Register). The I2S mode conforms to the I2S bus standard.

The relationships among the number of slots, the data length, and the input mode are shown in Table 3.20-1.

If the data length is less than 24 bits, the idle bits on the LSB side are reset to 0 to output 24-bit parallel data.

Table 3.20-1 Conditions for the Number of Slots, Data Length, and Input Mode

Number of slots	Data length	Input mode
64 slots	24 bits	Front-aligned
		Rear-aligned
		I2S
	20 bits	Front-aligned
		Rear-aligned
		I2S
	18 bits	Front-aligned
		Rear-aligned
		I2S
	16 bits	Front-aligned
		Rear-aligned
		I2S
48 slots	24 bits	Front-aligned (= rear-aligned)
		I2S
	20 bits	Front-aligned
		Rear-aligned
		I2S
	18 bits	Front-aligned
		Rear-aligned
		I2S
	16 bits	Front-aligned
		Rear-aligned
		I2S
32 slots	16 bits	Front-aligned (= rear-aligned)
		I2S

3.20.4 FIFO Block of the Input (Receive) System

The data input according to the input format is converted into 24-bit parallel data per channel in the I2S block for output to the FIFO block.

The FIFO block contains 2 banks (banks A and B) of 24-bit x 8-word FIFOs, each of which can store 4 fs of data. The FIFO block receives data of 24-bit words input from the I2S block and is accessed through the I2S_IN_FIFOData register. Figure 3.20-5 shows the block diagram of the FIFO block. Note that the FIFO block allows only read accesses on a word basis.

In the FIFO block, banks A and B are switched automatically depending on the count of accesses from the I2S block. To make sure of the bank being accessed from the I2S block (access side), the Access Side signal is output.

With the Access Side signal passed to the DMA I/F block, an interrupt signal is generated each time the access side switches. This interrupt signal becomes the Hardware Request signal to the external DMAC to enable the FIFO to output data through the I2S_IN_FIFOData register. For details of the signals such as Hardware Request, refer to Section 3.20.10 "DMA Interface."

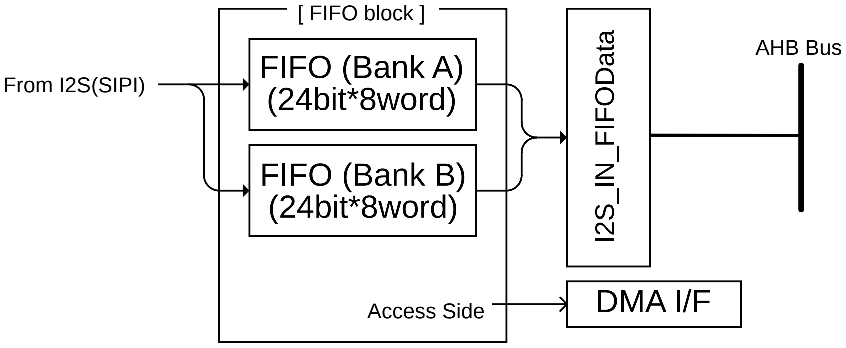


Figure 3.20-5 Block Diagram of the FIFO Block

Writing the received data to the FIFO block is performed in such a way that the L-channel data and the R-channel data is written alternately. When the FIFO block is read, therefore, the L-channel data and the R-channel data is output alternately. Figure 3.20-6 shows a rough input timing chart and the data transition in the FIFO block.

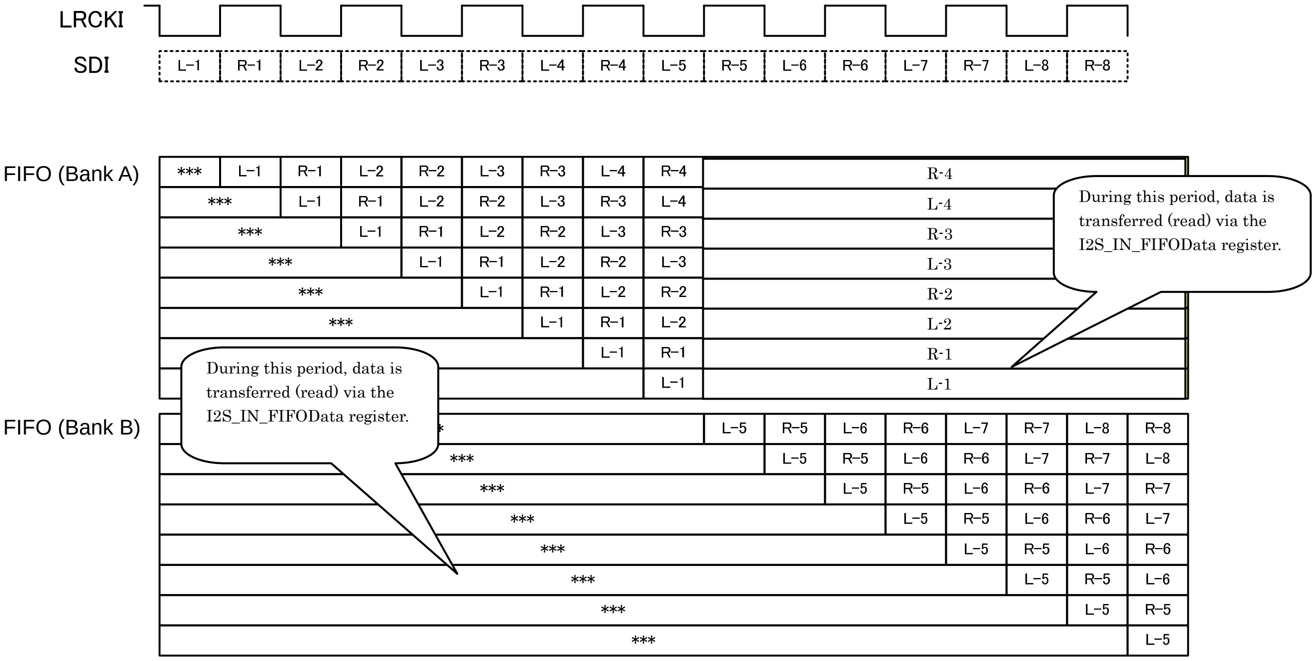

LRCKI
SDI
L-1 R-1 L-2 R-2 L-3 R-3 L-4 R-4 L-5 R-5 L-6 R-6 L-7 R-7 L-8 R-8
FIFO (Bank A)
*** L-1 R-1 L-2 R-2 L-3 R-3 L-4 R-4 R-4
*** L-1 R-1 L-2 R-2 L-3 R-3 L-4 L-4
*** L-1 R-1 L-2 R-2 L-3 R-3 R-3
*** L-1 R-1 L-2 R-2 L-3 L-3
*** L-1 R-1 L-2 R-2 R-2
*** L-1 R-1 L-2 L-2
L-1 R-1 R-1
L-1 L-1
During this period, data is transferred (read) via the I2S_IN_FIFOData register.
FIFO (Bank B)
*** L-5 R-5 L-6 R-6 L-7 R-7 L-8 R-8
*** L-5 R-5 L-6 R-6 L-7 R-7 L-8
*** L-5 R-5 L-6 R-6 L-7 R-7
*** L-5 R-5 L-6 R-6 L-7
*** L-5 R-5 L-6 R-6
*** L-5 R-5 L-6
*** L-5 R-5
*** L-5
During this period, data is transferred (read) via the I2S_IN_FIFOData register.

Figure 3.20-6 Rough Input Timing Chart and Data Transition in the FIFO Block

Besides, the FIFO block outputs the information shown below. These information items can be checks by accessing the I2S_IN_FIFOState register (I2S Input FIFO State Register).

– FIFO Pointer of BUS/I2S (IPOINT_BUS/IPOINT_I2S)

– FIFO Access Side (IFIFOBANK)

– FIFO Underflow (IFIFOUDR)

– FIFO Overflow (IFIFOOVR)

– FIFO Error (IFIFOERR)

– FIFO Clear (IFIFOCLR)

– FIFO Interrupt Enable (IFIFOINT)

The following is a description of each information item:

– FIFO Pointer (I2S/BUS) information –

This indicates the information about the pointer to the FIFO accessed by each of the I2S block and the bus interface block.

The information about the pointers accessible to individual FIFOs is displayed immediately after Operation Enable is set in each system. The pointer indicates 0x00 while it is being reset and is incremented at each access to the FIFO block.

– FIFO Access Side information –

This indicates the information about the FIFO block side (bank A or B) accessed by the I2S block.

– FIFO Underflow information –

This information indicates that more than the required amount of data (8 words) is read from the bus before the FIFO bank (access side) is switched from the I2S block.

If this information is set, the status is kept until it is cleared by the Clear bit or another appropriate action is performed.

– FIFO Overflow information –

This information indicates that the required amount of data (8 words) has not yet been read from the bus when the FIFO bank (access side) is switched from the I2S block.

If this information is set, the status is kept until it is cleared by the Clear bit or other appropriate action is performed.

– FIFO Error information –

This information indicates that either Underflow or Overflow is set. If this information is set, the status is kept until it is cleared by the Clear bit or other appropriate action is performed.

– FIFO Clear –

This function can reset (clear) the Underflow, Overflow, and Error information when it is set.

– FIFO Interrupt Enable –

This function outputs an interrupt signal the moment when Error is set. The function can be enabled or disabled.

If Operation Enable (IOPEN) is disabled, the Pointer, Access Side, Underflow, Overflow, and Error information is cleared.

3.20.5 FIFO Block of the Input (Receive) System

The data in the FIFO block is accessed through the I2S_IN_FIFOData register (refer to Figure 3.20-5 "Block Diagram of the FIFO Block."

When the FIFO data is read from the I2S_IN_FIFOData register, the L-channel data and the R-channel data is output alternately.

The I2S_IN_FIFOData register allows only read access on a word basis. Operation is undefined if byte read access, half-word read access, or write access is made to this register.

The FIFO data can be read from the I2S_IN_FIFOData register in either of the two transfer modes shown in Table 3.20-2.

Table 3.20-2 Transfer Mode

Transfer mode	Description
Normal transfer mode	The data of each channel is output as a 32-bit word (MSB-aligned).
16-bit data transfer mode	If the data of each channel is 16 bits long, the L-channel data is set to the lower 16 bits and the R-channel data to the upper 16 bits for output as a 32-bit word.

The transfer mode can be selected by setting the ITMODE (16-Bit Data Transfer Mode) bit of the I2S_IN_CONT register (I2S Input Control Register).

Figure 3.20-7 and Figure 3.20-8 show data alignment from the FIFO block data output (24 bits) to the I2S_IN_FIFOData register (32 bits) in each transfer mode. The data is MSB-aligned and the lower idle bits are reset to 0.

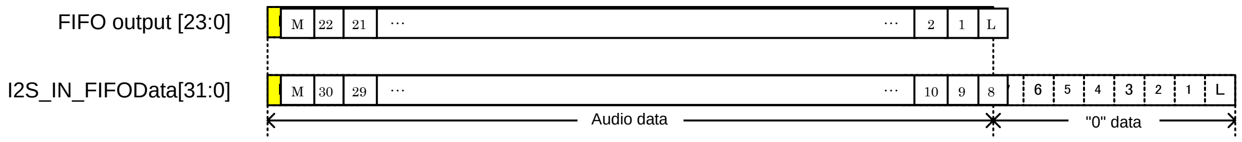


Figure 3.20-7 Data Alignment from the FIFO Block to the I2S_IN_FIFOData Register (Normal Transfer Mode)

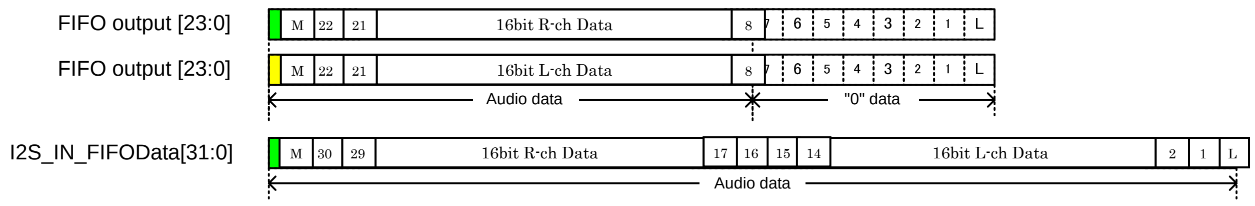

FIFO output [23:0]
M 22 21 16bit R-ch Data 8 7 6 5 4 3 2 1 L
FIFO output [23:0]
M 22 21 16bit L-ch Data 8 7 6 5 4 3 2 1 L
Audio data
"0" data
I2S_IN_FIFOData[31:0]
M 30 29 16bit R-ch Data 17 16 15 14 16bit L-ch Data 2 1 L
Audio data

Figure 3.20-8 Data Alignment from the FIFO Block to the I2S_IN_FIFOData Register (16-Bit Transfer Mode)

3.20.6 I2S Block of the Output (Transmit) System

The I2S block outputs the 24-bit FIFO audio data in synchronization with the trailing edge of the Bit Clock (BCKO) signal and selects the L-channel data or the R-channel data by the Word Select (LRCKO) signal to output serial digital data from the external pin (SDO). Note that the BCKO signal and the LRCKO signal are only input to the I2S block. The master/slave configuration should be supported by the external timing generator block.

Any of the data lengths of 24, 20, 18, and 16 bits can be used. The data length can be selected by setting the OLNG (Output Data Length) bits of the I2S_OUT_CONT register (I2S Output Control Register).

Any of the numbers of slots of 32, 48, and 64 can be used. The number of slots can be selected by setting the OSLT (Output Slot) bits of the I2S_OUT_CONT register (I2S Output Control Register).

Any of front-aligned, rear-aligned, and I2S can be used as the output mode. The output mode can be selected by setting the OFMT (Output Format) bits of the I2S_OUT_CONT register (I2S Output Control Register). The I2S mode conforms to the I2S bus standards.

The relationships among the number of slots, the data length, and the input mode are shown in Table 3.20-3.

If the data length is less than 24 bits, the LSB side is padded with zeros to output 24-bit data.

Table 3.20-3 Conditions for Setting the Number of Slots, Data Length, and Output Mode

Number of slots	Data length	Output mode
64 slots	24 bits	Front-aligned
		Rear-aligned
		I2S
	20 bits	Front-aligned
		Rear-aligned
		I2S
	18 bits	Front-aligned
		Rear-aligned
		I2S
	16 bits	Front-aligned
		Rear-aligned
		I2S
48 slots	24 bits	Front-aligned (= rear-aligned)
		I2S
	20 bits	Front-aligned
		Rear-aligned
		I2S
	18 bits	Front-aligned
		Rear-aligned
		I2S
	16 bits	Front-aligned
		Rear-aligned
		I2S
32 slots	16 bits	Front-aligned (= rear-aligned)
		I2S

3.20.7 FIFO Block of the Output (Transmit) System

Parallel data stored in the FIFO block is converted to serial digital data for output to the I2S block.

The FIFO block contains 2 banks (banks A and B) of 24-bit x 8-word FIFOs, each of which can store 4 fs of data. The FIFO block receives the data input through the I2S_OUT_FIFOData register and outputs data of 24-bit words to the I2S (Figure 3.20-9). Note that the I2S_OUT_FIFOData register allows only write accesses on a word basis.

In the FIFO block, banks A and B are switched automatically depending on the count of accesses from the I2S block. To make sure of the bank being accessed from the I2S block (access side), the Access Side signal is output.

With the Access Side signal passed to the DMA I/F block, an interrupt signal is generated each time the access side switches. This interrupt signal becomes the Hardware Request signal to the external DMAC to enable the FIFO to receive input data through the I2S_OUT_FIFOData register. For details of the signals such as Hardware Request, refer to Section 3.20.10, "DMA Interface."

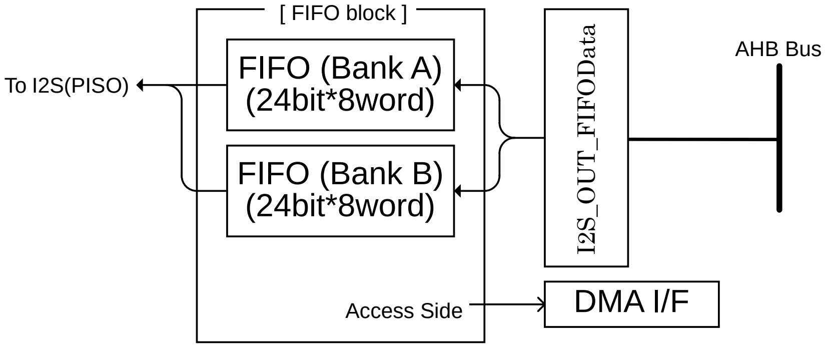


Figure 3.20-9 Block Diagram of the FIFO Block

When data is sent from the FIFO block, the L-channel data and the R-channel data is read alternately. This means that when data is written to the I2S_OUT_FIFOData register (for preparation of the data to be sent to the FIFO block), the order in which the data is read in FIFO block must be taken into consideration. Figure 3.20-10 shows a rough timing chart of writing to the I2S_OUT_FIFOData register and the data transition in the FIFO block.

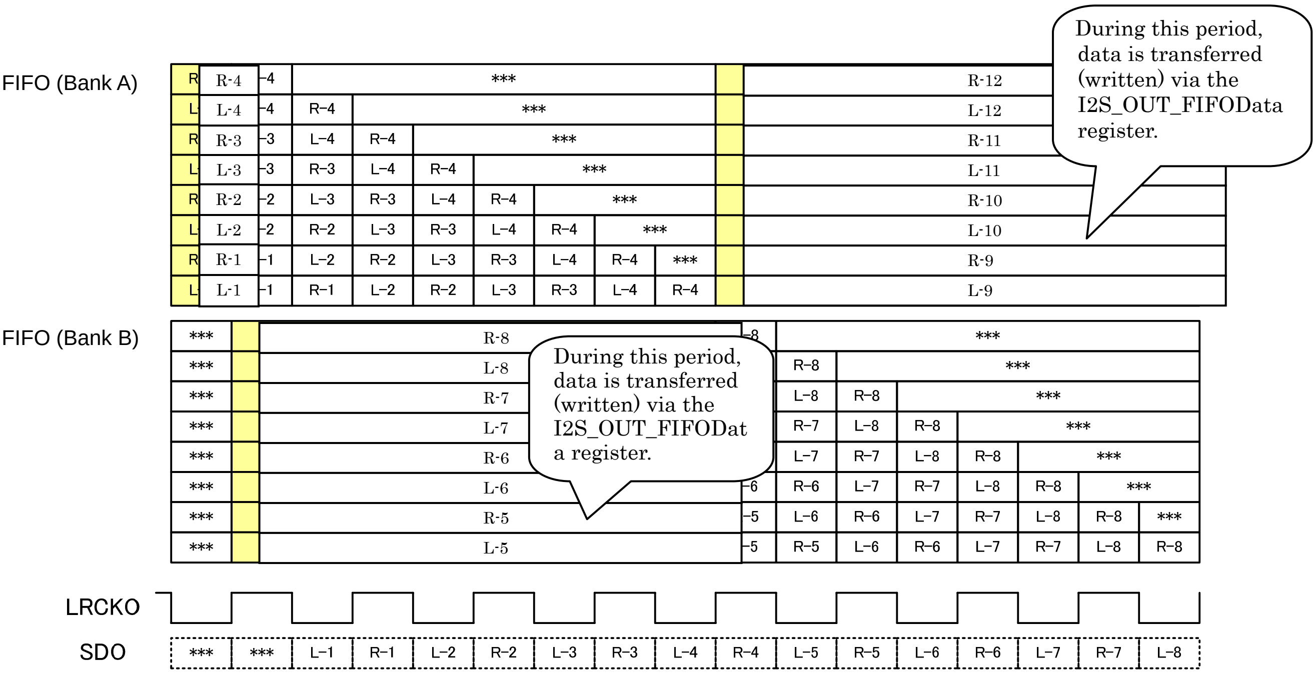

FIFO (Bank A)
FIFO (Bank B)
LRCKO
SDO
R-4
L-4
R-3
L-3
R-2
L-2
R-1
L-1
R-12
L-12
R-11
L-11
R-10
L-10
R-9
L-9
R-8
L-8
R-7
L-7
R-6
L-6
R-5
L-5
***
During this period, data is transferred (written) via the I2S_OUT_FIFOData register.
During this period, data is transferred (written) via the I2S_OUT_FIFOData register.
*** *** L-1 R-1 L-2 R-2 L-3 R-3 L-4 R-4 L-5 R-5 L-6 R-6 L-7 R-7 L-8

Figure 3.20-10 Rough Output Timing Chart of the FIFO Block of the Transmitter and Data Transition in the FIFO Block

The FIFO block outputs the information shown below to the I2S_OUT_FIFOState register (I2S Output FIFO State Register). These information items can be checked by accessing the *I2S_OUT_FIFOState* register.

– FIFO Pointer of I2S/BUS (OPOINT_I2S/OPOINT_BUS)

– FIFO Access Side (OFIFOBANK)

– FIFO Underflow (OFIFOUDR)

– FIFO Overflow (OFIFOOVR)

– FIFO Error (OFIFOERR)

– FIFO Clear (OFIFOCLR)

– FIFO Interrupt Enable (OFIFOINT)

The following is a description of each information item:

– FIFO Pointer information –

This indicates the information about the pointer to the FIFO accessed by each of the I2S block and the bus interface block.

The information about the pointers accessible to individual FIFOs is displayed immediately after Operation Enable is set in each system. The pointer indicates 0x00 while it is being reset and is incremented immediately after access to the FIFO block.

– FIFO Access Side information –

This indicates the information about the FIFO block side (bank A or B) accessed by the I2S block.

– FIFO Underflow information –

This information indicates that the required amount of data (8 words) has not yet been written from the bus when the FIFO bank (access side) is switched from the I2S block.

If this information is set, the status is kept until it is cleared by the Clear bit or another appropriate action is performed.

– FIFO Overflow information –

This information indicates that more than the required amount of data (8 words) is written from the bus before the FIFO bank (access side) is switched from the I2S block.

If this information is set, the status is kept until it is cleared by the Clear bit or other appropriate action is performed.

– FIFO Error information –

This information indicates that either Underflow or Overflow is set. If this information is set, the status is kept until it is cleared by the Clear bit or other appropriate action is performed.

– FIFO Clear –

This function can reset (clear) the Underflow, Overflow, and Error information when it is set.

– FIFO Interrupt Enable –

This function outputs an interrupt signal the moment when Error is set. The function can be enabled or disabled.

If Operation Enable (OOPEN) is disabled, the Pointer, Access Side, Underflow, Overflow, and Error information is cleared.

3.20.8 FIFO Block of the Output (Transmit) System

Data is input to the FIFO block through the I2S_OUT_FIFOData register.

The I2S_OUT_FIFOData register allows only write access on a word basis. Operation is undefined if byte write access, half-word write access, or read access is made to this register.

For input to the I2S_OUT_FIFOData register, the L-channel data and the R-channel data must be input alternately.

For preparation of the data to be sent to the FIFO block, 32-bit data can be input to the I2S_OUT_FIFOData register in either of the following two transfer modes:

Table 3.20-4 Transfer Mode

Transfer mode	Description
Normal transfer mode	The data of each channel is written as a 32-bit word (MSB-aligned).
16-bit data transfer mode	If the data of each channel is 16 bits long, the L-channel data is set to the lower 16 bits and the R-channel data to the upper 16 bits for input as a 32-bit word.

The transfer mode can be selected by setting the OTMODE (16-Bit Data Transfer Mode) bit of the I2S_OUT_CONT register (I2S Output Control Register).

Figure 3.20-11 and Figure 3.20-12 show data alignment from the I2S_OUT_FIFOData register (32 bits) to the transmit FIFO block data (24 bits) in each transfer mode. The data must be MSB-aligned with 0 written to lower idle bits.

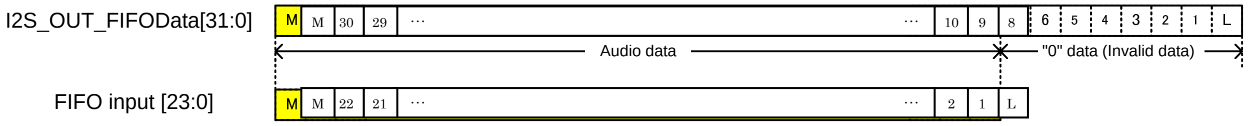


Figure 3.20-11 Data Alignment from the I2S_OUT_FIFOData Register to the Transmit FIFO (Normal Transfer Mode)

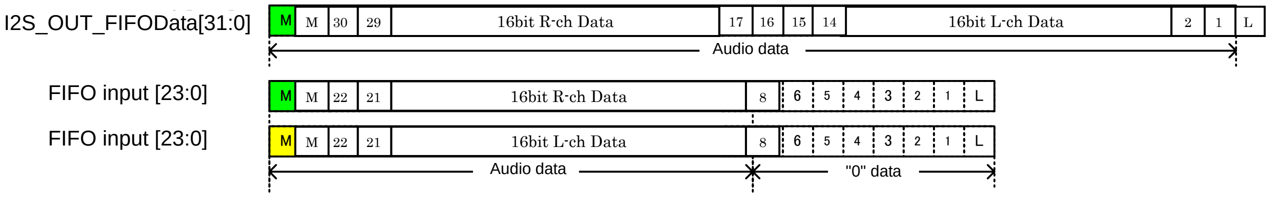


Figure 3.20-12 Data Alignment from the I2S_OUT_FIFOData Register to the Transmit FIFO (16-Bit Transfer Mode)

3.20.9 Digital Audio I/O Format

The input mode for digital audio data to be input/output can be selected from among front-aligned, rear-aligned, and I2S.The I2S mode conforms to the I2S standards.

The number of slots for digital audio data to be input/output can be selected from among 32, 48, and 64.

The data length of digital audio data to be input/output can be selected from among 16, 18, 20, and 24 bits. If a data length of less than 24 bits is set, the lower-bit data must be masked.

Figure 3.20-12 and Figure 3.20-14 shows I/O formats. (Important: The polarity of the LRCKI and LRCKO signals is determined by the timing specific to the specifications of this Module (always L-channel data when LRCK = Low).)

Table 3.20-5 Conditions for Setting the Number of Slots, Data Length, and I/O Mode

Number of slots	Data length	I/O mode
64 slots	24 bits	Front-aligned
		Rear-aligned
		I2S
	20 bits	Front-aligned
		Rear-aligned
		I2S
	18 bits	Front-aligned
		Rear-aligned
		I2S
	16 bits	Front-aligned
		Rear-aligned
		I2S
48 slots	24 bits	Front-aligned (= rear-aligned)
		I2S
	20 bits	Front-aligned
		Rear-aligned
		I2S
	18 bits	Front-aligned
		Rear-aligned
		I2S
	16 bits	Front-aligned
		Rear-aligned
		I2S
32 slots	16 bits	Front-aligned (= rear-aligned)
		I2S

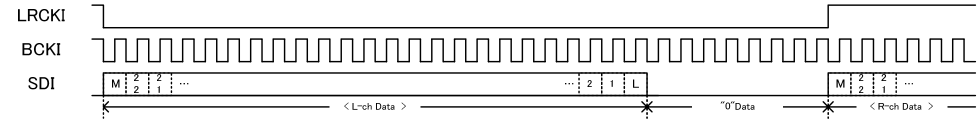
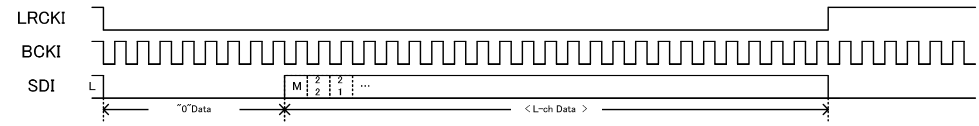
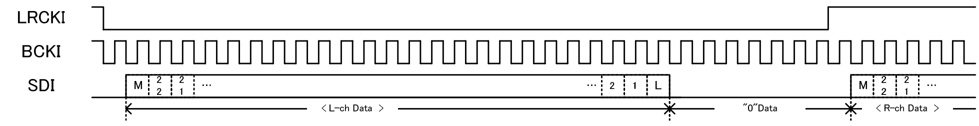
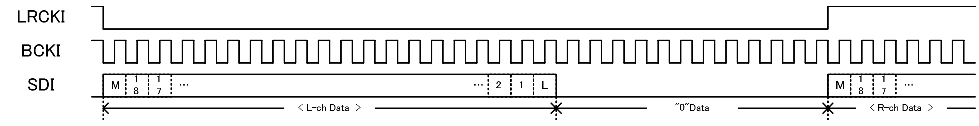
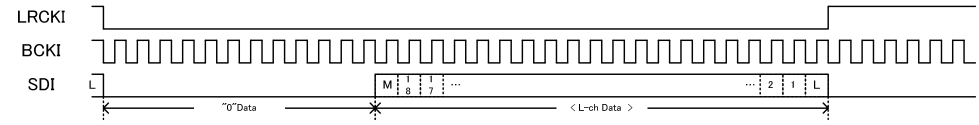
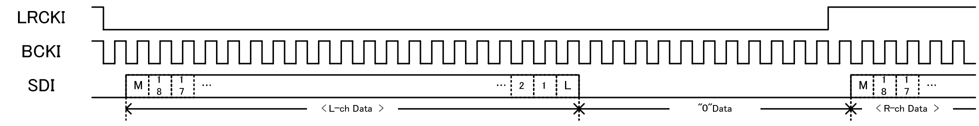
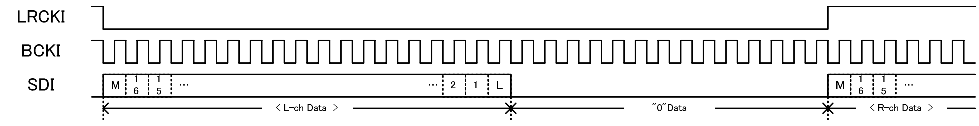
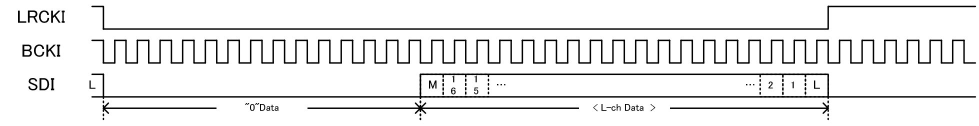
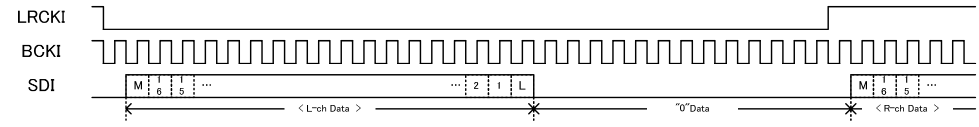
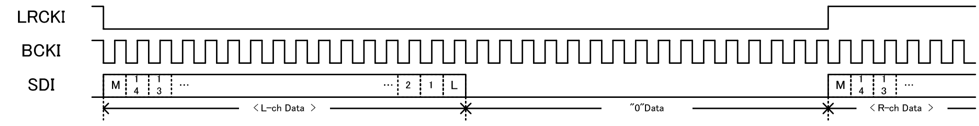
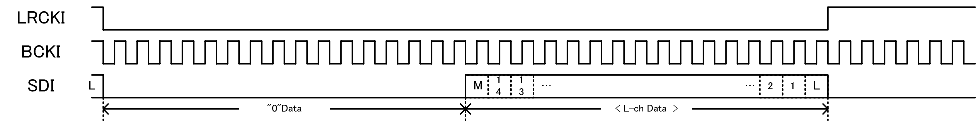
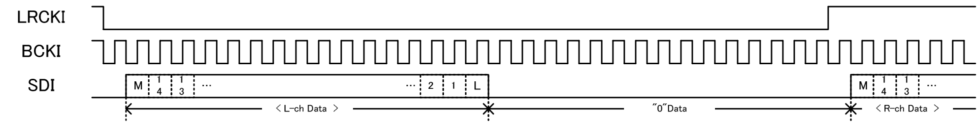
Number of slots	Number of bits	Format	I/O format (Note: "M" stands for MSB and "L" stands for LSB in the figure.)
64	24	Front-aligned	 LRCKI BCKI SDI M 22 21 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 22 21 … "0"Data < L-ch Data >
		I2S	 LRCKI BCKI SDI M 22 21 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
	20	Front-aligned	 LRCKI BCKI SDI M 18 17 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 18 17 … … 2 1 L "0"Data < L-ch Data >
		I2S	 LRCKI BCKI SDI M 18 17 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
	18	Front-aligned	 LRCKI BCKI SDI M 16 15 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 16 15 … … 2 1 L "0"Data < L-ch Data >
		I2S	 LRCKI BCKI SDI M 16 15 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
	16	Front-aligned	 LRCKI BCKI SDI M 14 13 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 14 13 … … 2 1 L "0"Data < L-ch Data >
		I2S	 LRCKI BCKI SDI M 14 13 … … 2 1 L < L-ch Data > "0"Data < R-ch Data >

Figure 3.20-13 I/O Format (1)

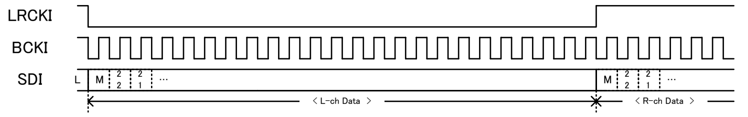
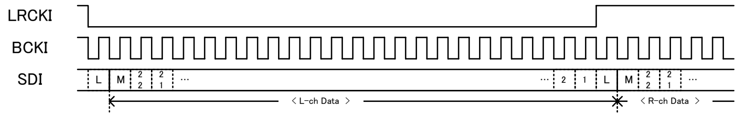
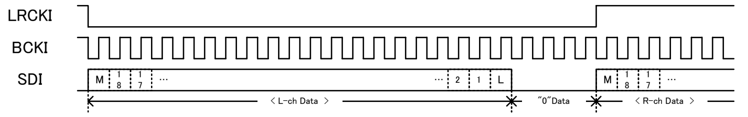
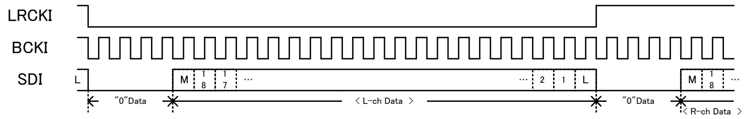
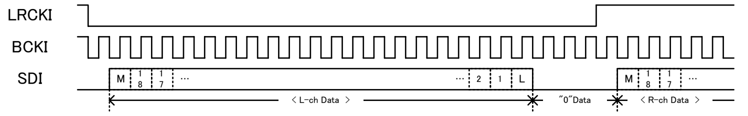
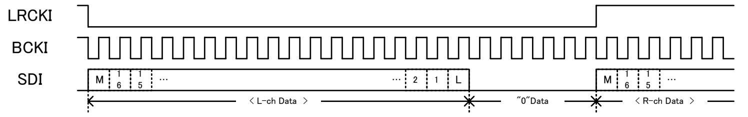
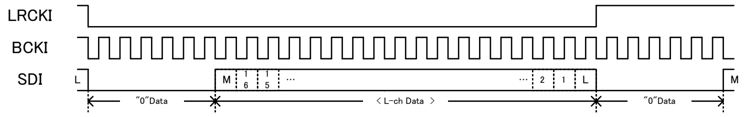
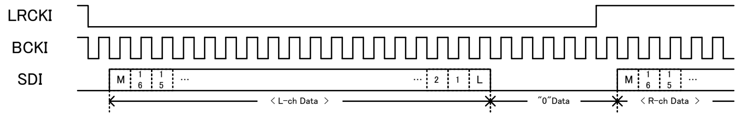
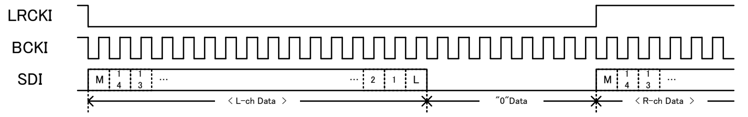
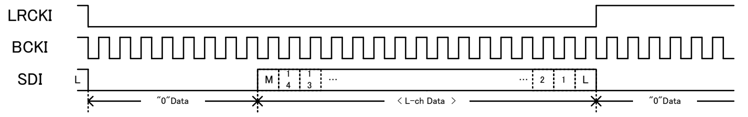
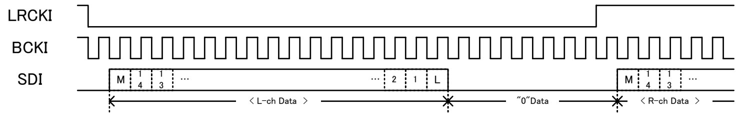
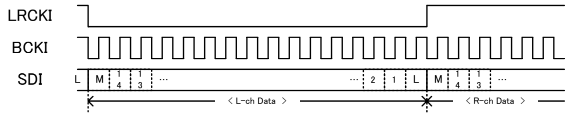
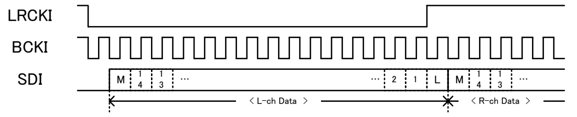
Number of slots	Number of bits	Format	I/O format (Note: "M" stands for MSB and "L" stands for LSB in the figure.)
48	24	Front-aligned (= rear-aligned)	 LRCKI BCKI SDI L M 22 21 … M 22 21 … < L-ch Data > < R-ch Data >
		I2S	 LRCKI BCKI SDI L M 22 21 … … 2 1 L M 22 21 … < L-ch Data > < R-ch Data >
	20	Front-aligned	 LRCKI BCKI SDI M 18 17 … … 2 1 L M 18 17 … < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 18 17 … … 2 1 L M 18 … "0"Data < L-ch Data > "0"Data < R-ch Data >
		I2S	 LRCKI BCKI SDI M 18 17 … … 2 1 L M 18 17 … < L-ch Data > "0"Data < R-ch Data >
	18	Front-aligned	 LRCKI BCKI SDI M 16 15 … … 2 1 L M 16 15 … < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 16 15 … … 2 1 L M "0"Data < L-ch Data > "0"Data
		I2S	 LRCKI BCKI SDI M 16 15 … … 2 1 L M 16 15 … < L-ch Data > "0"Data < R-ch Data >
	16	Front-aligned	 LRCKI BCKI SDI M 14 13 … … 2 1 L M 14 13 … < L-ch Data > "0"Data < R-ch Data >
		Rear-aligned	 LRCKI BCKI SDI L M 14 13 … … 2 1 L "0"Data < L-ch Data > "0"Data
		I2S	 LRCKI BCKI SDI M 14 13 … … 2 1 L M 14 13 … < L-ch Data > "0"Data < R-ch Data >
32	16	Front-aligned (= rear-aligned)	 LRCKI BCKI SDI L M 14 13 … … 2 1 L M 14 13 … < L-ch Data > < R-ch Data >
		I2S	 LRCKI BCKI SDI M 14 13 … … 2 1 L M 14 13 … < L-ch Data > < R-ch Data >

Figure 3.20-14 I/O Format (2)

3.20.10 DMA Interface

This section describes the basic operation of the DMA interface.

The DMA interface provides the Hardware Request (HReq) signal and the Clear (Ack) signal used to de-assert (clear) the Hardware Request signal. The status of the Hardware Request signal can be monitored on the Hardware Request bit in the Control Register.

The Hardware Request signal is asserted each time the access side of the FIFO block accessed by the I2S block switches. The asserted Hardware Request signal is de-asserted by the input of Ack.

Figure 3.20-14 shows how the Hardware Request signal is asserted and de-asserted.

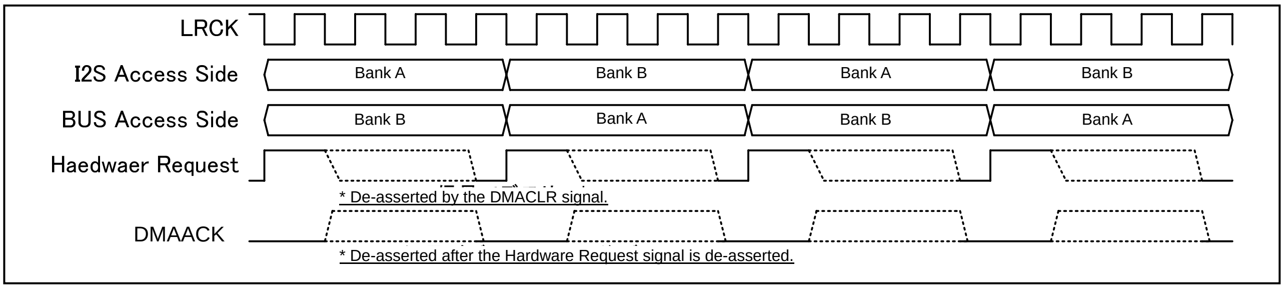
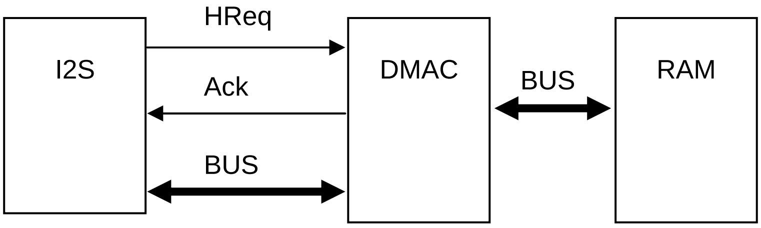


Figure 3.20-15 Assert/De-assert Timing Chart of the Hardware Request Signal

Table 3.20-6 I/O Signals for DMAC Connection

Signal name	I/O	Description/Use
RxHReq/TxHReq	O	This Request signal outputs the High level each time the access side of the FIFO block switches. It is used as the Request signal for a hardware request to the DMAC.
RxAck/TxAck	I	This is the input signal used to clear the Request signal. The Request signal can be cleared by connecting this signal to the Acknowledge signal from the DMAC.



DMA transfer images:

1. I2S Enable
2. FIFO (empty)
3. HReq = 1 from I2S
4. The DMAC transfers the RAM data to I2S.
5. Eight contiguous words have been transferred.
6. An Ack is returned from the DMAC.
7. HReq = 0 from I2S

The asserted Hardware Request signal can be forcibly de-asserted by writing "0" on the Hardware Request bit in the Control Register.

The priority order among the CPU action, FIFO action, and the Ack signal from the DMAC (DMAACK) is as follows when they compete:

(Writing "0" or "1" by the CPU) > (De-asserting the Hardware Request signal by the DMAACK) > (Asserting the Hardware Request signal by FIFO bank switching)

To use this DMA interface, Operation Enable (IOPEN and OOPEN) of the Receiver and the Transmitter must be set to Enable. The Hardware Request signal is cleared if Operation Enable is set to Disable.

Figure 3.20-1 is a connection diagram with the DMAC. It is assumed that the DMAC can receive two or more Hardware Request signals.

In DMA transfer, access to the FIFO must be made in units of 8 words. This requires the DAMC burst size to be set to 8 beats and the total transfer account be set to a multiple of 8. Otherwise, correct DMA transfer by switching between FIFO banks A and B is impossible.

Figure 3.20-16 shows a timing chart of the DMA transfer with the burst size set to 8 beats.

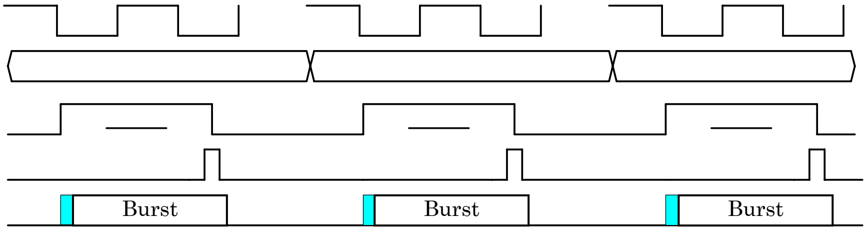


Figure 3.20-16 DMA Transfer Timing Chart When the Burst Size is 8 Beats

An example of DMA setting for I2S is shown below. For details of the control registers of the DMA, refer to Section 3.8, "DMAC."

```
// Tx Side (Transmitter)
// DMAC CH0 transfer condition Set
DMACC0SrcAddr        = 0x01034000 ; // DMAC Src Address (Memory)
DMACC0DestAddr       = 0x40016300 ; // DMAC Dst Address (I2S_OUT_FIFOData: Address is fixed)
DMACC0LLI            = 0x00000000 ; // LLI  (not USE LLI)
DMACC0Control        = 0x04492040 ; // DMAC Control & INT enable (Transfer Size =  a multiple of 64 *8)
DMACC0Configuration = 0x0000c801 ; // Kick off, TC no-mask and Mem to Peri(I2S)

// Rx Side (Receiver)
// DMAC CH1 transfer condition Set
DMACC1SrcAddr        = 0x40016100 ; // DMAC Src Address (I2S_IN_FIFOData: Address is fixed)
DMACC1DestAddr       = 0x01035000 ; // DMAC Dst Address (Memory)
DMACC1LLI            = 0x00000000 ; // LLI  (not USE LLI)
DMACC1Control        = 0x08492040 ; // DMAC Control (Transfer Size =  a multiple of 64 *8), INT disable
DMACC1Configuration = 0x0000d003 ; // Kick off, TC no-mask and Peri(I2S) to Mem
```

3.20.11 About Restrictions of Transfer Timing by the Hardware Request Signal

Some restrictions are imposed on the FIFO access sections (time) by the Hardware Request signal.

Figure 3.20-17 and Figure 3.20-18 show the timing charts of the data transfer sections.

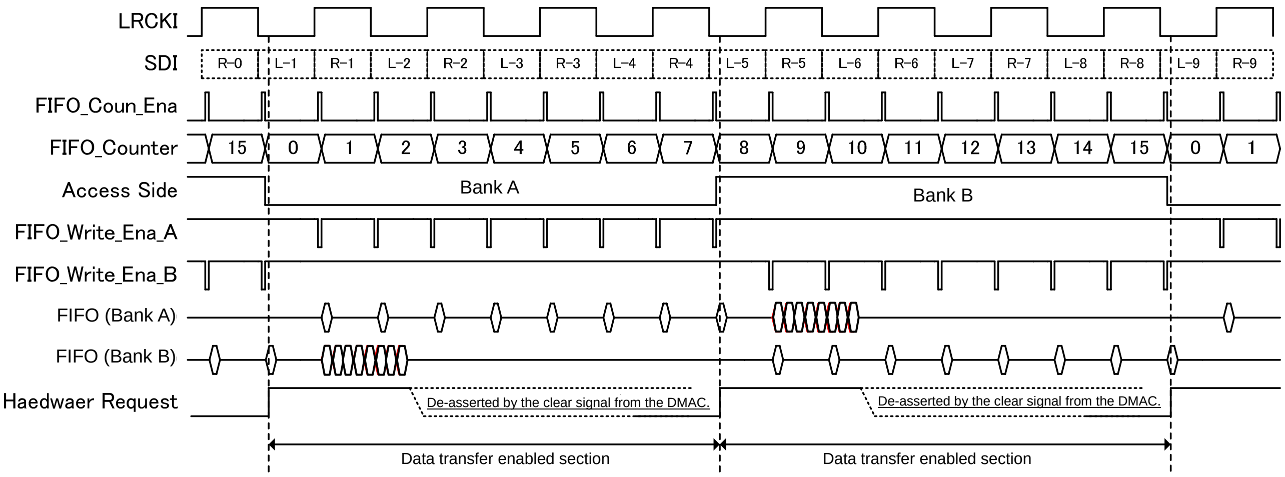


Figure 3.20-17 Data Transfer Enabled Section (Receiver)

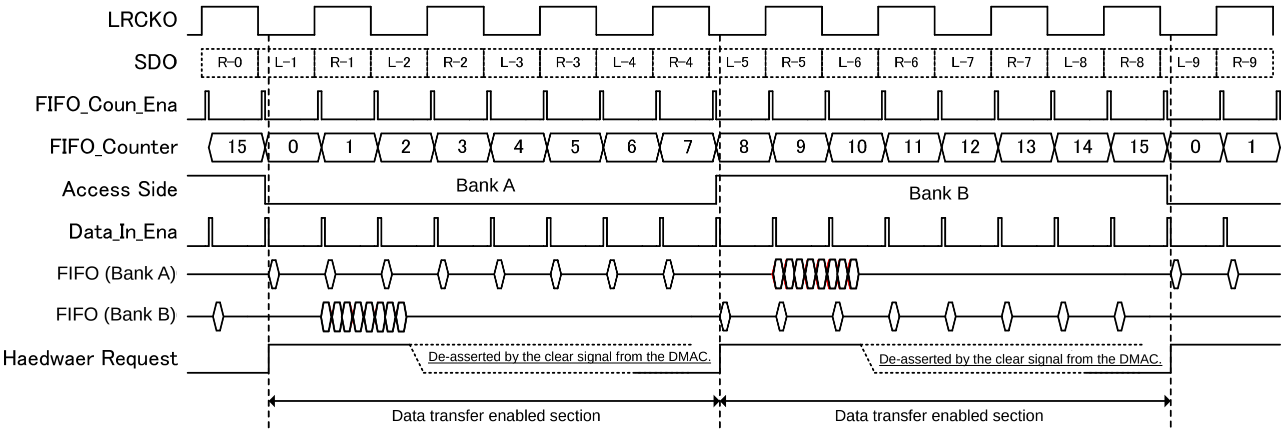


Figure 3.20-18 Data Transfer Enabled Section (Transmitter)

The data transfer enabled section is 4 LRCK cycles after the Hardware Request signal is asserted. The following table shows the data transfer enabled time for each fs:

Table 3.20-7 Clock Frequencies in Ordinary Sampling Frequencies

Sampling frequency (fs)	Data transfer enabled time [μs]
32kHz	125
44.1kHz	90.702
48kHz	83.333

If transfer fails to finish within the period indicated in the table, the error bit in the FIFO Stats is set on access side switching.

3.20.12 About the Error Interrupt Signal

This I2S interface provides an error interrupt output.

The error interrupt signal is asserted when an error occurs in the FIFO. Since this signal is connected to the external interrupt controller, an interrupt occurs whenever an error occurs in the FIFO.

The error interrupt signal is ORed with the error signals of each FIFO. To output this signal, set the FIFO Interrupt Enable bit of the Control Register to Enable.

Figure 3.20-19 shows the logical circuit of error interrupt output.

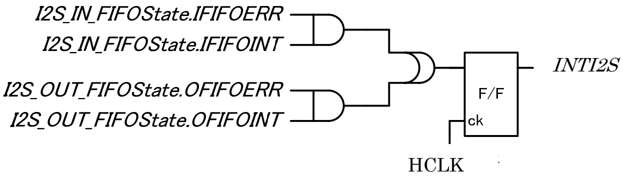

I2S_IN_FIFOState.IFIFOERR
I2S_IN_FIFOState.IFIFOINT
I2S_OUT_FIFOState.OFIFOERR
I2S_OUT_FIFOState.OFIFOINT
F/F
ck
INTI2S
HCLK

Figure 3.20-19 Error Interrupt Output

3.20.13 About Frequency Generation Errors (LRCK)

The I2S can select clocks internally generated by the CG or external input clocks as the source of the transmitter clocks (LRCK, BCK, and MCLK). Use the I2S_CG_CNT register (I2S Clock Generator Control Register) for the setting of the source. Clear both LRCKOSEL and BCKOSEL to 0 to select the internally generated clocks.

Important: Select the same clock source for LRCK and BCK. If different sources are selected, synchronization will be lost between LRCK and BCK, causing erroneous operations.

Next, set the CG_I2SFSCTRL register to set the internally generated clock. For details of the CG_I2SFSCTRL register, refer to Section 3.5.3, "CG_I2SFSCTRL Register" in Chapter 3.5, "Clock Controller."

The method of calculating the counter set value to the CG_I2SFSCTRL register is as follows:

Example) When the I2S is operating with fHCLK =144 MHz, FS = 44.1 kHz, the number of slots = 64, and MCLK not used:

Consider the High/Low width of FS (fLRCK) as the basis for the counter set value.

Therefore, first determine the count required for 1/2 FS = 88.2 kHz.

Period of 1/2 FS (1/88.2 [kHz]) / Period of HCLK (1/144 [MHz]) = 1632.653….

Round the integer part for smaller error and get a count value of 1633 (1/2 FS).

Then determine the value to set for BCK (when MCLK is not used).

Since the number of slots (SLOT) is 64, BCK is 32 clock cycles at 1/2 FS. Thus, the count for 1 BCK period is:

1633 (count)/ 32 = 51.03125 count

Since the counter counts the High/Low width of the BCK, round the number to an even number for smaller error.

The count required for l BCK interval is 52.

The count required for 1/2 period of BCK is calculated as follows:

52 / 2 = 26 = 0x1A

Based on this calculation, the value set to the CG_I2SFSCTRL register is determined as follows:

0x1A - 1 = 0x19

At this time,

f_{BCK} (calculated value) = 144 [MHz] * 52 (count) = 2769230.769 [Hz]

f_{BCK} (logical value) = 144 [MHz] * 64 (SLOT) = 2822400 [Hz]

Error of f_{BCK} = 1.88%

FS = f_{LRCK} (calculated value) = 2769230.769 [Hz] * 64 (slots) = 43269.23077 [Hz]

FS = f_{LRCK} (logical value) = 44.1 [Hz]

Error of FS = 1.88%

The following table shows counter values and frequency errors for typical sampling frequencies set to the CG_I2SFSCTRL register, and frequencies generated for those values.

(Note) The errors in this table are logical values when the base clock of 144 MHz is used and the approximate setting values of the target sampling frequencies. For the count values of the frequencies to be generated, logical values minus 1 (-1) must be set as shown in the table.

Table 3.20-8 List of LRCK Frequency Errors (MCLK Not Used (3-Wire Interface), Operation at 144 MHz)

Sampling frequency [Hz]	Value set for BCK generation count (hex)			Generated frequency [Hz]			Error [%]		
	Number of slots			Number of slots			Number of slots		
	32	48	64	32	48	64	32	48	64
8000	118h	BBh	8Ch	8007.117	7978.723	7978.723	0.09%	-0.27%	-0.27%
11025	CBh	87h	65h	11029.41	11029.41	11029.41	0.04%	0.04%	0.04%
12000	BBh	7Ch	5Dh	11968.09	12000	11968.09	-0.27%	0.00%	-0.27%
16000	8Ch	5Dh	45h	15957.45	15957.45	16071.43	-0.27%	-0.27%	0.45%
22050	65h	43h	32h	22058.82	22058.82	22058.82	0.04%	0.04%	0.04%
24000	5Dh	3Eh	2Eh	23936.17	23809.52	23936.17	-0.27%	-0.79%	-0.27%
32000	45h	2Eh	22h	32142.86	31914.89	32142.86	0.45%	-0.27%	0.45%
44100	32h	21h	19h	44117.65	44117.65	43269.23	0.04%	0.04%	-1.88%
48000	2Eh	1Eh	16h	47872.34	48387.1	48913.04	-0.27%	0.81%	1.90%

Table 3.20-9 List of LRCK Frequency Errors (MCLK x4 Used (4-Wire Interface), Operation at 144 MHz)

Sampling frequency [Hz]	Value set for MCLK generation count (hex)			Generated frequency [Hz]			Error [%]		
	Number of slots			Number of slots			Number of slots		
	32	48	64	32	48	64	32	48	64
8000	45h	2Eh	22h	8035.714	7978.723	8035.714	0.45%	-0.27%	0.45%
11025	32h	21h	19h	11029.41	11029.41	10817.31	0.04%	0.04%	-1.88%
12000	2Eh	1Eh	16h	11968.09	12096.77	12228.26	-0.27%	0.81%	1.90%
16000	22h	16h	11h	16071.43	16304.35	15625	0.45%	1.90%	-2.34%
22050	19h	10h	Ch	21634.62	22058.82	21634.62	-1.88%	0.04%	-1.88%
24000	16h	Fh	Bh	24456.52	23437.5	23437.5	1.90%	-2.34%	-2.34%
32000	11h	Bh	8h	31250	31250	31250	-2.34%	-2.34%	-2.34%
44100	Ch	8h	5h	43269.23	41666.67	46875	-1.88%	-5.52%	6.29%
48000	Bh	7h	5h	46875	46875	46875	-2.34%	-2.34%	-2.34%

3.20.14 I2S Startup Sequence

This section describes the startup sequence for using this I2S interface.

1. Setting when the Receiver and the Hardware Request signal are used:

   (1) Release the system reset status.

   (2) Implement the PLL initialization sequence.

   (3) Implement I2S FS settings. (The following is an example of setting.)

   ```
   // CG I2S FS=48KHz Set using MCLK x 4
   CG_FSCTRL    = 0x00000004;
   CG_I2SFSCTRL = 0x00000005;
   ```

   (4) Implement the DMA controller settings.

   – Source Address

   – Destination Address

   – Burst length (Set a multiple of 8 beats.)

   (5) Implement each setting of the I2S_IN_CONT register.

   – Input format

   – Number of slots

   – Bit length

   – Availability of 16-bit data transfer mode

   (6) Set the Operation Enable bit to "Enable."

   Since this bit is in the I2S_IN_CONT register, it can be set together with other bits.

   (7) Implement DMA transfer on each access side (bank) switching in the FIFO block.

2. Setting when the Transmitter and the Hardware Request signal are used:

   (1) Release the system reset status.

   (2) Implement the PLL initialization sequence.

   (3) Implement I2S FS settings. (Not required if the PLL is already operating and the FS has been set.)

   (4) Implement the DMA controller settings.

   – Source Address

   – Destination Address

   – Burst length (Set a multiple of 8 beats.)

   (5) Implement each setting of the I2S_OUT_CONT register.

   – Output format

   – Number of slots

   – Bit length

   – Availability of 16-bit data transfer mode

   (6) Set the Operation Enable bit to "Enable."

   Since this bit is in the I2S_OUT_CONT register, it can be set together with other bits.

   (7) Implement DMA transfer on each access side (bank) switching in the FIFO block.

4. Electrical Characteristics

4.1 Absolute Maximum Rating

Symbol	Item	Rating	Unit
DVCC33IO	Power supply voltage	-0.3 to 3.9	V
DVCC33DRM		-0.3 to 3.9	
DVCC12 DVCC12PLL DVCC12DRM DVCC12USB		-0.3 to 2.0	
AVDD33ADC		-0.3 to 3.9	
AVDD33USBx		-0.3 to 3.9	
V_{IN}	Input voltage	-0.3 to DVCC33IO+0.3 (Note 1) -0.3 to AVDD33ADC+0.3 (Note 2) -0.3 to AVDD33USBx+0.3 (Note 2)	V
IOL	Output current (Per terminal)	2	mA
IOL2	Output current (Per terminal)	3.5	mA
IOH	Output current (Per terminal)	2	mA
Σ_{IOL}	Output current (Total)	40	mA
Σ_{IOH}	Output current (Total)	-40	mA
P_D	Power consumption (Ta = 85ºC)	700	mW
T_{SOLDER}	Soldering temperature (10 s)	260	°C
T_{STG}	Storage temperature	-65 to150	°C
T_{OPR}	Operating temperature	-40 to 85	°C

Note 1) For the USB terminals, D+ and D-, the maximum rating of AVDD33USB is applied.

Note 2) The absolute maximum rating refers to a standard that must not be exceeded even for an instant, where any one of the ratings must not be exceeded. Exceeding the absolute maximum ratings may cause breaks and deterioration as well as injuries due to ruptures and combustion. Therefore, be sure to have applied equipment designed such that the absolute maximum ratings are not exceeded.

Cautions as to solder wettability for Pb-free items (G-products)

Test item	Test condition	Remarks
Solderability	230°C: Use R-type flux at a rate of once per 5 seconds (When using Sn-37Pb PB solder). 245°C: Use R-type flux at a rate of once per 5 seconds (When using Sn-3.0Ag-0.5Cu PB-free solder).	Items with a solder adhesion of 95% up until forming are specified as good.

4.2 DC Electrical Characteristics

Operating voltage

Symbol	Item	Min	Typical	Max	Unit	Condition
DVCC33IO	General I/O Power Supply Voltage	3.0	3.3	3.6	V	XI = 12 MHz CPU CLK (-144 MHz) (DVSSCOM = DVSSPLL = AVSSADC = AVSS33USBx = DVSSUSB = 0 V)
DVCC33DRM	DRAM Power	3.0	3.3	3.6		
AVDD33ADC	ADC Power	3.0	3.3	3.6		
AVDD33USBx	USB Power for (HS and FS modes)	3.0	3.3	3.6		
DVCC12	Internal Power	1.1	1.2	1.3		
DVCC12PLL	PLL Power					
DVCC12DRM	DRAM Power					
DVCC12USB	USB Power	1.1	1.2	1.3		

There are multiple power supply terminals for each identical system, but this is predicated on the assumption that all power supply terminals for each identical system are electrically connected externally and equal voltage is supplied to all of them.

Input voltage (1)

Symbol	Item	Min	Typical	Max	Unit	Condition
VIL0	Input Low Voltage for D0-15, PA1, PA3, PA5, PA7, PC1-7, PD0-3, PD5, PD7	-0.3	−	0.3 × DVCC33IO	V	3.0 ≤ DVCC33IO ≤ 3.6 V
VIL1	Input Low Voltage for PA0, PA2, PA4, PA6, PB0-7, PC0, PD4, PD6, PE0-5, PF0-7, PG0-7		−	0.25 × DVCC33IO		
VIL2	Input Low Voltage for XI		−	0.2 × DVCC33IO		
VIL3	Input Low Voltage for /RESET			0.25 × DVCC33IO		
VIL4	Input Low Voltage for MODE0-1		−	0.3		

Input voltage (2)

Symbol	Item	Min	Typical	Max	Unit	Condition
VIH0	Input Low Voltage for D0-15, PA1, PA3, PA5, PA7, PC1-7, PD0-3, PD5, PD7	0.7 × DVCC33IO	−	DVCC33IO + 0.3	V	3.0 ≤ DVCC33IO ≤ 3.6 V
VIH1	Input Low Voltage for PA0, PA2, PA4, PA6, PB0-7, PC0, PD4, PD6, PE0-5, PF0-7, PG0-7	0.75 × DVCC33IO	−			
VIH2	Input Low Voltage for XI	0.8 × DVCC33IO	−			
VIH3	Input Low Voltage for /RESET	0.75 × DVCC33IO				
VIH4	Input Low Voltage for MODE-1	DVCC33IO - 0.3	−			

Output voltage (1)

Symbol	Item	Min	Typical	Max	Unit	Condition
VOL0	Terminals other than Output Low Voltage for VOL1	−	−	0.4	V	IOL = 2.0 mA 3.0 ≤ DVCC33IO ≤ 3.6 V
VOL1	Output Low Voltage for PB2-7					IOL = 3.0 mA 3.0 ≤ DVCC33IO ≤ 3.6 V

Output voltage (2)

Symbol	Item	Min	Typical	Max	Unit	Condition
VOH	Output High Voltage for	DVCC33IO - 0.4	−	−	V	IOH = -1.0 mA 3.0 ≤ DVCC3IO ≤ 3.6 V

Others

Symbol	Item	Min	Typical	Max	Unit	Condition
ILI	Input Leakage Current	–	TBD	±10	μA	VSS ≤ VIN ≤ DVCC33IO
ILO	Output Leakage Current	–	TBD	±10	μA	0.2V ≤ VIN ≤ DVCC33IO - 0.2 V
CIO	Pin Capacitance	–	TBD	–	pF	fc = 1 MHz
VTH	Schmitt Width for PA0, PA2, PA4, PA6, PB0-7, PC0, PD4, PD6, PE0-5, PF0-7, PG0-7, /RESET	–	0.4	–	V	3.0 ≤ DVCC3IO ≤ 3.6 V

Note) Unless specified otherwise, the Typ values are based on Ta = 25°C, DVCC33IO = 3.3 V, DVCC12 = 1.2 V.

ICC current consumption

Symbol	Item	Min	Typical	Max	Unit	Condition	
ICC	NORMAL	−	TBD	TBD	mA	PLL_ON f_{FCLK} = 144 MHz	DVCC33IO = 3.6 V
		−	TBD	TBD			AVDD33ADC = 3.6 V DVCC33DRM = 3.6 V AVDD33USBx = 3.6 V
		—	TBD	TBD			DVCC12 = 1.3 V
		−	TBD	TBD			DVCC12DRM = 1.3 V DVCC12USB = 1.3 V DVDD12PLL = 1.3 V
		−	TBD	TBD		PLL_OFF f_{FCLK} = 12 MHz	DVCC33IO = 3.6 V
		−	TBD	TBD			AVDD33ADC = 3.6 V DVCC33DRM = 3.6 V AVDD33USBx = 3.6 V
		—	TBD	TBD			DVCC12 = 1.3 V
		−	TBD	TBD			DVCC12DRM = 1.3 V DVCC12USB = 1.3 V DVDD12PLL = 1.3 V
	CPU HALT	−	TBD	TBD	mA	PLL_OFF f_{FCLK} = 12 MHz	DVCC33IO = 3.6 V
		−	TBD	TBD			AVDD33ADC = 3.6 V DVCC33DRM = 3.6 V AVDD33USBx = 3.6 V
		—	TBD	TBD			DVCC12 = 1.3 V
		−	TBD	TBD			DVCC12DRM = 1.3 V DVCC12USB = 1.3 V DVDD12PLL = 1.3 V

Operating status: NORMAL

For the CPU, a slightly modified program based on Drystone2.1 is running:

Program memory: Built-in SRAM

Data memory: Built-in DRAM

Stack memory: Built-in SRAM

The peripheral circuits are running with the TSB original flows.

Operating status: HALT

CPU in HALT state

USB in Suspend state

The peripheral circuits are running with the TSB original flows.

Note 1) Unless specified otherwise, the Typ values are based on Ta = 25°C, DVCC33IO = 3.3 V, DVCC12 = 1.2 V.

Note 2) Measurement conditions of ICC: Memory bus terminal CL = 30 pF; output terminals other than the memory bus are open; the input terminals are level-fixed.

Note 3) The above data shows data not for the debug mode.

4.3 AC Electrical Characteristics

The AC specifications shown below are the measurement results under the following conditions unless specified otherwise.

AC measurement conditions

- The “T” used in the equations in the table shows the period of internal bus frequency (f_{HCLK}) × 2.
- Output level: High = 0.7 × DVCC33IO, Low = 0.3 × DVCC33IO
- Input level: High =0.9 × DVCC33IO, Low = 0.1 × DVCC33IO
- Load capacitance: CL = 30 pF

Note) The “equations” in the table show the specifications in the range of DVCC33IO = 3.0 V to 3.6 V, DVCC12 = 1.1 to 1.3 V.

4.3.1 Basic Bus Cycle

Read cycle (Asynchronous mode)

No.	Item	Symbol	Equation		f_{HCLK} = 144 MHz N = 8 M = 3 K = 8 L = 4 P = 1	Unit
			Min	Max		
1	Internal bus period x 2 (=T, internal SMCCLK)	t_{CYC}	13.9	166.6	13.9	nS
2	A0 to A25 enable -> D0 to D31 input	t_{AD}		(N)T – 35.0	76.2	
3	SMCOEn falling -> D0 to D31 input	t_{OED}		(N-M)T – 17.0	52.5	
4	SMCOEn low-level pulse width	t_{OEW}	(N-M)T – 13.0		56.5	
5	A0 to A25 enable -> SMCOEn falling	t_{AOE}	MT – 13.0		28.7	
6	SMCOEn rising -> D0 to D31 hold	t_{HR}	0		0	
7	SMCOEn high-level pulse width	t_{OEHW}	MT – 13.0		28.7	

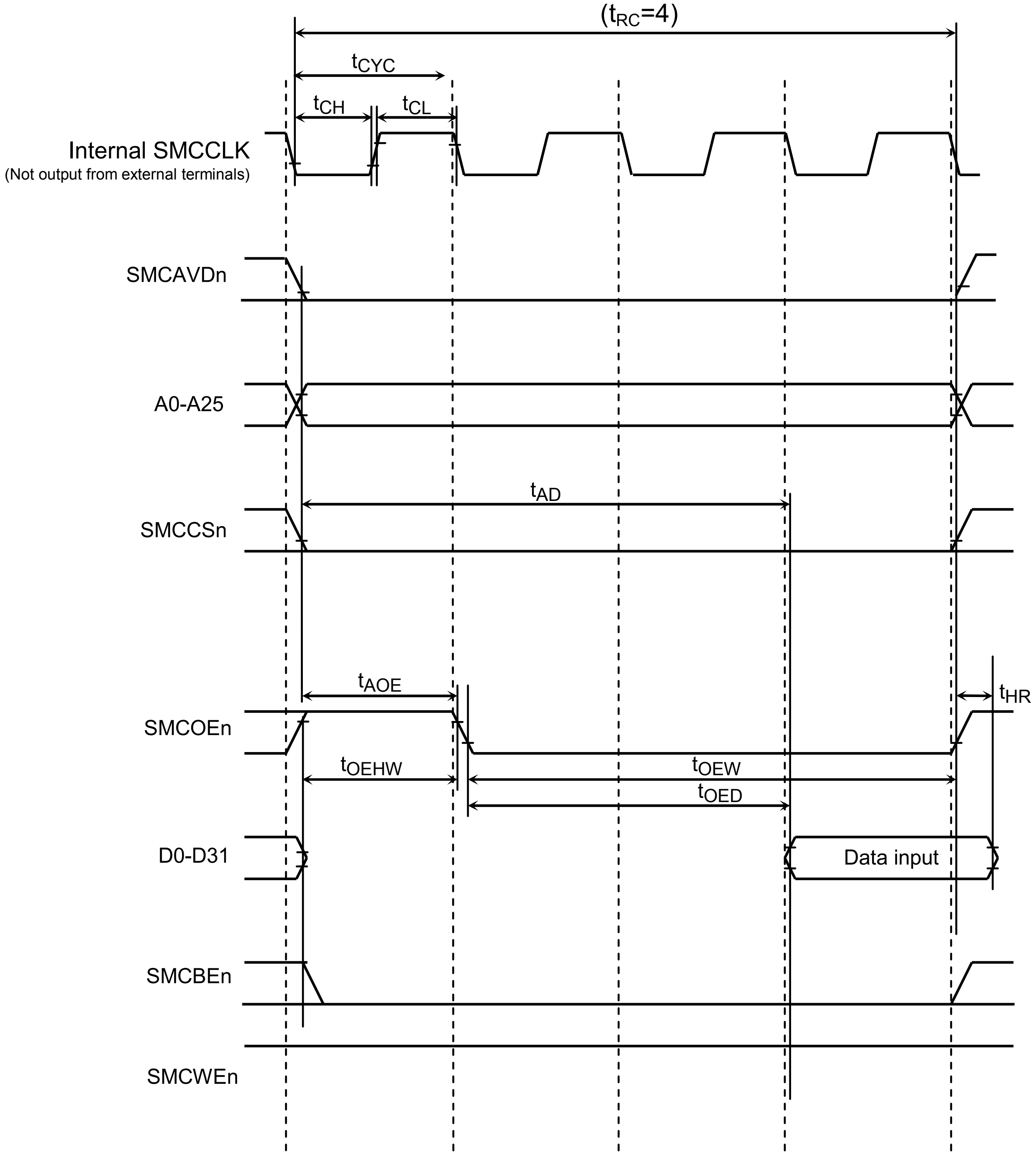
Write cycle (Asynchronous mode)

8	D0 to D31 enable -> SMCWEn rising	t_{DW}	LT – 23.0		32.6	nS
9	D0 to D31 enable -> SMCBEn rising (bls=1)	t_{SDS}	(L+1)T – 23.0		46.5	
10	SMCWEn low-level pulse width	t_{WW}	LT – 13.0		42.6	
11	A0 to A25 enable -> SMCWEn falling	t_{AW}	T – 13.0		0.9	
12	SMCWEn rising -> A0 to A25 hold	t_{WA}	(K-L-1)T – 10.0		31.7	
13	SMCWEn rising -> D0 to D31 hold	t_{WD}	(K-L-1)T – 10.0		31.7	
14	SMCOEn rising -> D0 to D31 output	t_{TR}	（P+1）T－10		17.8	
15	Data byte control to Write end time	t_{SBW}	(L+1)T – 13.0		56.5	

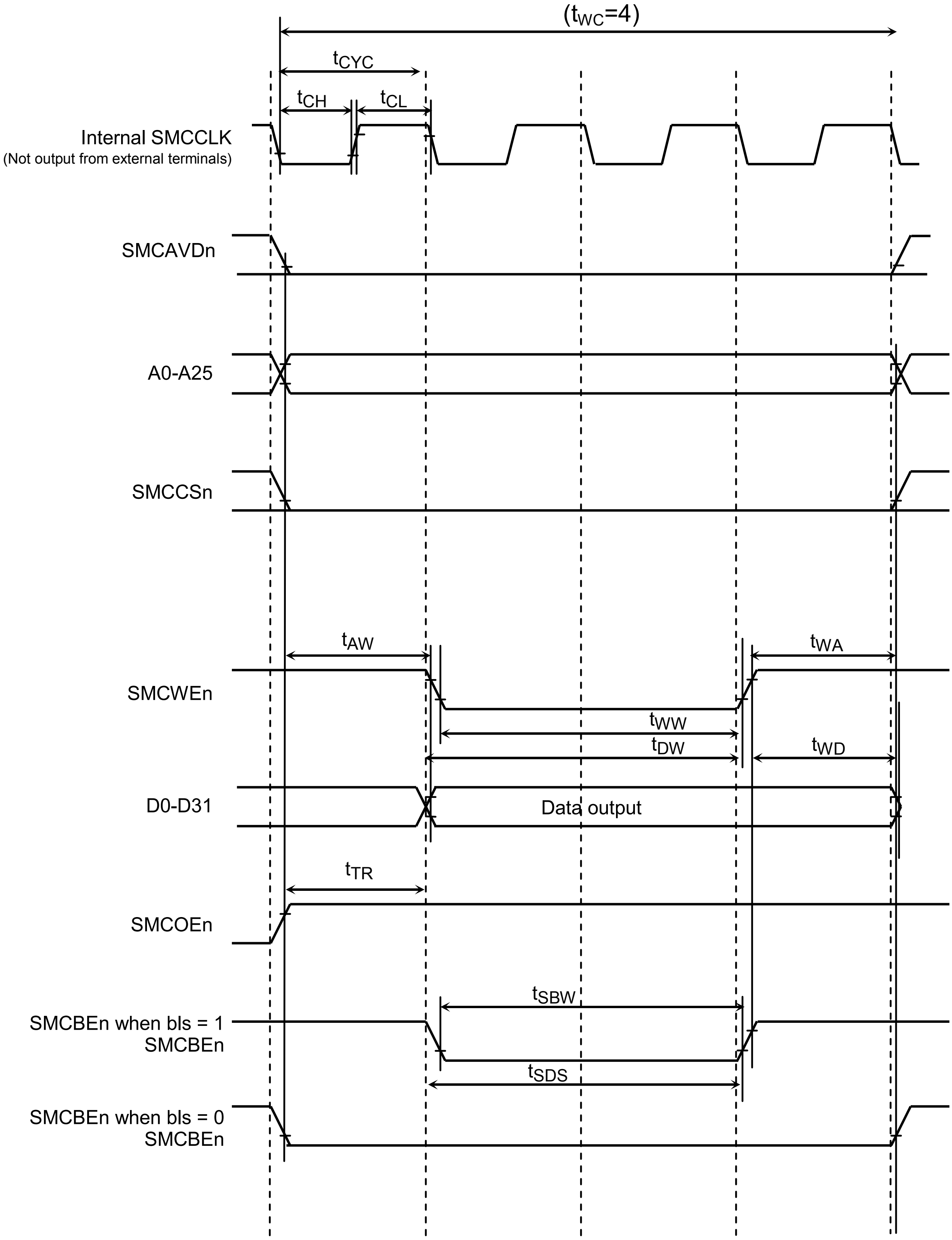
- The variables used in the equations in the table are defined as follows:

N = Number of t_{RC} cycles ≥ 3　　M = Number of t_{CEOE} cycles ≥ 1
K = Number of t_{WC} cycles ≥ 3　　L = Number of t_{WP} cycles ≥ 1
P = Number of t_{TR} cycles ≥ 1

(1) Asynchronous memory read cycle (Example when set to t_{RC}=4, t_{CEOE}=1)



(2) Asynchronous memory write cycle (Example when set to tWC=4, tWP=2)



4.3.2 SSP Controller

AC measurement conditions

- The “T” used in the equations in the table shows the period of internal prescaler input clock f_{PCLK}.
- Output level: High = 0.7 × DVCC33IO, Low = 0.3 × DVCC33IO
- Input level: High = 0.9 × DVCC33IO, Low = 0.1 × DVCC33IO
- Load capacitance: CL = 30 pF

Note) The “equations” in the table show the specifications in the range of DVCC3IO = 3.0 V to 3.6 V, DVCC12 = 1.1 to 1.3 V.

Item	Symbol	Equation		HCLK 144 MHz (m = 8 n = 12)	Unit
		Min	Max		
SPxCLK period (Master)	T_m	(m)T where 50 nS or more		55.6 (18 MHz)	nS
SPxCLK period (Slave)	T_s	(n)T		83.3 (12 MHz)	
SPxCLK rising time	t_r		10.0	10.0	
SPxCLK falling time	t_f		10.0	10.0	
SPxCLK low-level pulse width for master mode	t_{WLM}	(m)T / 2 - 7.0		20.8	
SPxCLK high-level pulse width for master mode	t_{WHM}	(m)T / 2 - 7.0		20.8	
SPxCLK low-level pulse width for slave mode	t_{WLS}	(n)T / 2 - 7.0		34.65	
SPxCLK high-level pulse width for slave mode	t_{WHS}	(n)T / 2 - 7.0		34.65	
Master mode: SPxCLK rising/falling -> Output data enable	t_{ODSM}		15.0	15.0	
Master mode: SPxCLK rising/falling -> Output data hold	t_{ODHM}	(m)T/2 -10		17.8	
Master mode: SPxCLK rising/falling -> Input data enable, Delay time	t_{IDSM}		(m)T /2 – 15	12.8	
Master mode: SPxCLK rising/falling -> Input data hold	t_{IDHM}	5.0		5.0	
Master mode: SPxFSS enable -> SPxCLK rising/falling	t_{OFSM}	(m)T -10	(m)T+10	45.6 – 65.6	
Slave mode: SPxCLK rising/falling (Output data enable, Delay time	t_{ODSS}		(3T) + 15	35.8	
Slave mode: SPxCLK rising/falling -> Output data hold	t_{ODHS}	(n)T /2 ＋ (2T)		55.5	
Slave mode: SPxCLK rising/falling -> Input data enable, Delay time	t_{IDSS}		(n)T /2 ＋ (2T) - 10.0	45.5	
Slave mode: SPxCLK rising/falling -> Input data hold	t_{IDHS}	(3T) +10		30.8	
Slave mode: SPxFSS enable -> SPxCLK rising/falling	t_{OFSS}	(n)T			

Note 1) The communication baud rate clock needs to be set in the following condition ranges:

Master mode:

m = (<CPSDVSR> × (1+<SCR>)) = f_{PCLK} / SPxCLK

Only an even number can be set in <CPSDVR>. The range of “m” is as follows: 65024 ≥ m ≥ 8.

Slave mode:

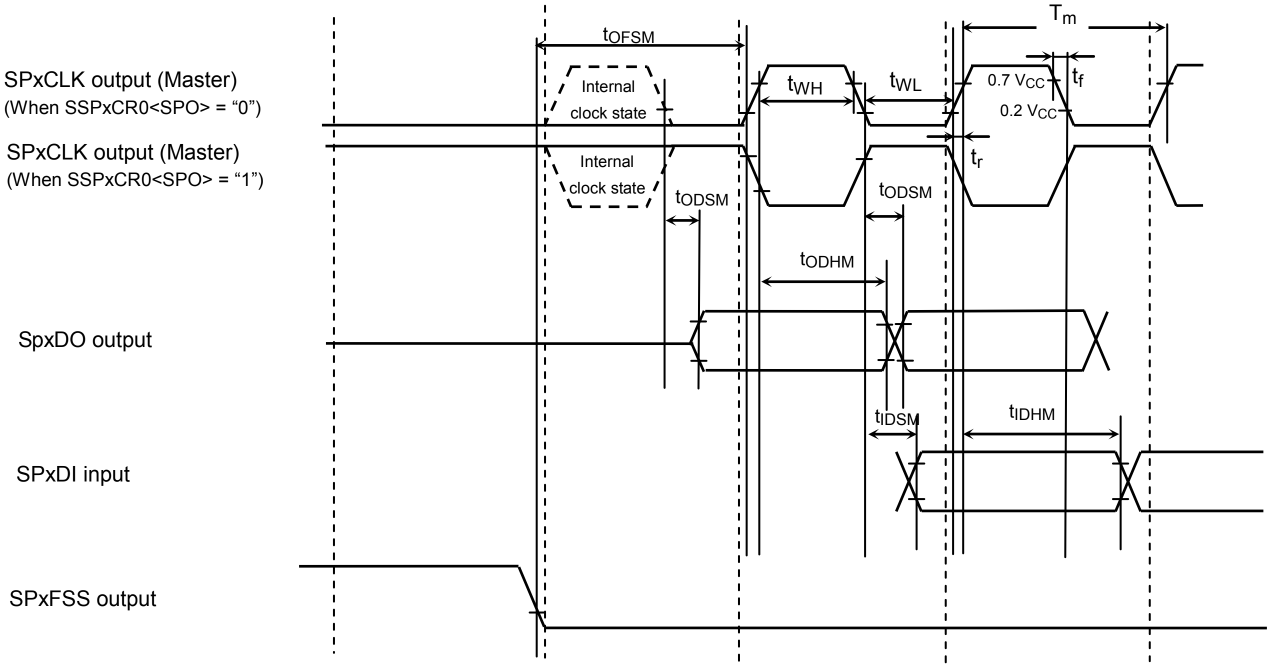
n = f_{PCLK} / SPxCLK (65024 ≥ n ≥ 12)

SSP SPI mode (Master)

$f_{PCLK} \geq 8 \times$ SPxCLK (maximum)

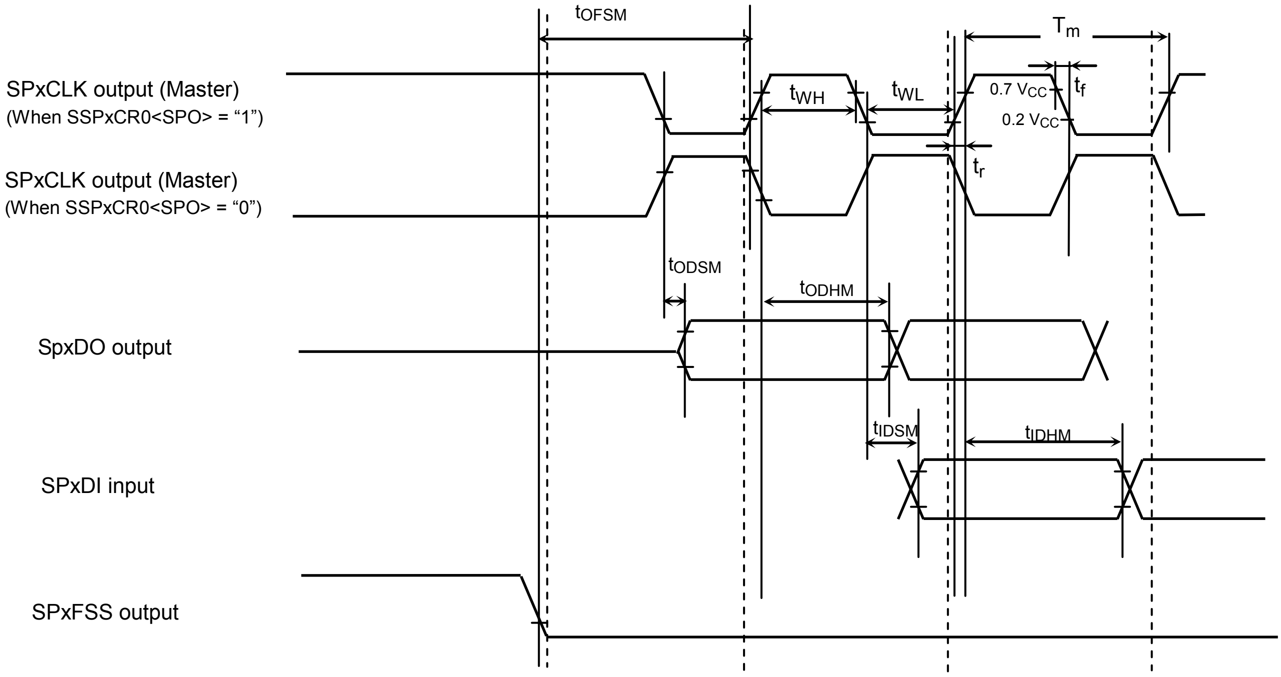
$f_{PCLK} \geq 65024 \times$ SPxCLK (minimum)

(1) Master SSPxCR0<SPH>= "0" (Data is latched at the 1st edge.)



SSP SPI mode (Master)

(2) Master SSPxCR0<SPH>= "1" (Data is latched at the 2nd edge.)

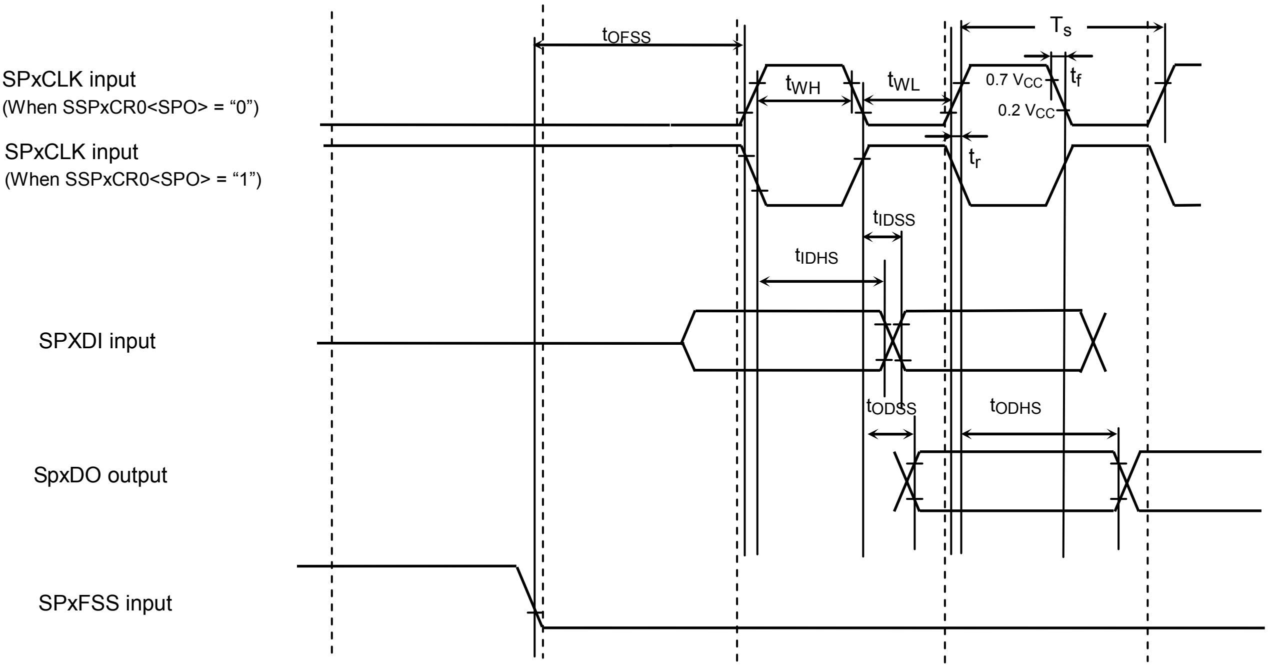


SSP SPI mode (Slave)

$f_{PCLK} \geq 12 \times$ SPxCLK (maximum)

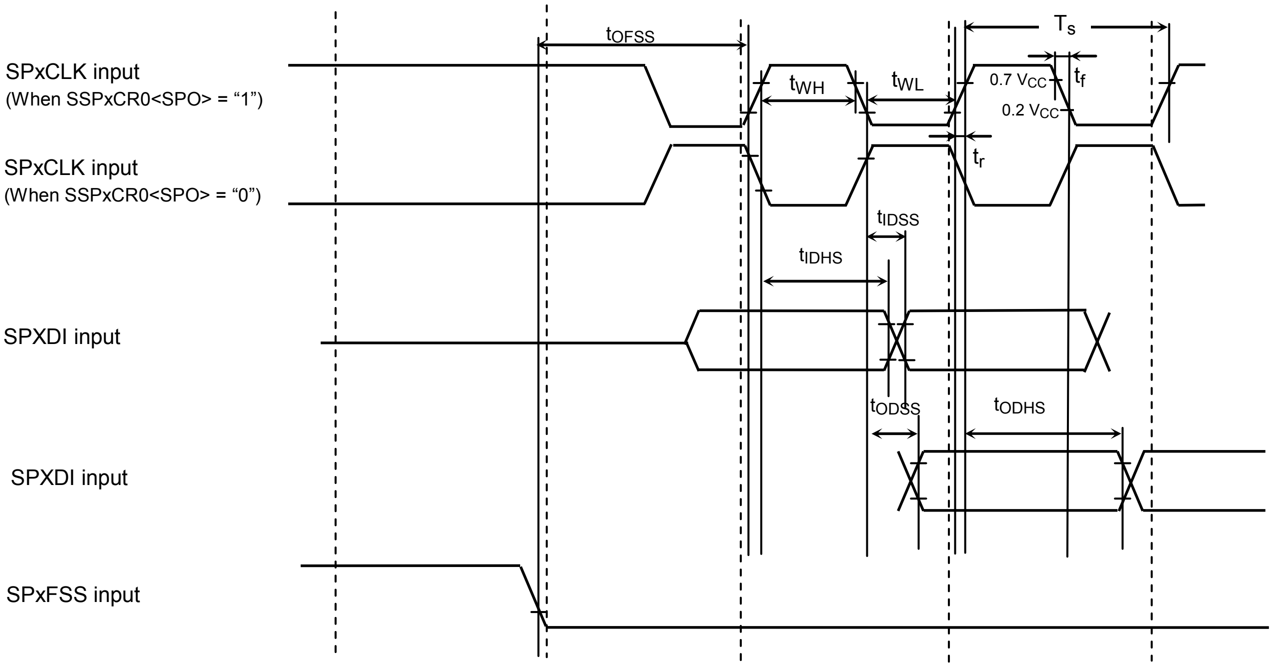
$f_{PCLK} \geq 65024 \times$ SPxCLK (minimum)

(3) Slave SSPxCR0<SPH>= “0” (Data is latched at the 1st edge.)



SSP SPI mode (Slave)

(4) Slave SSPxCR0<SPH>= “1” (Data is latched at the 2nd edge.)



4.4 AD Conversion Characteristics

AVDD33ADC = DVCC33IO , AVSSADC = DVSSCOM

Item	Symbol	Condition	Min	Typ.	Max	Unit
AD converter power supply voltage	AVDD33ADC		3.0	3.3	3.6	V
AD converter GND	AVSSADC		0			V
Analog input voltage range	AVIN		AVSSADC		3.0	V
Differential non-linear error (DNL)	ERDF	AVDD33ADC = 3.0 to 3.6 V		TBD	TBD	LSB
Integral non-linear error (INL)	ERINT			TBD	TBD	LSB
Total error (Not including a quantization error)	ERT			TBD	±10	LSB

Note 1) A quantization error of ±0.5 LSB is not included.

Note 2) 1LSB = (AVDD33ADC − AVSSADC)/1024 [V]

Note 3) The maximum operating clock (ADCLK) of the AD converter is 9 MHz, and the minimum operating clock is 1.125 MHz.
The minimum conversion time is 2.56 μS for 9 MHz, and the maximum conversion time is 13.44 μS for 1.125 MHz.

Note 4) Establish an open process at unused AIN terminals (Do not pull up).

Example of conversion time

f_{PCLK} = 144 MHz

<ADCLK1:0>	ADCLK[MHz]	Conversion time [us]
00	9	2.55
01	4.5	4.11
10	2.25	7.22
11	1.125	13.44

Conversion speed = 14 × (1/ADCLK) + 160 × (1/PCLK)

5. List of Special Function Registers

[1] DMAC (DMA Controller) (1/2)

base address = 0x4000_0000

Register Name	Address (base+)	Type	Description
DMACIntStatus	0x0000	RO	Interrupt status register
DMACIntTCStatus	0x0004	RO	Terminal count interrupt status register
DMACIntTCClear	0x0008	WO	Terminal count interrupt clear register
DMACIntErrorStatus	0x000C	RO	Error request interrupt status register
DMACIntErrClr	0x0010	WO	Error request interrupt clear register
DMACRawIntTCStatus	0x0014	RO	Original terminal count interrupt status register
DMACRawIntErrorStatus	0x0018	RO	Original error request interrupt clear register
DMACEnbldChns	0x001C	RO	DMA channel enable register
DMACSoftBReq	0x0020	R/W	Software DMA Burst request register
DMACSoftSReq	0x0024	R/W	Software DMA Single request register
-	0x0028	-	reserved
-	0x002C	-	reserved
DMACConfiguration	0x0030	R/W	DMA setting register
-	0x0034	-	reserved
DMACC0SrcAddr	0x0100	R/W	DMA channel 0 transfer source address register
DMACC0DestAddr	0x0104	R/W	DMA channel 0 transfer destination address register
DMACC0LLI	0x0108	R/W	DMA channel 0 linked list item register
DMACC0Control	0x010C	R/W	DMA channel 0 control register
DMACC0Configuration	0x0110	R/W	DMA channel 0 setting register
DMACC1SrcAddr	0x0120	R/W	DMA channel 1 transfer source address register
DMACC1DestAddr	0x0124	R/W	DMA channel 1 transfer destination address register
DMACC1LLI	0x0128	R/W	DMA channel 1 linked list item register
DMACC1Control	0x012C	R/W	DMA channel 1 control register
DMACC1Configuration	0x0130	R/W	DMA channel 1 setting register
DMACC2SrcAddr	0x0140	R/W	DMA channel 2 transfer source address register
DMACC2DestAddr	0x0144	R/W	DMA channel 2 transfer destination address register
DMACC2LLI	0x0148	R/W	DMA channel 2 linked list item register
DMACC2Control	0x014C	R/W	DMA channel 2 control register
DMACC2Configuration	0x0150	R/W	DMA channel 2 setting register
DMACC3SrcAddr	0x0160	R/W	DMA channel 3 transfer source address register
DMACC3DestAddr	0x0164	R/W	DMA channel 3 transfer destination address register
DMACC3LLI	0x0168	R/W	DMA channel 3 linked list item register
DMACC3Control	0x016C	R/W	DMA channel 3 control register
DMACC3Configuration	0x0170	R/W	DMA channel 3 setting register
DMACC4SrcAddr	0x0180	R/W	DMA channel 4 transfer source address register
DMACC4DestAddr	0x0184	R/W	DMA channel 4 transfer destination address register
DMACC4LLI	0x0188	R/W	DMA channel 4 linked list item register
DMACC4Control	0x018C	R/W	DMA channel 4 control register
DMACC4Configuration	0x0190	R/W	DMA channel 4 setting register
DMACC5SrcAddr	0x01A0	R/W	DMA channel 5 transfer source address register
DMACC5DestAddr	0x01A4	R/W	DMA channel 5 transfer destination address register
DMACC5LLI	0x01A8	R/W	DMA channel 5 linked list item register
DMACC5Control	0x01AC	R/W	DMA channel 5 control register
DMACC5Configuration	0x01B0	R/W	DMA channel 5 setting register
DMACC6SrcAddr	0x01C0	R/W	DMA channel 6 transfer source address register
DMACC6DestAddr	0x01C4	R/W	DMA channel 6 transfer destination address register
DMACC6LLI	0x01C8	R/W	DMA channel 6 linked list item register
DMACC6Control	0x01CC	R/W	DMA channel 6 control register
DMACC6Configuration	0x01D0	R/W	DMA channel 6 setting register

[1] DMAC (DMA Controller) (2/2)

base address = 0x4000_0000

Register Name	Address (base+)	Type	Description
DMACC7SrcAddr	0x01E0	R/W	DMA channel 7 transfer source address register
DMACC7DestAddr	0x01E4	R/W	DMA channel 7 transfer destination address register
DMACC7LLI	0x01E8	R/W	DMA channel 7 linked list item register
DMACC7Control	0x01EC	R/W	DMA channel 7 control register
DMACC7Configuration	0x01F0	R/W	DMA channel 7 setting register
-	0x0500	-	reserved
-	0x0504	-	reserved
-	0x0508	-	reserved
-	0x050C	-	reserved
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[2] eDRAMC (eDRAM Controller)

base address = 0x4000_1000

Register Name	Address (base+)	Type	Description
R_MDRAC	0x0000	R/W	Read AC parameter register
R_MDWAC	0x0004	R/W	Write AC parameter register
R_MDREFAC	0x0008	R/W	Auto refresh AC parameter register
R_MDRSTAC	0x000C	R/W	Reset AC parameter register
R_MDTRASMAX	0x0010	R/W	tRAS Max AC parameter register
R_MDREFMODE	0x0014	R/W	Auto refresh mode setting register
R_MDRL	0x0018	R/W	Read latency setting register
R_MDSTART	0x001C	R/W	Controller start register
R_MDSRST1	0x0020	R/W	Controller software reset register
R_MDSRST2	0x0024	R/W	Microsoft software reset register
R_MDSTATUS1	0x0028	RO	Controller status register
R_MDSTATUS2	0x002C	RO	Macro status register
R_MDCKE	0x0030	R/W	Macro clock enable register
-	0x0034	-	reserved

[3] SMC (Static Memory Controller)

base address = 0x4000_4000

Register Name	Address (base+)	Type	Description
-	0x0000	-	reserved
-	0x0004	-	reserved
-	0x0008	-	reserved
-	0x000C	-	reserved
smc_direct_cmd	0x0010	WO	SMC Direct Command Register
smc_set_cycles	0x0014	WO	SMC Set Cycles Register
smc_set_opmode	0x0018	WO	SMC Set Opmode Register
-	0x0020	-	reserved
smc_sram_cycles0_0	0x0100	RO	SMC SRAM Cycles Register
smc_opmode0_0	0x0104	RO	SMC Opmode Register
smc_sram_cycles0_1	0x0120	RO	SMC SRAM Cycles Register
smc_opmode0_1	0x0124	RO	SMC Opmode Register
-	0x0140	-	reserved
-	0x0144	-	reserved
-	0x0160	-	reserved
-	0x0164	-	reserved
-	0x0200	-	reserved
-	0x0204	-	reserved
-	0x0E00	-	reserved
-	0x0E04	-	reserved
-	0x0E08	-	reserved
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[4] CG (Clock Controller)

base address = 0x4000_5000

Register Name	Address (base+)	Type	Description
CG_PLLCTRL0	0x0000	R/W	CG PLL Control Register 0
CG_PLLCTRL1	0x0004	R/W	CG PLL Control Register 1
-	0x0008	-	reserved
CG_PLLCTRL3	0x000C	R/W	CG PLL Control Register 3
CG_PLLCTRL4	0x0010	R/W	CG PLL Control Register 4
CG_PLLCTRL5	0x0014	R/W	CG PLL Control Register 5
CG_PLLCTRL6	0x0018	R/W	CG PLL Control Register 6
CG_CLKDIS	0x0020	R/W	CG Clock Disable Register
-	0x0100	-	reserved
-	0x0104	-	reserved
-	0x0108	-	reserved
CG_BSIFCTRL	0x0200	R/W	CG Bit Stream Interface Control Register
CG_DMASELR	0x0300	R/W	DMA Request Select Control Register
-	0x0304	-	reserved
CG_SYSTICK	0x0310	R/W	SYSTICK Control Register

[5] WDT (Watch Dog Timer)

base address = 0x4000_6000

Register Name	Address (base+)	Type	Description
WdogLoad	0x0000	R/W	Wdog timer interval
WdogValue	0x0004	RO	Current Wdog timer counter value
WdogControl	0x0008	R/W	Wdog timer control register
WdogIntClr	0x000C	WO	Wdog timer interrupt clear register
WdogRIS	0x0010	RO	Original (before-masking) Wdog timer interrupt status
WdogMIS	0x0014	RO	(After-masking) Wdog timer interrupt status
WdogLock	0x0C00	RO WO	Wdog Lock register
-	0x0F00	-	reserved
-	0x0F04	-	reserved
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[6] ADC (Analog Digital Converter)

base address = 0x4000_7000

Register Name	Address (base+)	Type	Description
ADCTRL	0x0000	R/W	AD control register
ADSELAIN	0x0004	R/W	AD channel select register
ADREG	0x0008	RO	AD conversion result register
ADCLK	0x000C	R/W	AD conversion clock setting register
ADIE	0x0010	R/W	AD interrupt enable register
ADIS	0x0014	RO	AD interrupt status register
ADIC	0x0018	WO	AD interrupt clear register

[7] PA (PortA)

base address = 0x4000_8000

Register Name	Address (base+)	Type	Description
GPIOADATA	0x0000 0x03FC	R/W	Data register
GPIOADIR	0x0400	R/W	Data direction register
GPIOAFR1	0x0424	R/W	Function register 1
GPIOAFR2	0x0428	R/W	Function register 2

[8] PB (PortB)

base address = 0x4000_9000

Register Name	Address (base+)	Type	Description
GPIOBDATA	0x0000 0x03FC	R/W	Data register
GPIOBDIR	0x0400	R/W	Data direction register
GPIOBFR1	0x0424	R/W	Function register 1
GPIOBFR2	0x0428	R/W	Function register 2
GPIOBODE	0x0C00	R/W	Open drain output enable register

[9] PC (PortC)

base address = 0x4000_A000

Register Name	Address (base+)	Type	Description
GPIOCDATA	0x0000 0x03FC	R/W	Data register
GPIOCDIR	0x 0400	R/W	Data direction register
GPIOCFR1	0x 0424	R/W	Function register 1
GPIOCFR2	0x 0428	R/W	Function register 2

[10] PD (PortD)

base address = 0x4000_B000

Register Name	Address (base+)	Type	Description
GPIODDATA	0x0000 0x03FC	R/W	Data register
GPIODDIR	0x 0400	R/W	Data direction register
GPIODFR1	0x 0424	R/W	Function register 1
GPIODFR2	0x 0428	R/W	Function register 2

[11] PE (PortE)

base address = 0x4000_C000

Register Name	Address (base+)	Type	Description
GPIOEDATA	0x0000 0x03FC	R/W	Data register
GPIOEDIR	0x 0400	R/W	Data direction register
GPIOEFR1	0x 0424	R/W	Function register 1
GPIOEFR2	0x 0428	R/W	Function register 2
GPIOEIS	0x0804	R/W	Interrupt detection register
GPIOEIBE	0x0808	R/W	Interrupt both-edge register
GPIOEIEV	0x080C	R/W	Interrupt event register
GPIOEIE	0x0810	R/W	Interrupt enable register
GPIOERIS	0x0814	RO	Pre-interrupt enable status register
GPIOEMIS	0x0818	RO	Post-interrupt enable status register
GPIOEIC	0x081C	WO	Interrupt clear register

[12] PF (PortF)

base address = 0x4000_D000

Register Name	Address (base+)	Type	Description
GPIOFDATA	0x0000 0x03FC	R/W	Data register
GPIOFDIR	0x 0400	R/W	Data direction register
GPIOFFR1	0x 0424	R/W	Function register 1
GPIOFFR2	0x 0428	R/W	Function register 2

[13] PG (PortG)

base address = 0x4000_E000

Register Name	Address (base+)	Type	Description
GPIOGDATA	0x0000 0x03FC	R/W	Data register
GPIOGDIR	0x 0400	R/W	Data direction register
GPIOGFR1	0x 0424	R/W	Function register 1
GPIOGFR2	0x 0428	R/W	Function register 2

[14] TC01 (16bit Timer01)

base address = 0x4000_F000

Register Name	Address (base+)	Type	Description
Timer0Load	0x0000	R/W	Timer0 Load value
Timer0Value	0x0004	RO	The current value for Timer0
Timer0Control	0x0008	R/W	Timer0 control register
Timer0IntClr	0x000C	WO	Timer0 interrupt clear
Timer0RIS	0x0010	RO	Timer0 raw interrupt status
Timer0MIS	0x0014	RO	Timer0 masked interrupt status
Timer0BGLoad	0x0018	R/W	Background load value for Timer0
Timer0Mode	0x001C	R/W	Timer mode register
-	0x0020	-	reserved
-	0x0040	-	reserved
-	0x0060	-	reserved
-	0x0064	-	reserved
-	0x0068	-	reserved
Timer0Compare1	0x00A0	RO	Timer0 Compare value
Timer0CmpIntClr1	0x00C0	R/W	Timer0 Compare Interrupt clear
Timer0CmpEn	0x00E0	R/W	Timer0 Compare Enable
Timer0CmpRIS	0x00E4	RO	Timer0 Compare raw interrupt status
Timer0CmpMIS	0x00E8	RO	Timer0 Compare masked int status
Timer0BGCmp	0x00EC	R/W	Background compare value for Timer0
-	0x00F0	-	reserved
Timer1Load	0x0100	R/W	Timer1 Load value
Timer1Value	0x0104	RO	The current value for Timer1
Timer1Control	0x0108	R/W	Timer1 control register
Timer1IntClr	0x010C	WO	Timer1 interrupt clear
Timer1RIS	0x0110	RO	Timer1 raw interrupt status
Timer1MIS	0x0114	RO	Timer1 masked interrupt status
Timer1BGLoad	0x0118	R/W	Background load value for Timer1
-	0x011C	-	reserved
-	0x0120	-	reserved
-	0x0140	-	reserved
-	0x0160	-	reserved
-	0x0164	-	reserved
-	0x0168	-	reserved
-	0x01A0	-	reserved
-	0x01C0	-	reserved
-	0x01E0	-	reserved
-	0x01E4	-	reserved
-	0x01E8	-	reserved
-	0x01EC	-	reserved
-	0x01F0	-	reserved
-	0x0E00	-	reserved
-	0x0E04	-	reserved
-	0x0E08	-	reserved
-	0x0E0C	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[15] TC23 (16bit Timer23)

base address = 0x4001_0000

Register Name	Address (base+)	Type	Description
Timer2Load	0x0000	R/W	Timer2 Load value
Timer2Value	0x0004	RO	The current value for Timer2
Timer2Control	0x0008	R/W	Timer2 control register
Timer2IntClr	0x000C	WO	Timer2 interrupt clear
Timer2RIS	0x0010	RO	Timer2 raw interrupt status
Timer2MIS	0x0014	RO	Timer2 masked interrupt status
Timer2BGLoad	0x0018	R/W	Background load value for Timer2
Timer2Mode	0x001C	R/W	Timer mode register
-	0x0020	-	reserved
-	0x0040	-	reserved
-	0x0060	-	reserved
-	0x0064	-	reserved
-	0x0068	-	reserved
Timer2Compare1	0x00A0	RO	Timer2 Compare value
Timer2CmpIntClr1	0x00C0	R/W	Timer2 Compare Interrupt clear
Timer2CmpEn	0x00E0	R/W	Timer2 Compare Enable
Timer2CmpRIS	0x00E4	RO	Timer2 Compare raw interrupt status
Timer2CmpMIS	0x00E8	RO	Timer2 Compare masked int status
Timer2BGCmp	0x00EC	R/W	Background compare value for Timer2
-	0x00F0	-	reserved
Timer3Load	0x0100	R/W	Timer3 Load value
Timer3Value	0x0104	RO	The current value for Timer3
Timer3Control	0x0108	R/W	Timer3 control register
Timer3IntClr	0x010C	WO	Timer3 interrupt clear
Timer3RIS	0x0110	RO	Timer3 raw interrupt status
Timer3MIS	0x0114	RO	Timer3 masked interrupt status
Timer3BGLoad	0x0118	R/W	Background load value for Timer3
-	0x011C	-	reserved
-	0x0120	-	reserved
-	0x0140	-	reserved
-	0x0160	-	reserved
-	0x0164	-	reserved
-	0x0168	-	reserved
-	0x01A0	-	reserved
-	0x01C0	-	reserved
-	0x01E0	-	reserved
-	0x01E4	-	reserved
-	0x01E8	-	reserved
-	0x01EC	-	reserved
-	0x01F0	-	reserved
-	0x0E00	-	reserved
-	0x0E04	-	reserved
-	0x0E08	-	reserved
-	0x0E0C	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[16] TC45 (16bit Timer45)

base address = 0x4001_1000

Register Name	Address (base+)	Type	Description
Timer4Load	0x0000	R/W	Timer4 Load value
Timer4Value	0x0004	RO	The current value for Timer4
Timer4Control	0x0008	R/W	Timer4 control register
Timer4IntClr	0x000C	WO	Timer4 interrupt clear
Timer4RIS	0x0010	RO	Timer4 raw interrupt status
Timer4MIS	0x0014	RO	Timer4 masked interrupt status
Timer4BGLoad	0x0018	R/W	Background load value for Timer4
Timer4Mode	0x001C	R/W	Timer4 mode register
-	0x0020	-	reserved
-	0x0040	-	reserved
-	0x0060	-	reserved
-	0x0064	-	reserved
-	0x0068	-	reserved
Timer4Compare1	0x00A0	RO	Timer4 Compare value
Timer4CmpIntClr1	0x00C0	R/W	Timer4 Compare Interrupt clear
Timer4CmpEn	0x00E0	R/W	Timer4 Compare Enable
Timer4CmpRIS	0x00E4	RO	Timer4 Compare raw interrupt status
Timer4CmpMIS	0x00E8	RO	Timer4 Compare masked int status
Timer4BGCmp	0x00EC	R/W	Background compare value for Timer4
-	0x00F0	-	reserved
Timer5Load	0x0100	R/W	Timer5 Load value
Timer5Value	0x0104	RO	The current value for Timer5
Timer5Control	0x0108	R/W	Timer5 control register
Timer5IntClr	0x010C	WO	Timer5 interrupt clear
Timer5RIS	0x0110	RO	Timer5 raw interrupt status
Timer5MIS	0x0114	RO	Timer5 masked interrupt status
Timer5BGLoad	0x0118	R/W	Background load value for Timer5
-	0x011C	-	reserved
-	0x0120	-	reserved
-	0x0140	-	reserved
-	0x0160	-	reserved
-	0x0164	-	reserved
-	0x0168	-	reserved
-	0x01A0	-	reserved
-	0x01C0	-	reserved
-	0x01E0	-	reserved
-	0x01E4	-	reserved
-	0x01E8	-	reserved
-	0x01EC	-	reserved
-	0x01F0	-	reserved
-	0x0E00	-	reserved
-	0x0E04	-	reserved
-	0x0E08	-	reserved
-	0x0E0C	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[17] TC67 (16bit Timer67)

base address = 0x4001_2000

Register Name	Address (base+)	Type	Description
Timer6Load	0x0000	R/W	Timer6 Load value
Timer6Value	0x0004	RO	The current value for Timer6
Timer6Control	0x0008	R/W	Timer6 control register
Timer6IntClr	0x000C	WO	Timer6 interrupt clear
Timer6RIS	0x0010	RO	Timer6 raw interrupt status
Timer6MIS	0x0014	RO	Timer6 masked interrupt status
Timer6BGLoad	0x0018	R/W	Background load value for Timer6
Timer6Mode	0x001C	R/W	Timer6 mode register
-	0x0020	-	reserved
-	0x0040	-	reserved
-	0x0060	-	reserved
-	0x0064	-	reserved
-	0x0068	-	reserved
Timer6Compare1	0x00A0	RO	Timer6 Compare value
Timer6CmpIntClr1	0x00C0	R/W	Timer6 Compare Interrupt clear
Timer6CmpEn	0x00E0	R/W	Timer6 Compare Enable
Timer6CmpRIS	0x00E4	RO	Timer6 Compare raw interrupt status
Timer6CmpMIS	0x00E8	RO	Timer6 Compare masked int status
Timer6BGCmp	0x00EC	R/W	Background compare value for Timer6
-	0x00F0	-	reserved
Timer7Load	0x0100	R/W	Timer7 Load value
Timer7Value	0x0104	RO	The current value for Timer7
Timer7Control	0x0108	R/W	Timer7 control register
Timer7IntClr	0x010C	WO	Timer7 interrupt clear
Timer7RIS	0x0110	RO	Timer7 raw interrupt status
Timer7MIS	0x0114	RO	Timer7 masked interrupt status
Timer7BGLoad	0x0118	R/W	Background load value for Timer7
-	0x011C	-	reserved
-	0x0120	-	reserved
-	0x0140	-	reserved
-	0x0160	-	reserved
-	0x0164	-	reserved
-	0x0168	-	reserved
-	0x01A0	-	reserved
-	0x01C0	-	reserved
-	0x01E0	-	reserved
-	0x01E4	-	reserved
-	0x01E8	-	reserved
-	0x01EC	-	reserved
-	0x01F0	-	reserved
-	0x0E00	-	reserved
-	0x0E04	-	reserved
-	0x0E08	-	reserved
-	0x0E0C	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[18] I2C0

base address = 0x4001_3000

Register Name	Address (base+)	Type	Description
I2C0CR1	0x0000	R/W	I2C0 control register 1
I2C0DBR	0x0004	R/W	I2C0 data buffer register
I2C0AR	0x0008	R/W	I2C0 address register
I2C0CR2	0x000C	WO	I2C0 control register 2
I2C0SR		RO	I2C0 status register
I2C0PRS	0x0010	R/W	I2C0 prescaler setting register
I2C0IE	0x0014	R/W	I2C0 interrupt enable register
I2C0IR	0x0018	R/W	I2C0 interrupt request register

[19] I2C1

base address = 0x4001_4000

Register Name	Address (base+)	Type	Description
I2C1CR1	0x0000	R/W	I2C1 control register 1
I2C1DBR	0x0004	R/W	I2C1 data buffer register
I2C1AR	0x0008	R/W	I2C1 address register
I2C1CR2	0x000C	WO	I2C1 control register 2
I2C1SR		RO	I2C1 status register
I2C1PRS	0x0010	R/W	I2C1 prescaler setting register
I2C1IE	0x0014	R/W	I2C1 interrupt enable register
I2C1IR	0x0018	R/W	I2C1 interrupt request register

[21] I2S (1/2)

base address = 0x4001_6000

Register Name	Address (base+)	Type	Description
I2S_IN_CONT	0x0000	R/W	I2S Input Control Register
I2S_IN_FIFOState	0x0010	Varied	I2S Input FIFO State Register
I2S_IN_FIFOData	0x0100	W	I2S Input FIFO Data Access (32bit * 8word * 2bank)
I2S_OUT_CONT	0x0200	R/W	I2S Output Control Register
I2S_OUT_FIFOState	0x0210	Varied	I2S Output FIFO State Register
I2S_OUT_FIFOData	0x0300	W	I2S Output FIFO Data Access (32bit*8word*2bank)
I2S_CG_CNT	0x03FC	R/W	I2S Clock Generator Control Register

[24] UART0

base address = 0x4001_9000

Register Name	Address (base+)	Type	Description
UART0DR	0x0000	R/W	Receive (read) and transmit (write) data register
UART0SR	0x0004	RO	Receive status register
UART0ECR		WO	Error clear register
UART0FR	0x0018	RO	Flag register
-	0x0020	-	reserved
UART0IBRD	0x0024	R/W	Integer baud rate divisor register
UART0FBRD	0x0028	R/W	Fractional baud rate divisor register
UART0LCR_H	0x002C	R/W	Line control register, HIGH byte
UART0CR	0x0030	R/W	Control register
UART0IFLS	0x0034	R/W	Interrupt FIFO level selection register
UART0IMSC	0x0038	R/W	Interrupt master set/clear
UART0RIS	0x003C	RO	Pre-interrupt enable status
UART0MIS	0x0040	RO	Post-interrupt enable status
UART0ICR	0x0044	WO	Interrupt clear register
UART0DMACR	0x0048	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[25] UART1

base address = 0x4001_A000

Register Name	Address (base+)	Type	Description
UART1DR	0x0000	R/W	Receive (read) and transmit (write) data register
UART1SR	0x0004	RO	Receive status register
UART1ECR		WO	Error clear register
UART1FR	0x0018	RO	Flag register
-	0x0020	-	reserved
UART1IBRD	0x0024	R/W	Integer baud rate divisor register
UART1FBRD	0x0028	R/W	Fractional baud rate divisor register
UART1LCR_H	0x002C	R/W	Line control register, HIGH byte
UART1CR	0x0030	R/W	Control register
UART1IFLS	0x0034	R/W	Interrupt FIFO level selection register
UART1IMSC	0x0038	R/W	Interrupt master set/clear
UART1RIS	0x003C	RO	Pre-interrupt enable status
UART1MIS	0x0040	RO	Post-interrupt enable status
UART1ICR	0x0044	WO	Interrupt clear register
UART1DMACR	0x0048	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[26] UART2

base address = 0x4001_B000

Register Name	Address (base+)	Type	Description
UART2DR	0x0000	R/W	Receive (read) and transmit (write) data register
UART2SR	0x0004	RO	Receive status register
UART2ECR		WO	Error clear register
UART2FR	0x0018	RO	Flag register
-	0x0020	-	reserved
UART2IBRD	0x0024	R/W	Integer baud rate divisor register
UART2FBRD	0x0028	R/W	Fractional baud rate divisor register
UART2LCR_H	0x002C	R/W	Line control register, HIGH byte
UART2CR	0x0030	R/W	Control register
UART2IFLS	0x0034	R/W	Interrupt FIFO level selection register
UART2IMSC	0x0038	R/W	Interrupt master set/clear
UART2RIS	0x003C	RO	Pre-interrupt enable status
UART2MIS	0x0040	RO	Post-interrupt enable status
UART2ICR	0x0044	WO	Interrupt clear register
UART2DMACR	0x0048	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[27] UART3

base address = 0x4001_C000

Register Name	Address (base+)	Type	Description
UART3DR	0x0000	R/W	Receive (read) and transmit (write) data register
UART3SR	0x0004	RO	Receive status register
UART3ECR		WO	Error clear register
UART3FR	0x0018	RO	Flag register
-	0x0020	-	reserved
UART3IBRD	0x0024	R/W	Integer baud rate divisor register
UART3FBRD	0x0028	R/W	Fractional baud rate divisor register
UART3LCR_H	0x002C	R/W	Line control register, HIGH byte
UART3CR	0x0030	R/W	Control register
UART3IFLS	0x0034	R/W	Interrupt FIFO level selection register
UART3IMSC	0x0038	R/W	Interrupt master set/clear
UART3RIS	0x003C	RO	Pre-interrupt enable status
UART3MIS	0x0040	RO	Post-interrupt enable status
UART3ICR	0x0044	WO	Interrupt clear register
UART3DMACR	0x0048	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[28] SSP0

base address = 0x4001_D000

Register Name	Address (base+)	Type	Description
SSP0CR0	0x0000	R/W	Control register 0
SSP0CR1	0x0004	R/W	Control register 1
SSP0DR	0x0008	R/W	Receive FIFO (read) and transmit FIFO data register (write)
SSP0SR	0x000C	RO	Status register
SSP0CPSR	0x0010	R/W	Clock prescale register
SSP0IMSC	0x0014	R/W	Interrupt enable/disable register
SSP0RIS	0x0018	RO	Pre-interrupt enable status register
SSP0MIS	0x001C	RO	Post-interrupt enable status register
SSP0ICR	0x0020	WO	Interrupt clear register
SSP0DMACR	0x0024	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[29] SSP1

base address = 0x4001_E000

Register Name	Address (base+)	Type	Description
SSP1CR0	0x0000	R/W	Control register 0
SSP1CR1	0x0004	R/W	Control register 1
SSP1DR	0x0008	R/W	Receive FIFO (read) and transmit FIFO data register (write)
SSP1SR	0x000C	RO	Status register
SSP1CPSR	0x0010	R/W	Clock prescale register
SSP1IMSC	0x0014	R/W	Interrupt enable/disable register
SSP1RIS	0x0018	RO	Pre-interrupt enable status register
SSP1MIS	0x001C	RO	Post-interrupt enable status register
SSP1ICR	0x0020	WO	Interrupt clear register
SSP1DMACR	0x0024	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[30] SSP2

base address = 0x4001_F000

Register Name	Address (base+)	Type	Description
SSP2CR0	0x0000	R/W	Control register 0
SSP2CR1	0x0004	R/W	Control register 1
SSP2DR	0x0008	R/W	Receive FIFO (read) and transmit FIFO data register (write)
SSP2SR	0x000C	RO	Status register
SSP2CPSR	0x0010	R/W	Clock prescale register
SSP2IMSC	0x0014	R/W	Interrupt enable/disable register
SSP2RIS	0x0018	RO	Pre-interrupt enable status register
SSP2MIS	0x001C	RO	Post-interrupt enable status register
SSP2ICR	0x0020	WO	Interrupt clear register
SSP2DMACR	0x0024	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[31] SSP3

base address = 0x4002_0000

Register Name	Address (base+)	Type	Description
SSP3CR0	0x0000	R/W	Control register 0
SSP3CR1	0x0004	R/W	Control register 1
-	0x0008	-	reserved
SSP3SR	0x000C	RO	Status register
SSP3CPSR	0x0010	R/W	Clock prescale register
SSP3IMSC	0x0014	R/W	Interrupt enable/disable register
SSP3RIS	0x0018	RO	Pre-interrupt enable status register
SSP3MIS	0x001C	RO	Post-interrupt enable status register
SSP3ICR	0x0020	WO	Interrupt clear register
SSP3DMACR	0x0024	R/W	DMA control register
-	0x0FE0	-	reserved
-	0x0FE4	-	reserved
-	0x0FE8	-	reserved
-	0x0FEC	-	reserved
-	0x0FF0	-	reserved
-	0x0FF4	-	reserved
-	0x0FF8	-	reserved
-	0x0FFC	-	reserved

[32] USB EHCI

base address = 0x4000_2000

Register Name	Address (base+)	Type	Description
CAPLENGTH	0x0000	RO	Operational register offset address
HCIVERSION	0x0002	RO	EHCI version
HCSPARAMS	0x0004	RO	Number of USB1.1 controllers, and port power switch specifications
HCCPARAMS	0x0008	RO	PARK mode specifications
USBCMD	0x0010	R/W	Instructions requested to Host Controller
USBSTS	0x0014	R/W	Interrupts and various statuses
USBINTR	0x0018	R/W	Interrupt enable/disable
FRINDEX	0x001C	R/W	Periodic Frame List index
CTRLDSSEGMENT	0x0020	R/W	[63:32] address for 64-bit addressing
PERIODICLISTBASE	0x0024	R/W	Address of Periodic Frame List start
ASYNCLISTADDR	0x0028	R/W	Address of next non-periodic queue head
CONFIGFLAG	0x0050	R/W	Controller routing
PORTSC	0x0054	R/W	Port status
-	0x0090	-	reserved
ORGREG01	0x0094	R/W	Packet buffer threshold value
-	0x0098	-	reserved
ORGREG03	0x009C	R/W	Change of the maximum number of AHB burst transfers
-	0x00A0	-	reserved
-	0x00A4	-	reserved

[33] USB OHCI

base address = 0x4000_3000

Register Name	Address (base+)	Type	Description
HcRevision	0x0000	RO	HCI version
HcControl	0x0004	R/W	Definition of Host Controller operation mode
HcCommandStatus	0x0008	R/W	Status reflection and command receipt
HcInterruptStatus	0x000C	R/W	Status of event causing an interrupt
HcInterruptEnable	0x0010	R/W	Interrupt enable
HcInterruptDisable	0x0014	R/W	Interrupt disable
HcHCCA	0x0018	R/W	Communication Area setting
HcPeriodCurrentED	0x001C	R/W	Physical address of the current Isochronous transfer or Interrupt transfer Endpoint Descriptor
HcControlHeadED	0x0020	R/W	Physical address of the first Endpoint Descriptor of the Control transfer list
HcControlCurrentED	0x0024	R/W	Physical address of the current Endpoint Descriptor of the Control transfer list
HcBulkHeadED	0x0028	R/W	Physical address of the first Endpoint Descriptor of the Bulk transfer list
HcBulkCurrentED	0x002C	R/W	Physical address of the current Endpoint Descriptor of the Bulk transfer list
HcDoneHead	0x0030	R/W	Physical address of the last completed Transfer Descriptor in the Done queue.
HcFmInterval	0x0034	R/W	Bit time interval (between SOFs) and maximum packet size
HcFmRemaining	0x0038	R/W	Bit time remaining in the current Frame
HcFmNumber	0x003C	R/W	Frame number
HcPeriodStart	0x0040	R/W	Processing start time value of Periodic Transfer List
HcLSThreshold	0x0044	R/W	Threshold value for determining whether to execute LS packet transfer
HcRhDescriptorA	0x0048	R/W	Characteristics of Route Hub
HcRhDescriptorB	0x004C	R/W	Characteristics of Route Hub
HcRhStatus	0x0050	R/W	Hub status and hub status change
HcRhPortStatus	0x0054	R/W	Port event control and report

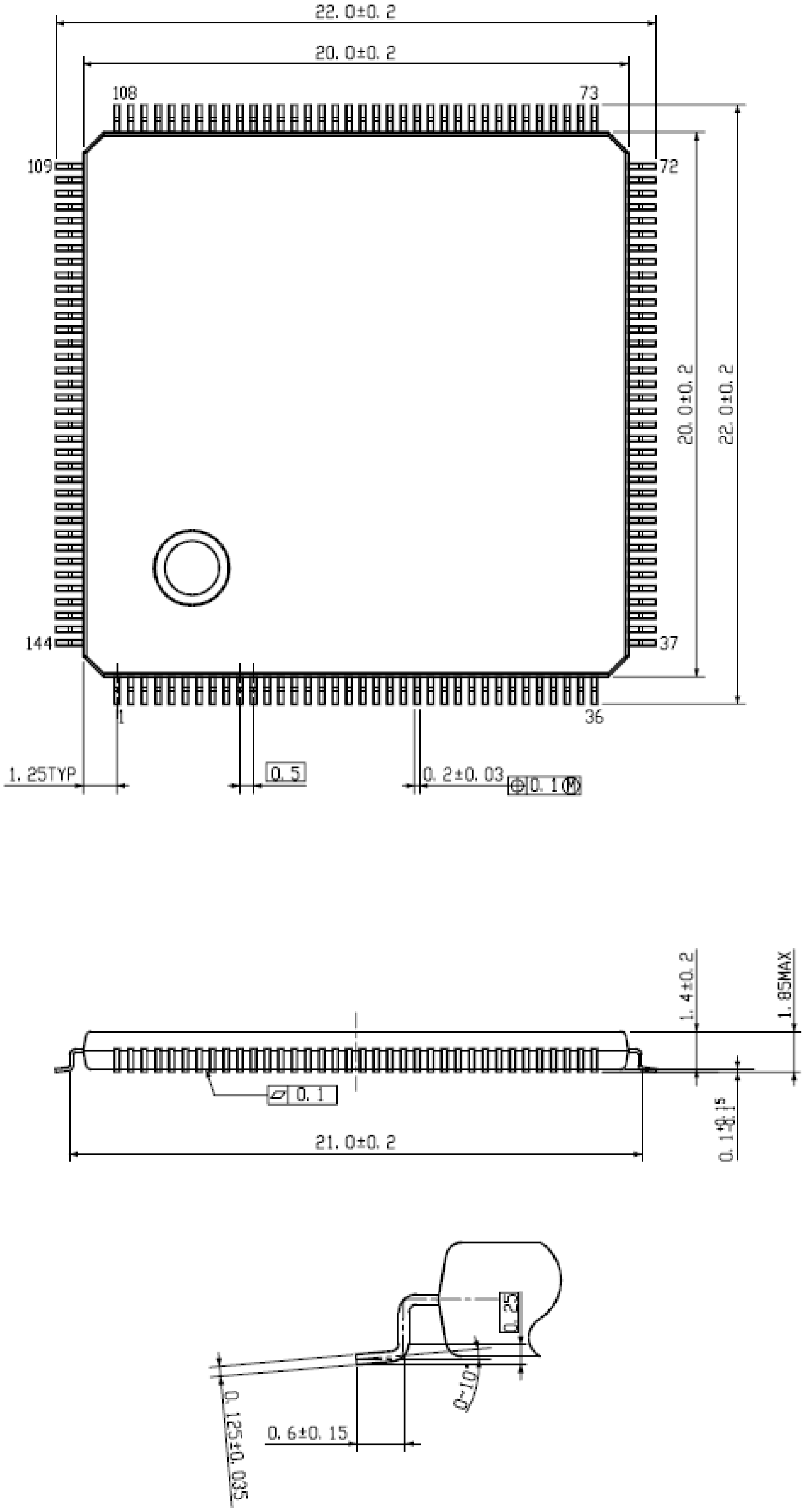
[34] USB Misc

base address = 0x4000_5000

Register Name	Address (base+)	Type	Description
FrameLengthAdjustment	0x0304	R/W	SOF cycle time adjustment

7. Outside Dimensions

LQFP144-P-2020-0.50E


22.0±0.2
20.0±0.2
108
73
109
72
144
37
1
36
20.0±0.2
22.0±0.2
1.25TYP
0.5
0.2±0.03
⊕ 0.1 Ⓜ
1.4±0.2
1.85MAX
▱ 0.1
21.0±0.2
0.1+0.15 −0.1
0.25
0~10°
0.6±0.15
0.125±0.035