TOSHIBA

TOSHIBA Original RISC 32-Bit Microprocessor

ARM Core Family

TMPA901CMXBG

TOSHIBA CORPORATION

Semiconductor Company

- 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 as undefined. Write as zero.
[7:6]	SAMPLE76	R/W	0y00	Sample 設定 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, USB host 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 Microcontroller

TMPA901CMXBG

1. Overview and Features

TMPA901CM is a 32-bit RISC microcontroller with a built-in ARM9™_cpu core.
TMPA901CMXBG is a 177-pin BGA package product.
Features of the product are as follows:

(1) ARM926EJ™-S manufactured by ARM is used.
- Data cache: 16 Kbytes
- Instruction cache: 16 Kbytes

(2) Maximum operating frequency: 200 MHz(@0 to 70°C) / 150MHz(@-20 to 85°C)

(3) 7-layer multi bus system is used.
- Bus Master1: CPU data
- Bus Master2: CPU instruction
- Bus Master3: LCD controller
- Bus Master4: LCD data process accelerator
- Bus Master5: DMA controller 1
- Bus Master6: DMA controller 2
- Bus Master7: USB device controller

(4) Memory access
- Built-in RAM: 32 Kbytes (can be used as program, data, and display memory)
- Built-in ROM: 16 Kbytes (boot memory)
  It can be loaded to the built-in RAM from USB.
- 4 GB linear access space (effective space: approximately 1.7 GB)
- Separate bus system:

External address	24 bits: A0-A23
External data bus	16bit : D0-D15

(5) Memory controller
- Chip-select output: 2 channels
- Chip-select exclusive for DRAM: 1 channel
- Depending on the external pin selection, SDR (Single Data Rate)-type SDRAM and DDR (Double Data Rate) LVCMOS_I/O type SDRAM can be supported (SSTL_IO type DDR SDRAM cannot be supported).
- Support Asynchronous Static Memory, but Not support synchronous Static Memory.

(6) 16-bit timer
- 6 channels 16-bit timers including 2 channel timers with PWM function.

(7) Synchronous serial bus interface: 1 channel
- Supports SPI mode / Microwire mode

(8) I²C bus interface: 1channel

(9) UART: 2 channels
- Channel 0: supports TXD/RXD 2 wires UART/ supports IrDA1.0 mode.
- Channel 1: supports TXD/RXD/U1CTS 3 wires UART

(10) USB Device controller: 1 channel
- Supports high communication speed (480Mbps) (does not support Low Speed).
- Supports 4 endpoints.
  End-point 0: Control 64 bytes × 1- FIFO
  End-point 1: Bulk (Device → Host: IN transfer) 512 bytes × 2 -FIFO
  End-point 2: Bulk (Host → Device: OUT transfer) 512 bytes × 2- FIFO
  End-point 3: Interrupt 64 bytes × 1- FIFO

(11) USB Host controller: 1 channel
- Supports full communication speed (12Mbps) (does not support Low Speed)

(12) I^2S (Inter-IC Sound) interface: 2 channel
Channel 0 (for reception: 32-byte FIFO × 2)
Channel 1 (for transmission: 32-byte FIFO × 2)
Channel 0 and channel1 have common usage pins.

(13) LCD controller
- Supports 800 × 480 pixel size.
- Supports TFT/STN panels.
- For STN panels, 4/15/64 monochrome tones and 256/3375 color tones are supported.
- For TFT panels, 16-bit color is supported.

(14) LCD data process accelerator
- Scaling function (expansion/reduction)
- Filtering function (bi-cubic convolution)
- Image blending function (supports font blending)

(15) RTC (real-time clock)

(16) Melody/Alarm generator
- Supports output of 8 alarm sound patterns.

(17) Key-on wake up (key-input interrupt)

(18) 10-bit AD converter (with a built-in sample-and-hold circuit): 4 channels

(19) Supports touch-screen interfaces
- Since a low-resistance switch is built in to the product, external components for horizontal/vertical switching can be deleted.

(20) Watchdog timer

(21) Oscillation Frequency Detector
- Fail Safe mode for High frequency oscillation

(22) Interrupt function: 21 types
- External 3types (7 pins): External Interrupt(edge: rise and fall, level: High and Low) And Key In
- Internal : 18 types : 16bit Timer × 3, RTC × 1, A/D converter × 1, LCDC × 1, NANDFC × 1, UART × 2, SSP × 1, I^2C × 1, USB Device × 1, USB Host × 1, I^2S × 1, LCDDA × 1, DMAC × 2, and WDT × 1

(23) I/O port: 43 pins

(24) DMA controller: 8 channels

(25) NAND-flash memory interface: 2 channels
- Easy connection to NAND-flash memory.
- Supports both 2LC (2 values) and 4LC (4 values) types.
- Supports 8-bit data bus and 512/2048-byte page size.
- Built-in Reed Solomon operational circuit can correct 4 addresses and detect errors in more than 5 addresses.

(26) Standby function
- Status of each pin in standby mode can be set bit-by-bit.
- Built-in power management circuit (PMC) to prevent leakage current.

(27) Clock control function
- Two blocks of built-in clock multiple circuit (PLL) enables an external 10 to 25 MHz Oscillator to supply various clocks as below:

@ 0 to 70°C : USB Device clock frequency of 480 MHz and clock frequency of 200 MHz to The CPU (CPU clock frequency is 192 MHz when USB is in use).

@−20 to 85°C : USB Device clock frequency of 480MHz and clock frequency of 150 MHz to The CPU (CPU clock frequency is 144 MHz when USB is in use).

- Clock gear function: A high-frequency clock can be changed within the range of fc to fc/8.
- Real Time Clock (fs = 32.768 kHz)

(28) Oscillation Frequency Detector (OFD) function

(29) Operating voltage
- Internal DVCC1A and DVCC1B = 1.5V±0.1V
- High-frequency oscillator and power supply for PLL, DVCC1C = 1.5V±0.1V
- External I/O DVCCM for memory = 3.0V to 3.6V or 1.8V±0.1V
- LCD and General external I/O DVCC3IO = 3.0V to 3.6V
- External I/O AVCC3AD for AD converter = 3.0V to 3.6V
- External I/O AVDD3T/C for USB Device2.0 = 3.15V to 3.45V
- External I/O AVCC3H for USB Host = 3.0V to 3.6V

(30) DSU (JTAG) function
- JTAG supports of the ARM9 core.

(31) Package
- 177-pin FBGA : P-FBGA177-1313-0.8C4

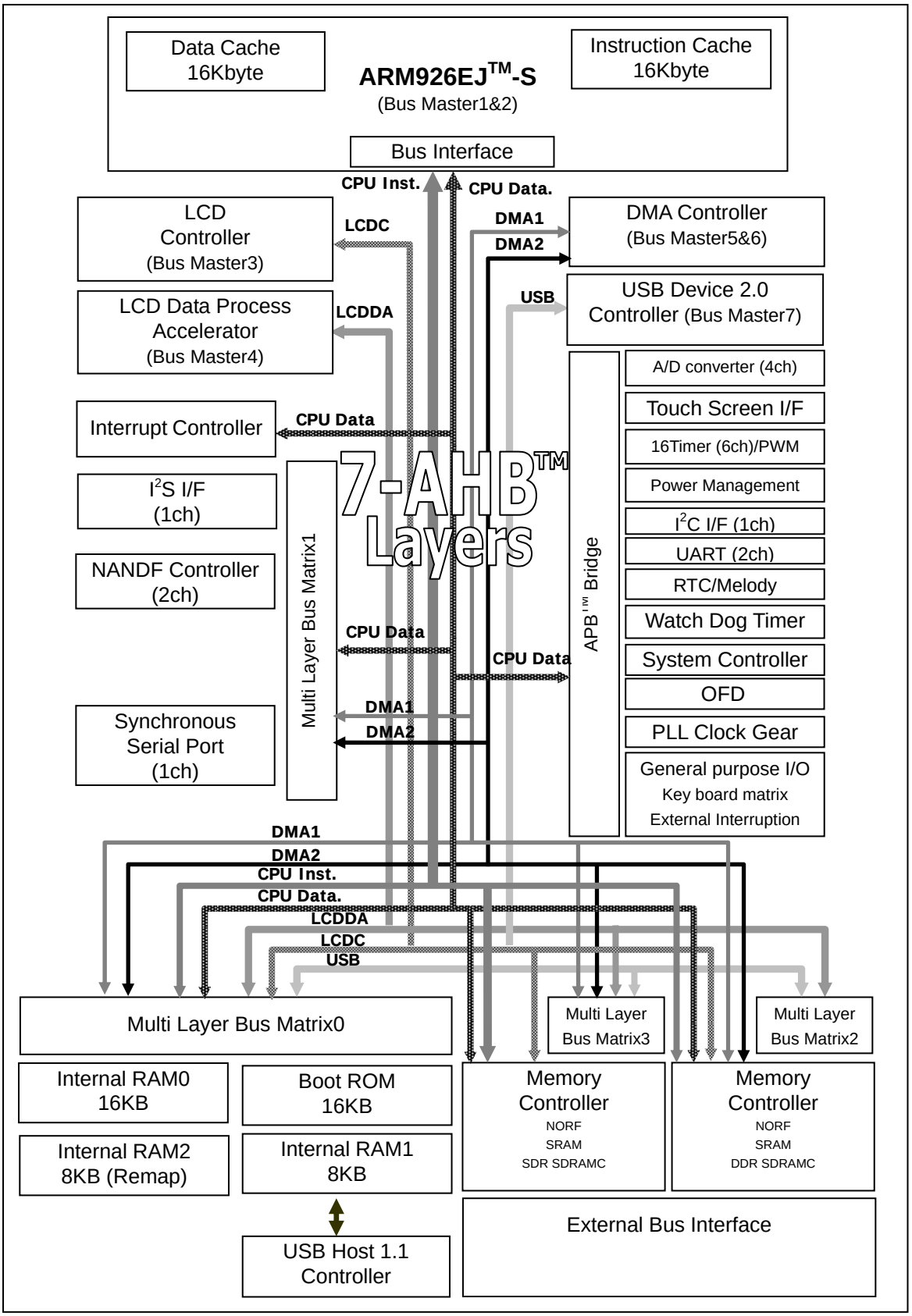


Figure 1.1 TMPA901CM block diagram

2. Pin Configuration and Functions

This section provides a TMPA901CM, names of I/O pins, and brief description of their functions.

2.1 Pin configuration diagram (Top View)

Figure 2.1.1 shows the TMPA901CM pin configuration(Package: FBGA177-P-1313-0.8C4)
About the detail pin configuration, please refer to Table 2.1.1 of next page

TMPA901CM
TOP VIEW
(Perspective view from the top)

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
E1	E2	E3	E4	E5							E12	E13	E14	E15
F1	F2	F3	F4								F12	F13	F14	F15
G1	G2	G3	G4								G12	G13	G14	G15
H1	H2	H3	H4								H12	H13	H14	H15
J1	J2	J3	J4								J12	J13	J14	J15
K1	K2	K3	K4								K12	K13	K14	K15
L1	L2	L3	L4								L12	L13	L14	L15
M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13	N14	N15
P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15

Figure 2.1.1 Pin configuration diagram

Table 2.1.1 Pin configuration

	1	2	3	4	5	6	7	8
A	A1 DVSSCOM	A2 SM3/XT2	A3 SM2/XT1	A4 PU3/NDD3/LD3	A5 PU2/NDD2/LD2	A6 PU1/NDD1/LD1	A7 PU0/NDD0/LD0	A8 SE5/A5
B	B1 SP0/TCK	B2 PC2/PWE	B3 PC3/MLDALM/PWM1OUT	B4 PU7/NDD7/LD7	B5 PU6/NDD6/LD6	B6 PU5/NDD5/LD5	B7 PU4/NDD4/LD4	B8 SF3/A11
C	C1 SP4/RTCK	C2 SP1/TMS	C3 PC4/FSOUT/PWM3OUT	C4 PV3/NDCLE/LD11	C5 PV2/NDALE/LD10	C6 PV1/NDWEn/LD9	C7 PV0/NDREn/LD8	C8 SG0/A16
D	D1 SP5/TDO	D2 SP2/TDI	D3 PC6/I2C0CL/USBPON	D4 PV7/LD15	D5 PV6/NDRB/LD14	D6 PV5/NDCE1n/LD13	D7 PV4/NDCE0n/LD12	D8 SG4/A20
E	E1 DVCC3IO	E2 SP3/TRSTn	E3 PC7/I2C0DA/INT9	E4 DVCC3IO	E5 DVSSCOM			
F	F1 DVCC1B	F2 DVCC3IO	F3 DVCC3IO	F4 DVCC3IO				
G	G1 DVSSCOM	G2 DVSSCOM	G3 DVSSCOM	G4 DVSSCOM				
H	H1 DVCC1A	H2 DVCC1A	H3 DVCC1A	H4 DVCC1A				
J	J1 AVCC3AD	J2 VREFH	J3 VREFL	J4 DVCC1B				
K	K1 PD4/AN4/MX	K2 PD5/AN5/MY	K3 AVSS3AD	K4 DVCC3IO				
L	L1 PD6/INTA(TSI)/AN6	L2 PD7/INTB/AN7	L3 DVCC3IO	L4 SM6/AM0				
M	M1 DVCC3IO	M2 DVCC3IO	M3 PA0/KI0	M4 PA2/KI2	M5 DVSSCOM	M6 AVSS3C	M7 DVCC1A	M8 DVCC3IO
N	N1 SM4/RESETn	N2 PN0/U0TXD/SIR0OUT	N3 PA1/KI1	N4 PA3/KI3	N5 DVSSCOM	N6 AVDD3C	N7 AVDD3T1	N8 AVDD3T0
P	P1 PN1/U0RXD/SIR0IN	P2 SM7/AM1	P3 DVCC1C	P4 DVSS1C	P5 DVSSCOM	P6 SR3/REXT	P7 AVSS3T2	P8 AVSS3T1
R	R1 DVSSCOM	R2 SM0/X1	R3 SM1/X2	R4 DVCC1C	R5 SR4/VSENS	R6 AVSS3T3	R7 SR1/DDM	R8 SR0/DDP
	1	2	3	4	5	6	7	8

Table2.1.2 Pin configuration

9	10	11	12	13	14	15	
A9 SE4/A4	A10 SE3/A3	A11 SE2/A2	A12 SE1/A1	A13 SE0/A0	A14 SL2/DMCAP	A15 DVSSCOM	A
B9 SG7/A23	B10 SF2/A10	B11 SF1/A9	B12 SF0/A8	B13 SE7/A7	B14 SE6/A6	B15 SL1/DMCD CLKN	B
C9 SF7/A15	C10 SG6/A22	C11 SF6/A14	C12 SF5/A13	C13 SF4/A12	C14 SK0/DMCSDQM0/D MCDDM0	C15 SL0/DMCD CLKP/DMC SCLK	C
D9 SG3/A19	D10 SG2/A18	D11 SG5/A21	D12 SG1/A17	D13 SK4/SMCWEn	D14 SK1/DMCSDQM1/D MCDDM1	D15 SL6/DMCCL KIN	D
			E12 SK5/SMCBE1n	E13 SJ5/DMCBA1	E14 SB7/D15	E15 SB6/D14	E
			F12 SJ6/DMCCKE	F13 SJ4/DMCBA0	F14 SB5/D13	F15 SB4/D12	F
			G12 DVCCM	G13 SJ3/DMCCASn	G14 SB3/D11	G15 SB2/D10	G
			H12 DVCCM	H13 SJ2/DMCRASn	H14 SB1/D9	H15 SB0/D8	H
			J12 DVCCM	J13 SJ1/DMCWEn	J14 SL5/DMCDDQS1	J15 SL4/DMCD DQS0	J
			K12 DVCC1A	K13 SJ0/SMCOEn	K14 SA7/D7	K15 SA6/D6	K
			L12 DVCC1B	L13 SH7/DMCCSn	L14 SA5/D5	L15 SA4/D4	L
M9 DVCC3IO	M10 SN2/SELJTAG	M11 AVCC3H	M12 SN1/SELDVCCM	M13 SH4/SMCCS1n	M14 SA3/D3	M15 SA2/D2	M
N9 PB2/KO2/LC LFP	N10 PB1/KO1/LCLAC	N11 PT2/SP0DO/I 2S0DATI	N12 PT4/U1TXD/USBPON	N13 SH3/SMCCS0n	N14 SA1/D1	N15 SA0/D0	N
P9 SN0/SELME MC	P10 PB0/KO0/LCLCP	P11 PT6/U1CTSn/ I2S0DATO	P12 PT1/SP0CLK/I2S0CL K	P13 PT0/SP0FSS/I2S 0WS	P14 PT3/SP0DI/I2S0MC LK	P15 SH2/SMCB E0n	P
R9 AVSS3T0	R10 PB3/KO3/LCLLP	R11 PT7/X1USB	R12 PT5/U1RXD/USBOC	R13 SN7/HDM	R14 SN6/HDP	R15 DVSSCOM	R
9	10	11	12	13	14	15	

2.2 Pin Names and Functions

The names and functions of I/O pins are shown below.

Pins associated with memory are switched to either of two types of MPMC (MPMC0/1) depending on the status of the external pin "SELMEMC".

Table 2.2.1 Pin names and functions (1/6)

Pin name	Number of pins	Input/Output	Function	Remarks
SA0 to SA7 D0 to D7	8	– Input/Output	– Data: Data bus D0 to D7	– For both MPMC0 and MPMC1
SB0 to SB7 D8 to D15	8	– Input/Output	– Data: Data bus D8 to D15	– For both MPMC0 and MPMC1
SE0 to SE7 A0 to A7	8	– Output	– Address: Address bus A0 to A7	– For both MPMC0 and MPMC1
SF0 to SF7 A8 to A15	8	– Output	– Address: Address bus A8 to A15	– For both MPMC0 and MPMC1
SG0 to SG7 A16 to A23	8	– Output	– Address: Address bus A16 to A23	– For both MPMC0 and MPMC1
SH2 SMCBE0n	1	– Output	– Byte enable signal (D0 to D7) for NORF/SRAM/MROM	– For both MPMC0 and MPMC1
SK5 SMCBE1n	1	– Output	– Byte enable signal (D8 to D15) for NORF/SRAM/MROM	– For both MPMC0 and MPMC1
SH3 SMCCS0n	1	– Output	– Chip select signal 0 for NORF/SRAM/MROM	– For both MPMC0 and MPMC1
SH4 SMCCS1n	1	– Output	– Chip select signal 1 for NORF/SRAM/MROM	– For both MPMC0 and MPMC1
SH7 DMCCSn	1	– Output Output	– Write-enable signal for SDR_SDRAM Write-enable signal for DDR_SDRAM	 When using MPMC0 When using MPMC1
SJ0 SMCOEn	1	– Output	– Out-enable signal for NORF/SRAM/MROM	– For both MPMC0 and MPMC1
SJ1 DMCWEn	1	– Output Output	– Write-enable signal for SDR_SDRAM Write-enable signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SJ2 DMCRASn	1	– Output Output	– Row address strobe signal for SDR_SDRAM Row address strobe signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SJ3 DMCCASn	1	– Output Output	– Column address strobe signal for SDR_SDRAM Column address strobe signal for DDR_SDRAM	– When using MPMC0 When using MPMC1

Note: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Table2.2.1 Pin names and functions (2/6)

Pin name	Number of pins	Input/Output	Function	Remarks
SJ4 DMCBA0	1	– Output Output	– BANK0 strobe signal for SDR_SDRAM BANK0 strobe signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SJ5 DMCBA1	1	– Output Output	– BANK1 strobe signal for SDR_SDRAM BANK1 address strobe signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SJ6 DMCCKE	1	– Output Output	– Clock-enable signal for SDR_SDRAM Clock-enable signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SK0 DMCSDQM0 DMCDDM0	1	– Output Output	– Byte enable signal (D0 to D7) for SDR_SDRAM Data mask signal (D0 to D7) for DDR_SDRAM	– When using MPMC0 When using MPMC1
SK1 DMCSDQM1 DMCDDM1	1	– Output Output	– Byte enable signal (D8 to D15) for SDR_SDRAM Data mask signal (D8 to D15) for DDR_SDRAM	– When using MPMC0 When using MPMC1
SK4 SMCWEn	1	– Output	– Write-enable signal for NORF/SRAM/MROM	– For both MPMC0 and MPMC1
SL0 DMCSCLK DMCDCLKP	1	– Output Output	– Clock signal for SDR_SDRAM Positive phase clock signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SL1 – DMCDCLKN	1	– – Output	– Not used Negative phase clock signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SL2 DMCAP	1	– Output Output	– Address/Precharge signal for SDR_SDRAM Address/Precharge signal for DDR_SDRAM	– When using MPMC0 When using MPMC1
SL4 – DMCDDQS0	1	– – Input/Output	– Not used Data strobe signal (D0 to D7) for DDR_SDRAM	– When using MPMC0 When using MPMC1
SL5 – DMCDDQS1	1	– – Input/Output	– Not used Data strobe signal (D8 to D15) for DDR_SDRAM	– When using MPMC0 When using MPMC1
SL6 DMCCLKIN	1	– Input	– FB clock for SDR/DDR_SDRAM	– For both MPMC0 and MPMC1

Note: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Table2.2.1 Pin names and functions (3/6)

Pin name	Number of pins	Input/Output	Function	Remarks
SM0 X1	1	– Input	– High-frequency oscillator connecting input pin	
SM1 X2	1	– Output	– High-frequency oscillator connecting output pin	
SM2 XT1	1	– Input	– Low-frequency oscillator connecting input pin	
SM3 XT2	1	– Output	– Low-frequency oscillator connecting output pin	
SM4 RESETn	1	– Input	– Reset: Initializes TMPA901CM (with Schmitt input and pull-up resistor)	
SM6 to SM7 AM0 to AM1	2	– Input	– Startup mode input pins	
SN0 SELMEMC	1	– Input	– Memory controller selection pin	
SN1 SELDVCCM	1	– Input	– Memory-related operating voltage selection pin	
SN2 SELJTAG	1	– Input	– Boundary scan switching pin	
SN6 HDP	1	– Input/Output	– D+ for USB Host Data	
SN7 HDM	1	– Input/Output	– D- for USB Host Data	
SP0 TCK	1	– Input	– Clock pin for JTAG	
SP1 TMS	1	– Input	– Pin for JTAG	
SP2 TDI	1	– Input	– Data input pin for JTAG	
SP3 TRSTn	1	– Input	– Reset pin for JTAG	
SP4 RTCK	1	– Output	– Clock output pin for JTAG	
SP5 TDO	1	– Output	– Data output pin for JTAG	
SR0 DDP	1	– Input/Output	– USB Device pin (D+)	
SR1 DDM	1	– Input/Output	– USB Device pin (D-)	
SR3 REXT	1	– Input	– Connect to the VSENS pin at 12 kΩ	
SR4 VSENS	1	– Input	– Connect to the REXT pin at 12 kΩ	

Note: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are differentfrom general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Table2.2.1 Pin names and functions (4/6)

Pin name	Number of pins	Input/Output	Function	Remarks
PA0~PA3 KI0~KI3	4	Input Input	Port A0 to A3: Input ports Key input KI0 to KI3: Pins for key-on wake up 0 to 3 (with Schmitt input and pull-up resistor)	
PB0 KO0 LCLCP	1	Output Output Output	Port B0: Output ports Key output KO0 : Key out pins (open-drain can be set) LCD driver output pin	
PB1 KO1 LCLAC	1	Output Output Output	Port B1: Output ports Key output KO1 : Key out pins (open-drain can be set) LCD driver output pin	
PB2 KO2 LCLFP	1	Output Output Output	Port B2: Output ports Key output KO2 : Key out pins (open-drain can be set) LCD driver output pin	
PB3 KO3 LCLLP	1	Output Output Output	Port B3: Output ports Key output KO3 : Key out pins (open-drain can be set) LCD driver output pin	
PC2 PWE	1	Output Output	Port C2: Output port External power source control output: This pin controls ON/OFF of the external power source. The "H" level is output during regular operations, and the "L" level is output during standby mode.	
PC3 MLDALM PWM0OUT	1	Output Output Output	Port C3: Output port Melody alarm output pin Timer PWM out port	
PC4 FSOUT PWM2OUT	1	Output Output Output	Port C4: Output port Low-frequency output clock pin Timer PWM out port	
PC6 I2C0CL	1	Input/Output Input/Output	Port C6: I/O port I2C clock I/O	
PC7 I2C0DA INT9	1	Input/Output Input/Output Input	Port C7: I/O port I2C data I/O Interrupt request pin9: an interrupt request pin that can program the rising/falling edge	
PD4 AN4 MX	1	Input Input Output	Port D4: Input port Analog input 4: AD converter input pin X-minus: X-connecting pin for touch panel	
PD5 AN5 MY	1	Input Input Output	Port D5: Input port Analog input 5: AD converter input pin Y-minus: Y-connecting pin for touch panel	
PD6 AN6 PX INTA(INTTSI)	1	Input Input Output Input	Port D6: Input port Analog input 6: AD converter input pin X-plus: X-connecting pin for touch panel Interrupt request pin A: an interrupt request pin that can program the rising/falling edge	
PD7 AN7 PY INTB	1	Input Input Output Input	Port D7: Input port Analog input 7: AD converter input pin Y-plus: Y-connecting pin for touch panel Interrupt request pin B: an interrupt request pin that can program the rising/falling edge	

Table2.2.1 Pin names and functions (5/6)

Pin name	Number of pins	Input/Output	Function	Remarks
PN0	1	Input/Output	Port N0: I/O port	
U0TXD		Output	UART function 0 transmission data	
SIR0OUT		Output	Data output pin for IrDA1.0	
PN1	1	Input/Output	Port N1: I/O port	
U0RXD		Input	UART function 0 receive data	
SIR0IN		Input	Data input pin for IrDA1.0	
PT0	1	Input/Output	Port T0: I/O port	
SP0FSS		Input/Output	FSS pin for SSP0	
I2S0WS		Input/Output	I2S0 word select Input/output	
PT1	1	Input/Output	Port T1: I/O port	
SP0CLK		Input/Output	Clock pin for SSP0	
I2S0CLK		Input/Output	I2S0 serial clock Input/output	
PT2	1	Input/Output	Port T2: I/O port	
SP0DO		Output	Data output pin for SSP0	
I2S0DATI		Input	I2S0 receive serial data input	
PT3	1	Input/Output	Port T3: I/O port	
SP0DI		Input	Data input pin for SSP0	
I2S0MCLK		Output	I2S0 master clock output for receive circuit	

Table2.2.1 Pin names and functions (6/6)

Pin name	Number of pins	Input/Output	Function	Remarks
PT4	1	Input/Output	Port T4: I/O port	
U1TXD		Output	UART function 1 transmission data	
USBPON		output	Power On Enable for USB Host	
PT5	1	Input/Output	Port T5: I/O port	
U1RXD		Input	UART function 1 receive data	
USBOCn		Input	Over Current detect for USB Host	
PT6	1	Input/Output	Port T6: I/O port	
U1CTSn		Output	UART1 handshake (Transmitter Enable)	
I2S1DATO		Output	I2S transmission serial data output	
PT7	1	Input/Output	Port T7: I/O port	
X1USB		Input	Clock input pin for USB	
PU0 to PU7	8	Input/Output	Port U0 to Port U7 : I/O port	
NDD0 to NDD7		Input/Output	Data buses for NANDF memory	
LD0 to LD7		Input/Output	Data buses for LCD driver	
PV0	1	Input/Output	Port V0: I/O port	
NDREn		Output	Read enable for NAND-Flash	
LD8		Output	Data bus for LCD drive	
PV1	1	Input/Output	Port V1: I/O port	
NDWEn		Output	Write enable for NAND-Flash	
LD9		Output	Data bus for LCD driver	
PV2	1	Input/Output	Port V2: I/O port	
NDALE		Output	Address latch enable for NAND-Flash	
LD10		Output	Data bus for LCD driver	
PV3	1	Input/Output	Port V3: I/O port	
NDCLE		Output	Command latch enable for NAND-Flash	
LD11		Output	Data bus for LCD driver	
PV4	1	Input/Output	Port V4: I/O port	
NDCE0n		Output	NAND-Flash0 chip select	
LD12		Output	Data bus for LCD driver	
PV5	1	Input/Output	Port V5: I/O port	
NDCE1n		Output	NAND-Flash1 chip select	
LD13		Output	Data bus for LCD driver	
PV6	1	Input/Output	Port V6: I/O port	
NDRB		Input	NAND-Flash Ready(1)/Busy(0) input	
LD14		Output	Data bus for LCD driver	
PV7	1	Input/Output	Port V7: I/O port	
LD15		Output	Data bus for LCD driver	

Pin Name	Number of pins	Power pins	Function	Remarks
DVCC1A	6	Power supply	VCC power supply for the main internal area	
DVCC1B	3	Power supply	VCC power supply for the internal B/U area	
DVCC1C	2	Power supply	VCC power supply for high-frequency clock/PLL circuit	
DVSS1C	1	Power supply	VSS power supply for high-frequency clock/PLL circuit	
DVCC3IO	11	Power supply	VCC power supply for external I/O (general and LCD)	
DVCCM	3	Power supply	VCC power supply for external I/O (for memory)	
AVCC3AD	1	Power supply	VCC power supply for external I/O (A/DC)	
AVSS3AD	1	Power supply	VSS power supply for external I/O (A/DC)	
VREFH	1	Input	Reference voltage for A/D converter	
VREFL	1	Input	Reference voltage for A/D converter	
AVDD3Tx	2	Power supply	VDD power supply for external I/O (USB Device)	
AVSS3Tx	4	Power supply	VSS power supply for external I/O (USB Device)	
AVDD3C	1	Power supply	VDD power supply for external I/O (USB Device)	
AVSS3C	1	Power supply	VSS power supply for external I/O (USB Device)	
AVCC3H	1	Power supply	VCC power supply for external I/O (USB Host)	
DVSSCOM	12	Power supply	Shared VSS power supply (GND)	

Pin Functions and Initial Values Arranged by Type of Power Supply - 1 (DVCCM)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/Down	Input Buffer	Initial value after reset function/pin state
DVCCM	SA0 to SA7	D0 to D7	−	−	ON	D0 to D7 / Hz*
	SB0 to SB7	D8 to D15	−	−	ON	D8 to D15 / Hz*
	SE0 to SE7	A0 to A7	−	−	−	Address out / “L” output
	SF0 to SF7	A8 to A15	−	−	−	Address out / “L” output
	SG0 to SG7	A16 to A23	−	−	−	Address out / “L” output
	SH2	SMCBE0n	−	−	−	SMCBE0n out / “H” output
	SK5	SMCBE1n	−	−	−	SMCBE1n out / “H” output
	SH3	SMCCS0n	−	−	−	SMCCS0n out / “H” output
	SH4	SMCCS1n	−	−	−	SMCCS1n out / “H” output
	SH7	DMCCSn	−	−	−	DMCCSn out / “H” output
	SJ0	SMCOEn	−	−	−	SMCOEn out / “H” output
	SJ1	DMCWEn	−	−	−	DMCWEn out / “H” output
	SJ2	DMCRASn	−	−	−	DMCRASn out / “H” output
	SJ3	DMCCASn	−	−	−	DMCCASn out / “H” output
	SJ4	DMCBA0	−	−	−	DMCBA0n out / “L” output
	SJ5	DMCBA1	−	−	−	DMCBA1n out / “L” output
	SJ6	DMCCKE	−	−	−	DMCCKEn out / “H” output
	SK0	DMCSDQM0	DMCDDM0	−	−	When SELMEMC = 0 DMCSDQM0 out / “L” output When SELMEMC = 1 DMCDDM0 out / “L” output
	SK1	DMCSDQM1	DMCDDM1	−	−	When SELMEMC = 0 DMCSDQM1 out / “L” output When SELMEMC = 1 DMCDDM1 out / “L” output
	SK4	SMCWEn	−	−	−	SMCWEn out / “H” output

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally. The data bus pins (SA0-SA7, SB0-SB7, SC0-SC7, SD0-SD7) are always enabled as inputs. These pins must be tied externally (pulled up/down, etc.) to prevent flow-through current.

Pin Functions and Initial Values Arranged by Type of Power Supply – 2 (DVCCM)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/down	Input buffer	Initial value after reset function/pin state
DVCCM	SL0	DMCSCLK	DMCDCLKP	–	–	When SELMEMC = 0 DMCSCLK out / CLK output When SELMEMC = 1 DMCDCLKP out / CLK output
	SL1	DMCDCLKN	–	–	–	When SELMEMC = 0 Invalid signal/ “H” output When SELMEMC = 1 DMCDCLKN out / Inverted CLK output
	SL2	DMCAP	–	–	–	DMCAP out / “L” output
	SL4	DMCDDQS0	–	–	ON	DMCDDQS0 / Hz*
	SL5	DMCDDQS1	–	–	ON	DMCDDQS1 / Hz*
	SL6	DMCCLKIN	–	–	ON	DMCCLKIN input / Hz

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally. When DDR SDRAM is used, the DQS signals (DMCDDQS0, DMCDDQS1) are always enabled as inputs. These pins must be tied externally (pulled up/down, etc.) to prevent flow-through current.

Pin Functions and Initial Values Arranged by Type of Power Supply – 3 (DVCC3IO)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/down	Input buffer	Initial state after reset function/pin state
DVCC3IO	SM2	XT1	–	–	–	Oscillating
	SM3	XT2	–	–	–	Oscillating
	SM4	RESETn	–	PU	ON	RESETn input / “H” output
	SM6	AM0	–	–	ON	AM0 input / Hz
	SM7	AM1	–	–	ON	AM1 input / Hz
	SN0	SELMEMC	–	–	ON	SELMEMC input / Hz
	SN1	SELDVCCM	–	–	ON	SELDVCCM input / Hz
	SN2	SELJTAG	–	–	ON	SELJTAG input / Hz
	SP0	TCK	–	–	ON	TCK input / Hz
	SP1	TMS	–	–	ON	TMS input/ Hz
	SP2	TDI	–	–	ON	TDI input / Hz
	SP3	TRSTn	–	–	ON	TRSTn input / Hz
	SP4	RTCK	–	–	–	RTCK out / CLK output
	SP5	TDO	–	–	–	TDO out / TDO output

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally. The data bus pins for NAND Flash memory (NDD0-NDD7) are disabled as inputs in the initial state.

Pin Functions and Initial Values Arranged by Type of Power Supply – 4 (DVCC3IO)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/down	Input buffer	Initial state after reset function/pin state
DVCC3IO	PA0 to PA4	KI0 to KI4	–	PU	ON	PA0 to PA4 input / “H” Output
	PB0 to PB3	KO0 to KO3	LCLxx	–	–	PB0 to PB3 out / “H” Output
	PC2	PWE	–	–	–	PWE out / “H” Output
	PC3	MLDALM	PWM0OUT	–	–	PC3 out / “H” Output
	PC4	FSOUT	PWM2OUT	–	–	PC4 out / ”L” Output
	PC6	I2C0CL	–	–	ON	PC6 input / Hz
	PC7	I2C0DA	INT9	–	ON	PC7 input / Hz
	PN0	U0TXD	SIR0OUT	–	ON	PN0 input / Hz
	PN1	U0RXD	SIR0IN	–	ON	PN1 input / Hz
	PT0	SP0FSS	I2S0WS	–	ON	PT0 input / Hz
	PT1	SP0CLK	I2S0CLK	–	ON	PT1 input / Hz
	PT2	SP0DO	I2S0DATI	–	ON	PT2 input / Hz
	PT3	SP0DI	I2S0MCLK	–	ON	PT3 input / Hz
	PT4	U1TXD	USBPON	–	ON	PT4 input / Hz
	PT5	U1RXD	USBOC	–	ON	PT5 input / Hz
	PT6	U1CTSn	I2S1DATO	–	ON	PT6 input / Hz
	PT7	X1USB	–	–	ON	PT7 input / Hz
	PU0 to PU7	NDD0 to NDD7	LD0 to LD7	–	ON	PU0 to PU7 / Hz
	PV0	NDREn	LD8	–	ON	PV0 input / Hz
	PV1	NDWEn	LD9	–	ON	PV1 input / Hz
	PV2	NDALE	LD10	–	ON	PV2 input / Hz
	PV3	NDCLE	LD11	–	ON	PV3 input / Hz
	PV4	NDCE0n	LD12	–	ON	PV4 input / Hz
	PV5	NDCE1n	LD13	–	ON	PV5 input / Hz
	PV6	NDRB	LD14	–	ON	PV6 input / Hz
	PV7	–	LD15	–	ON	PV7 input / Hz

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally. And when using external NAND Flash memory, the pins which are PV4(NDCE0n), PV5(NDCE1n) and so on should be processed by Pull-up or be fixed the level externally.

Pin Functions and Initial Values Arranged by Type of Power Supply – 5 (AVCC3AD)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Alternative function	Pull up/down	Input buffer	Initial state after reset function/pin state
AVDD3C/T	PD4	AN4	MX	–	–	OFF	AN4 input / Hz
	PD5	AN5	MY	–	–	OFF	AN5 input / Hz
	PD6	AN6	PX	INTA(INTTSI)	PD*	ON	AN6 input / Hz
	PD7	AN7	PY	INTB	–	ON	AN7 input / Hz

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally.

Note 3: The pull-down resistor for PD6 is disabled after reset.

Pin Functions and Initial Values Arranged by Type of Power Supply – 6 (USB Device)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/down	Input buffer	Initial value after reset function/pin state
AVDD3C/T	SR0	DDP	–	PD	ON	DP input / “L” output
	SR1	DDM	–	PD	ON	DM input / “L” output
	SR3	REXT	–	–	–	REXT input / Hz
	SR4	VSENS	–	–	–	VSENS input / Hz

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally. The DDP and DDM signals for USB contain a pull-down resistor in PHY.

Pin Functions and Initial Values Arranged by Type of Power Supply – 7 (USB Host)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/down	Input buffer	Initial value after reset function/pin state
AVCC3H	SN6	HDP	–	–	ON	HDP input / Hz
	SN7	HDM	–	–	ON	HDM input / Hz

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally. The HDP and HDM signals for USB contain a pull-down resistor in PHY.

Pin Functions and Initial Values Arranged by Type of Power Supply – 8 (OSC)

Power supply to be used	Typical pin name	Alternative function	Alternative function	Pull up/down	Input buffer	Initial value after reset function/pin state
DVCC1C	SM0	X1	–	–	–	Oscillating
	SM1	X2	–	–	–	Oscillating

Note 1: Pin names "SA0 through SA7, …, and SR0 through SR4" are symbols used for convenience and are different from general-purpose port functions "PA0 through PA7, …, and PV0 through PV7."

Note 2: When the “Input buffer” column shows “ON”, the pin is enabled as an input in the initial state. If necessary, the pin should be processed externally.

3. Operational Description

This chapter provides a brief description of the CPU circuitry of the TMPA901CM.

3.1 CPU

This section describes the basic operations of the CPU of the TMPA901CM for each block.

Note that this document provides only an overview of the CPU block. Please contact ARM Holdings for details of the operation.

The TMPA901CM has a built-in 32-bit RISC processor ARM926EJ-S™ manufactured by ARM.

The schematic diagram of the ARM926EJ-S™ core is shown below.

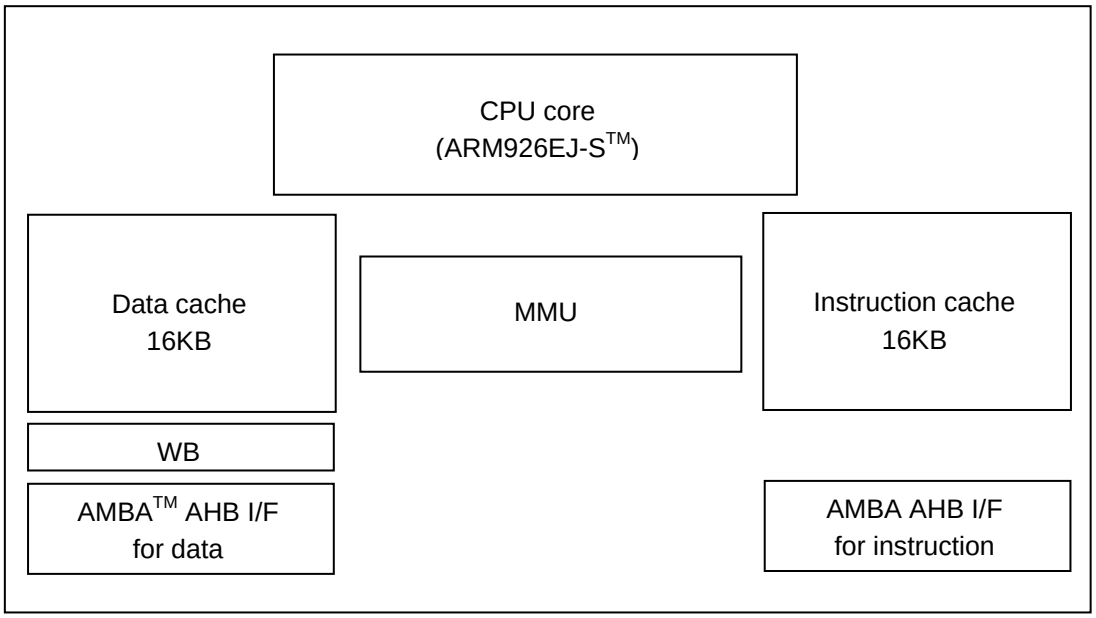


Figure 3.1.1 ARM926EJ-S™ core

The TMPA901CM does not contain the functions shown below.

1. Coprocessor I/F
2. Embedded ICE RT
3. TCM I/F
4. ETM9™ I/F
5. This CPU is under controlled by ARM Corporation and this version is r0p5.

3.1.1 Reset Operation

Before resetting the TMPA901CM, make sure that the power supply voltage is within the operating range, oscillation from the internal oscillator is stable at 20 system clock cycles (0.8 μs @ X1 = 25 MHz) at least, and the RESETn input pin is pulled Low.

When the TMPA901CM is reset, the PLL stops, the PLL output is unselected, and the clock gear is set to TOP (1/1).

The system clock therefore operates at 25 MHz (X1 = 25 MHz).

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

Reset the registers of the built-in I/O.

(Refer to the chapter on ports or on Pin, for reset values.)

Note 1: The IC has a built-in RAM, but its data may be lost due to the reset operation. Initialize data in the built-in RAM after the reset operation.

Note 2: Although this IC cuts off some of the power supplies (DVCC1A, DVCC1C, AVDD3Tx,AVDD3Cx,AVCC3H) to reduce standby current (PCM function), the reset operation may cause current penetration within the IC if it is executed while power to be cut off (DVCC1A, DVCC1C, AVDD3Tx, ADCC3Cx, AVCC3H) is not being supplied. Before executing the reset operation, make sure that the power supply to be cut off (DVCC1A, DVCC1C, AVDD3Tx, AVDD3Cx,AVCC3H) is sufficiently stable.

Although the original ARM926EJ-S™ allows selection of a vector location immediately after reset operation and endianness, they are already set as follows for this IC.

Endian	Boot vector
Little endian	0x00000000

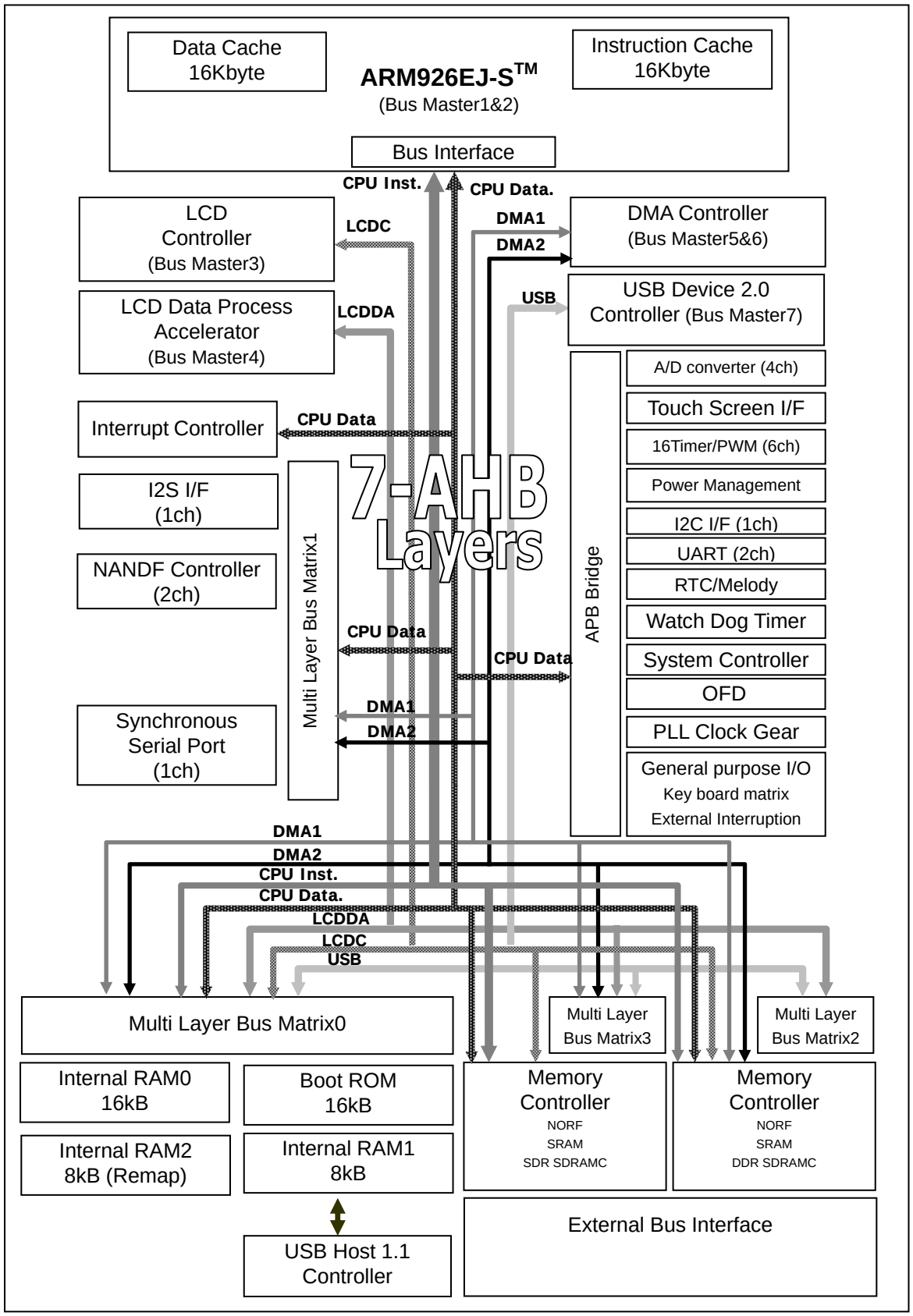
3.1.2 Exceptions

The TMPA901CM includes 7 types of exception, and each of them has privileged processing mode.

Exception	Address	Note
Reset	0x00000000	
Undefined instruction execution	0x00000004	
Software interrupt (SWI) instruction	0x00000008	It is used for operating system call.
Pre-fetch abort	0x0000000C	Instruction fetch memory abort
Data abort	0x00000010	Data access memory abort
IRQ	0x00000018	Normal interrupt
FIQ	0x0000001C	High-speed interrupt

3.1.3 Multilayer AHB

The TMP901CM uses a multilayer AHB bus system with 7 layers.



3.2 JTAG Interface

3.2.1 Overview

The TMPA901CMXBG provides a boundary-scan interface that is compatible with Joint Test Action Group (JTAG) specifications and uses the industry-standard JTAG protocol (IEEE Standard 1149.1・1990 <Includes IEEE Standard 1449.1a・1993>).

This chapter describes the JTAG interface, with the descriptions of boundary scan and the pins and signals used by the interface.

1) JTAG standard version

IEEE Standard 1149.1・1990 (Includes IEEE Standard 1149.1a・1993)

2) JTAG instructions

Standard instructions (BYPASS, SAMPLE/PRELOAD, EXTEST)

HIGHZ instruction

CLAMP instruction

3) IDCODE

Not available

4) Pins excluded from boundary scan register (BSR)

a) Oscillator circuit pins (SM0-3)

b) USB pins (SR0,SR1,SR3,SR4,SN6,SN7)

c) JTAG control pins (SN2, SP0-5)

d) Power supply/GND pins (including VREFH, REFL)

e) A/D pins (PD4-7)

f) Touch Panel PX,PY (PD6,PD7)

Note: PR2 pin is I/O pin. However, PR2 pin does not support the capture function by using SAMPLE/PRELOAD instructions because the BSR for the output is connected to the pin.

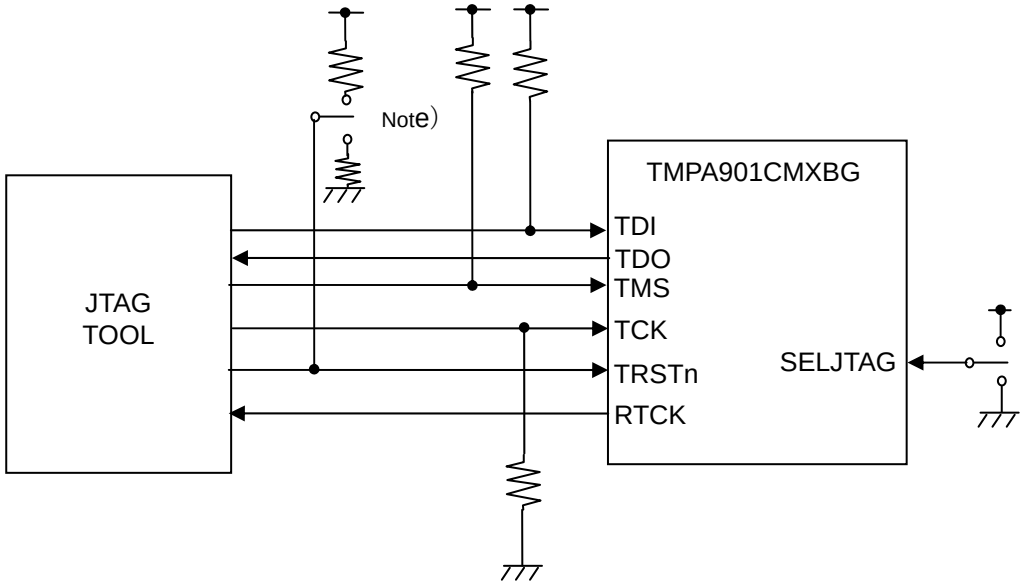
3.2.2 Signal Summary and Connection Example

The JTAG interface signals are listed below.

- TDI JTAG serial data input
- TDO JTAG serial data output
- TMS JTAG test mode select
- TCK JTAG serial clock input
- TRSTn JTAG test reset input
- RTCK JTAG test feedback serial clock output
- SELJTAG ICE/JTAG test select input (compatible with the Enable signal)
  0: ICE 1: JTAG

The TMPA901CM supports debugging by connecting the JTAG interface with a JTAG-compliant development tool.

For information about debugging, refer to the specification of the development tool used.



Note: In the case of not using JTAG Tool, fix the nTRST pin to GND.
In the case of using JTAG Tool, Once set the nTRST pin to "Low" level to reset the JTAG Circuits, and then translate to "High" level.
Pull-up resistance is built in some JTAG Tools, the value of external pull-up resistance need to be considered according to the JTAG Tools.

Mode Setting Pin	Operation mode
SELJTAG	
0	Set this pin to 0 except for Boundary Scan Mode. The TMPA901CM operates as regular Debug Mode. Note: Debugging is not available if the internal BOOT is carried out with AM1 = 1 and AM0 = 1.
1	The TMPA901CM operates in Boundary Scan Mode

Figure 3.2.1 Example of connection with a JTAG development tool

3.2.3 What Is Boundary Scan?

With the evolution of ever-denser integrated circuits (ICs), surface-mounted devices, double-sided component mounting on printed-circuit boards (PCBs), and set-in recesses, in-circuit tests that depend upon physical contact like the connection of the internal board and chip has become more and more difficult to use. The more ICs have become complex, the lager and more difficult the test program became.

As one of the solutions, *boundary-scan* circuits started to be developed. A boundary-scan circuit is a series of shift register cells placed between the pins and the internal circuitry of the IC to which the said pins are connected. Normally, these boundary-scan cells are bypassed; when the IC enters test mode, however, the scan cells can be directed by the test program to pass data along the shift register path and perform various diagnostic tests. To accomplish this, the tests use the six signals, TCK, TMS, TDI, TDO, RTCK and TRSTn.

The JTAG boundary-scan mechanism (hereinafter referred to as *JTAG mechanism* in the chapter) allows testing of the connections between the processor, the printed circuit board to which it is attached, and the other components on the circuit board.

The JTAG mechanism cannot test the processor alone.

3.2.4 JTAG Controller and Registers

The processor contains the following JTAG controller and registers:

Instruction register
Boundary scan register
Bypass register
Device identification register
Test Access Port (TAP) controller

JTAG basically operates to monitor the TMS input signal with the TAP controller state machine. When the monitoring starts, the TAP controller determines the test functionality to be implemented. This includes both loading the JTAG instruction register (IR) and beginning a serial data scan through a data register (DR), as shown in Table 3.2.1. As the data is scanned, the state of the TMS pin signals each new data word and indicates the end of the data stream. The data register is selected according to the contents of the instruction register.

3.2.5 Instruction Register

The JTAG instruction register includes four shift register-based cells. This register is used to select the test to be performed and/or the test data register to be accessed. As listed in Table 3.2.1, this instruction codes select either the boundary scan register or the bypass register.

Table 3.2.1 JTAG Instruction Register Bit Configuration

Instruction code (MSB to LSB)	Instruction	Selected data register
0000	EXTEST	Boundary scan register
0001	SAMPLE/PRELOAD	Boundary scan register
0100 to 1110	Reserved	Reserved
0010	HIGHZ	Bypass register
0011	CLAMP	Bypass register
1111	BYPASS	Bypass register

Figure 3.2.2 shows the format of the instruction register.

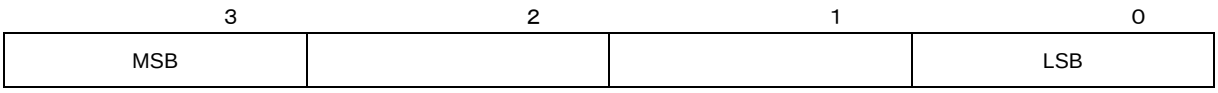


Figure 3.2.2 Instruction register

The instruction code is shifted out to the instruction register from the LSB.

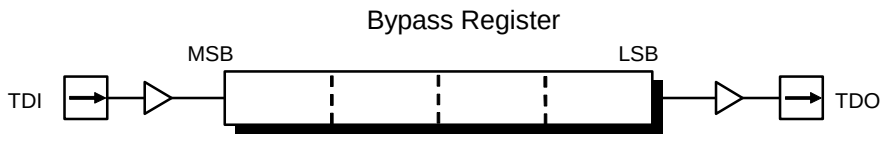


Figure 3.2.3 Instruction Register Shift Direction

The bypass register is 1 bit wide. When the TAP controller is in the Shift-DR (bypass) state, the data on the TDI pin is shifted into the bypass register, and the bypass register output shifts to the date out on the TDO output pin.

In essence, the bypass register is an alternative route which allows bypassing of board-level devices in the serial boundary-scan chain, which are not required for a specific test. The logical location of the bypass register in the boundary-scan chain is shown in Figure 3.2.4 .

Use of the bypass register speeds up access to the boundary scan register in the IC that remains active in the board-level test data path.

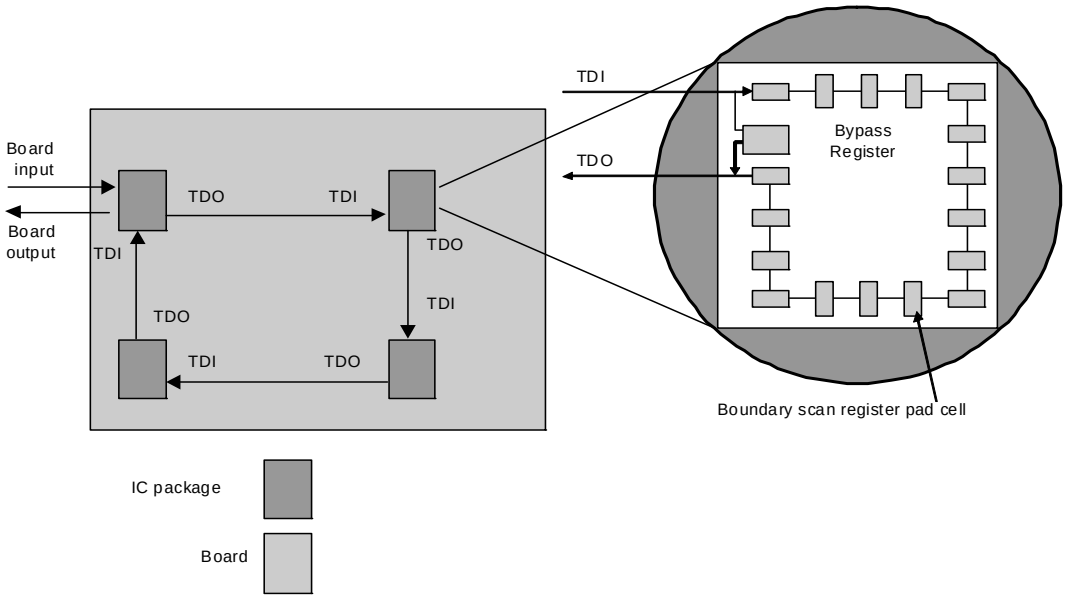


Figure 3.2.4 Bypass Register Operation

3.2.6 Boundary Scan Register

The boundary scan register provides all the inputs and outputs of the TMPA901CM processor except some analog outputs and control signals. The pins of the TMPA901CM allow any pattern to be driven by scanning the data into the boundary scan register in the Shift-DR state. Incoming data to the processor is examined by enabling the boundary scan register and shifting the data when the BSR is in the Capture-DR state.

The boundary scan register is a single, 231-bit-wide, shift register-based path containing cells connected to the input and output pads on the TMPA901CM.

The TDI input is loaded to the LSB of the boundary scan register. The MSB of the boundary scan register is shifted out on the TDO output.

3.2.7 Test Access Port (TAP)

The Test Access Port (TAP) consists of the five signal pins: TRSTn, TDI, TDO, TMS and TCK. These pins control a test by communicating the serial test data and instructions.

As Figure 3.2.5 shows, data is serially scanned into one of the three registers (instruction register, bypass register or boundary scan register) on the TDI pin, or it is scanned out from one of these three registers on the TDO pin.

The TMS input controls the state transitions of the main TAP controller state machine. The TCK input is a special test clock that allows serial JTAG data to be shifted synchronously, independent of any chip-specific or system clocks.

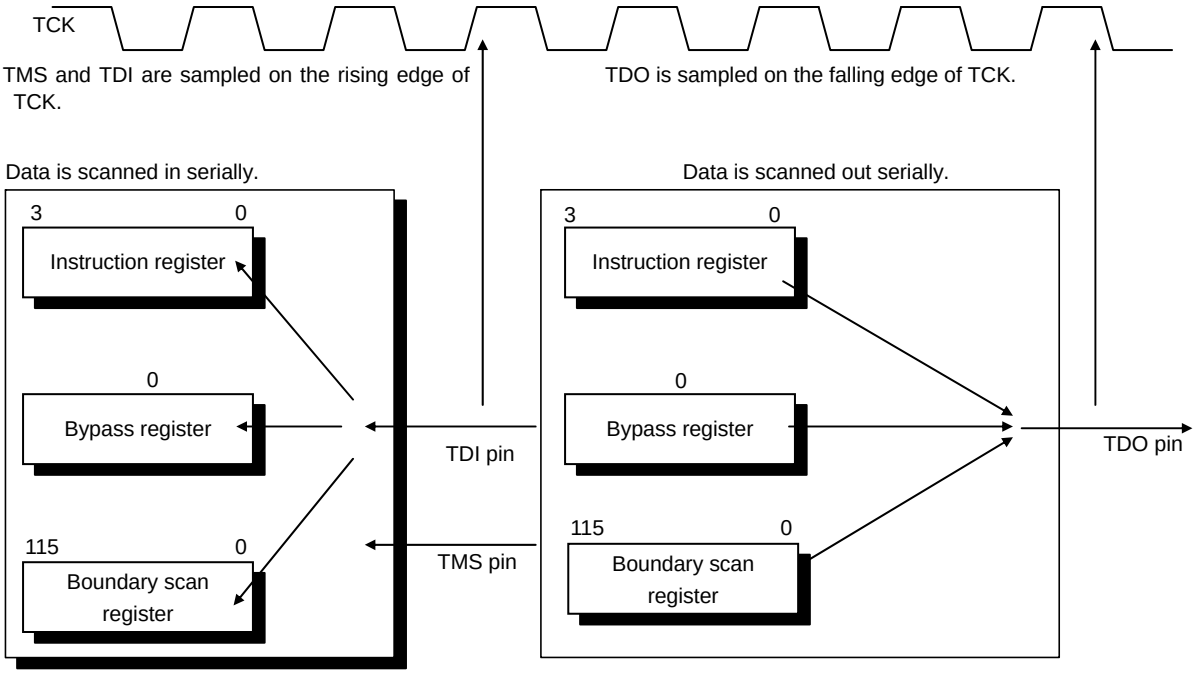


Figure 3.2.5 JTAG Test Access Port

Data on the TDI and TMS pins are sampled on the rising edge of the TCK input clock signal. Data on the TDO pin changes on the falling edge of the TCK clock signal.

3.2.8 TAP Controller

The processor incorporates the 16-state TAP controller stipulated in the IEEE JTAG specification.

3.2.9 Resetting the TAP Controller

The TAP controller state machine can be put into the Reset state by the following method.

Assertion of the TRSTn signal input (low) resets the TAP controller. After the processor reset state is released, keep the TMS input signal asserted through five consecutive rising edges of TCK input. Keeping TMS asserted maintains the Reset state.

3.2.10 State Transitions of the TAP Controller

The state transition diagram of the TAP controller is shown in Figure 3.2.6. Each arrow between states is labeled with a 1 or 0, indicating the logic value of TMS that must be set up before the rising edge of TCK to cause the transition.

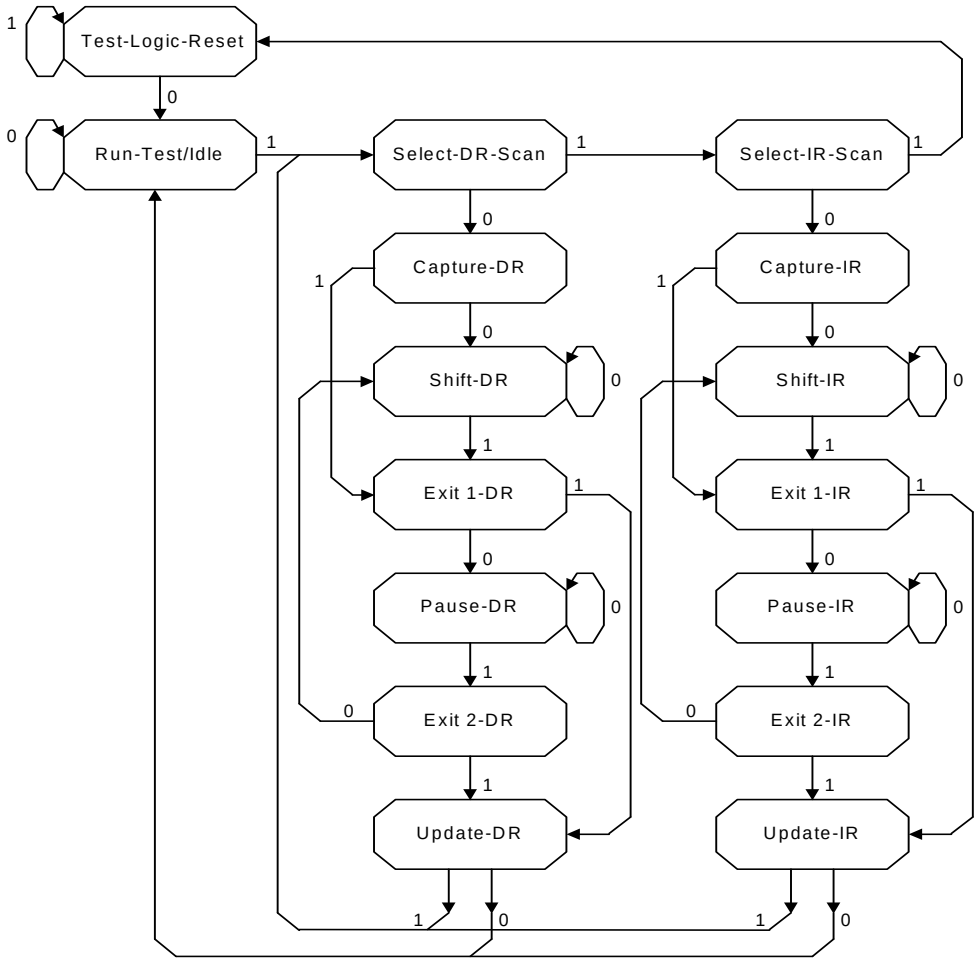


Figure 3.2.6 TAP Controller State Transition Diagram

The following paragraphs describe each of the controller states. The left column in Figure 3.2.6 is the data column, and the right column is the instruction column. The data column and instruction column reference the data register (DR) and the instruction register (IR), respectively.

- Test-Logic-Reset

  When the TAP controller is in the Reset state, the device identification register is selected by default. The MSB of the boundary scan register is cleared to 0 which disables the outputs.

  The TAP controller remains in this state while TMS is high. If TMS is held low while the TAP controller is in this state, then the controller moves to the Run-Test/Idle state.

- Run-Test/Idle

  In the Run-Test/Idle state, the IC is put in test mode only when certain instructions such as a built-in self test (BIST) instruction are present. For instructions that do not cause any activities in this state, all test data registers selected by the current instruction retain their previous states.

  The TAP controller remains in this state while TMS is held low. When TMS is held high, the controller moves to the Select-DR-Scan state.

- Select-DR-Scan

  This is a temporary controller state. Here, the IC does not execute any specific functions.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Capture-DR state. If TMS is held high, the controller moves to the Select-IR-Scan state.

- Select-IR-Scan

  This is a temporary controller state. Here, the IC does not execute any specific functions.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Capture-IR state. IF TMS is held high, the controller returns to the Test-Logic-Reset state.

- Capture-DR

  In this state, if the test data register selected by the current instruction has parallel inputs, then data is parallel-loaded into the shift portion of the data register. If the test data register does not have parallel inputs, or if data needs not be loaded into the selected data register, then the data register retains its previous state.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Shift-DR state. If TMS is held high, the controller moves to the Exit 1-DR state.

- Shift-DR

  In this controller state, the test data register connected between TDI and TDO shifts data out serially.

  When the TAP controller is in this state, then it remains in the Shift-DR state if TMS is held low, or moves to the Exit 1-DR state if TMS is held high.

- Exit 1-DR

  This is a temporary controller state.

  If TMS is held low when the TAP controller is in this state, the controller moves to the Pause-DR state. If TMS is held high, the controller moves to the Update-DR state.

- Pause-DR

  This state allows the shifting of the data register selected by the instruction register to be temporarily suspended. Both the instruction register and the data register retain their current states.

  When the TAP controller is in this state, then it remains in the Pause-DR state if TMS is held low, or moves to the Exit 2-DR state.

- Exit 2-DR

  This is a temporary controller state.

  When the TAP controller is in this state, it returns to the Shift-DR state if TMS is held low, or moves on to the Update-DR state if TMS is held high.

- Update-DR

  In this state, data is latched, on the rising edge of TCK, onto the parallel outputs of the data registers from the shift register path. The data held at the parallel output does not change while data is shifted in the associated shift register path.

  When the TAP controller is in this state, it moves to either the Run-Test/Idle state if TMS is held low, or the Select-DR-Scan state if TMS is held high.

- Capture-IR

  In this state, data is parallel-loaded into the instruction register. The data to be loaded is 0y0001. The Capture-IR state is used for testing the instruction register. Faults in the instruction register, if any, may be detected by shifting out the loaded data.

  When the TAP controller is in this state, it moves to either the Shift-IR state if TMS is held low, or the Exit 1-IR state if TMS is high.

- Shift-IR

  In this state, the instruction register is connected between TDI and TDO and shifts the captured data toward its serial output on the rising edge of TCK.

  When the TAP controller is in this state, it remains in the Shift-IR state if TMS is low, or moves to the Exit 1-IR state if TMS is high.

- Exit 1-IR

  This is a temporary controller state.

  When the TAP controller is in this state, it moves to either the Pause-IR state if TMS is held low, or the Update-IR state if TMS is held high.

- Pause-IR

  This state allows the shifting of the instruction register to be temporarily suspended. Both the instruction register and the data register retain their current states.

  When the TAP controller is in this state, it remains in the Pause-IR state if TMS is held low, or moves to the Exit 2-IR state if TMS is held high.

- Exit 2-IR

  This is a temporary controller state.

  When the TAP controller is in this state, it moves to either the Shift-IR state if TMS is held low, or the Update-IR state if TMS is held high.

- Update-IR

  This state allows the instruction previously shifted into the instruction register to be output in parallel on the rising edge of TCK. Then it becomes the current instruction, setting a new operational mode.

  When the TAP controller is in this state, it moves to either the Run-Test/Idle state if TMS is low, or the Select-DR-Scan state if TMS is high.

3.2.11 Boundary Scan Order

Table 3.2.2 shows the boundary scan order with respect to the processor signals.

TDI → 1 (PC6)→ 2(PC7)→ ···→180(PC4)→181(PC2)→TDO

Table 3.2.2 JTAG Scan Order of the TMPA901CM Processor Pins

No.	Pin Name	No.	Pin Name	No.	Pin Name	No.	Pin Name	No.	Pin Name
	TDI								
1	PC6	41	PT2	81	SH2	121	SK1	161	PV4
2	PC7	42	PT5	82	SJ1	122	TSBRSV66	162	PU1
3	TSBRSV00	43	PB3	83	TSBRSV50	123	SL0	163	TSBRSV71
4	TSBRSV01	44	SM4	84	SB0	124	SL1	164	PU5
5	TSBRSV02	45	PA2	85	SJ6	125	TSBRSV67	165	TSBRSV72
6	TSBRSV03	46	PT3	86	TSBRSV51	126	SL6	166	PV1
7	TSBRSV04	47	PA0	87	SA0	127	SL5	167	PU2
8	TSBRSV05	48	PT6	88	SB1	128	SK5	168	PV5
9	TSBRSV06	49	PT0	89	TSBRSV52	129	TSBRSV68	169	PU6
10	TSBRSV07	50	PA1	90	SA1	130	SL2	170	PV2
11	TSBRSV08	51	PT4	91	SH3	131	SL4	171	PU3
12	TSBRSV09	52	PT1	92	TSBRSV53	132	SE6	172	TSBRSV73
13	TSBRSV10	53	PB1	93	SB2	133	SF4	173	PU7
14	TSBRSV11	54	PA3	94	SA2	134	SE0	174	PV6
15	TSBRSV12	55	PB0	95	TSBRSV54	135	SE7	175	PV3
16	TSBRSV13	56	PN0	96	TSBRSV55	136	SG1	176	TSBRSV74
17	TSBRSV14	57	PB2	97	TSBRSV56	137	SE1	177	PC3
18	TSBRSV15	58	TSBRSV38	98	SB3	138	SF5	178	TSBRSV75
19	TSBRSV16	59	PN1	99	SA3	139	SG5	179	PV7
20	TSBRSV17	60	TSBRSV39	100	SH4	140	SF0	180	PC4
21	TSBRSV18	61	TSBRSV40	101	TSBRSV57	141	SG2	181	PC2
22	TSBRSV19	62	SM6	102	TSBRSV58	142	SE2		TDO
23	TSBRSV20	63	SM7	103	TSBRSV59	143	SF6		
24	TSBRSV21	64	PT7	104	SB4	144	TSBRSV69		
25	TSBRSV22	65	SN0	105	SA4	145	SF1		
26	TSBRSV23	66	SN1	106	SA5	146	SG6		
27	TSBRSV24	67	TSBRSV41	107	SB5	147	SE3		
28	TSBRSV25	68	TSBRSV42	108	TSBRSV60	148	SG3		
29	TSBRSV26	69	TSBRSV43	109	TSBRSV61	149	SF7		
30	TSBRSV27	70	TSBRSV44	110	TSBRSV62	150	SF2		
31	TSBRSV28	71	TSBRSV45	111	SA6	151	SE4		
32	TSBRSV29	72	TSBRSV46	112	SB6	152	TSBRSV70		
33	TSBRSV30	73	TSBRSV47	113	SH7	153	SG7		
34	TSBRSV31	74	SJ2	114	TSBRSV63	154	SG4		
35	TSBRSV32	75	SJ4	115	SA7	155	SG0		
36	TSBRSV33	76	TSBRSV48	116	TSBRSV64	156	SF3		
37	TSBRSV34	77	SJ0	117	SB7	157	SE5		
38	TSBRSV35	78	SJ5	118	SK0	158	PU0		
39	TSBRSV36	79	SJ3	119	TSBRSV65	159	PU4		
40	TSBRSV37	80	TSBRSV49	120	SK4	160	PV0		

Note: TSBRSV[00:75] of boundary scan order is described as reserved signal

3.2.12 Instructions Supported by the JTAG Controller Cells

This section describes the instructions supported by the JTAG controller cells of the TMPA901CM.

(1) EXTEST instruction

The EXTEST instruction is used for external interconnect tests. The EXTEST instruction permits BSR cells at output pins to shift out test patterns in the Update-DR state and those at input pins to capture test results in the Capture-DR state.

Typically, before EXTEST is executed, the initialization pattern is shifted into the boundary scan register using the SAMPLE/PRELOAD instruction. If the boundary scan register is not reset, indeterminate data will be transferred in the Update-DR state and bus conflicts between ICs may occur. Figure 3.2.7 shows data flow when the EXTEST instruction is selected.

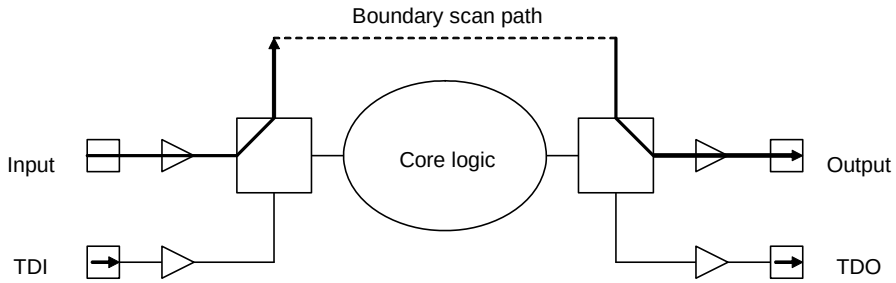


Figure 3.2.7 Test Data Flow when the EXTEST Instruction is Selected

The following steps describe the basic test procedure of the external interconnect test.

1. Reset the TAP controller to the Test-Logic-Reset state.
2. Load the instruction register with the SAMPLE/PRELOAD instruction. This causes the boundary scan register to be connected between TDI and TDO.
3. Reset the boundary scan register by shifting certain data in.
4. Load the test pattern into the boundary scan register.
5. Load the instruction register with the EXTEST instruction.
6. Capture the data applied to the input pin into the boundary scan register.
7. Shift out the captured data while simultaneously shifting the next test pattern in.
8. Send out the test pattern in the boundary scan register at the output on the output pin.

Repeat steps 6 to 8 for each test pattern.

(2) SAMPLE/PRELOAD instruction

This instruction targets the boundary scan register between TDI and TDO. As its name implies, the SAMPLE/PRELOAD instruction provides two functions.

SAMPLE allows the input and output pads of an IC to be monitored. While it does so, it does not disconnect the system logic from the IC pins. SAMPLE is executed in the Capture-DR state. It is mainly used to capture the values of the IC's I/O pins on the rising edge of TCK during normal operation. Figure 3.2.8 shows the flow of data for the SAMPLE phase of the SAMPLE/PRELOAD instruction.

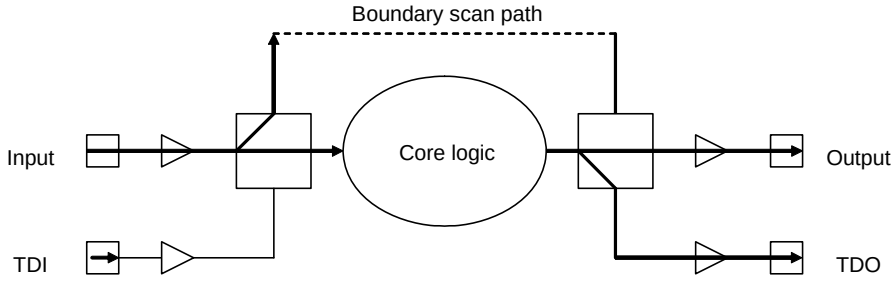


Figure 3.2.8 Test Data Flow while the SAMPLE is Selected

PRELOAD allows the boundary scan register to be reset before any other instruction is selected. For example, prior to selection of the EXTEST instruction, PRELOAD is used to load reset data into the boundary scan register. PRELOAD permits data shifting of the boundary scan register without interfering with the normal operation of the system logic. Figure 3.2.9 shows the data flow for the PRELOAD phase of the SAMPLE/PRELOAD instruction.

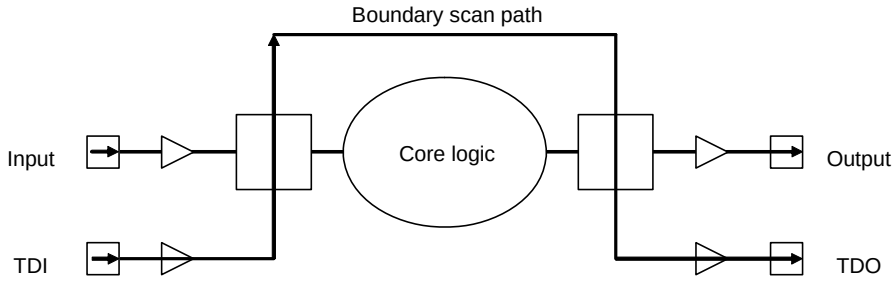


Figure 3.2.9 Test Data Flow while PRELOAD is Selected

(3) BYPASS instruction

This instruction targets the bypass register between JTDI and JTDO. The bypass register provides the shortest serial path that bypasses the IC (between JTDI and JTDO) when the test does not require control or monitoring of the IC. The BYPASS instruction does not cause interference in the normal operation of the on-chip system logic. Figure 3.2.10 shows the data flow through the bypass register when the BYPASS instruction is selected.

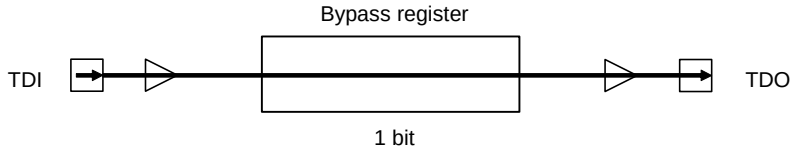


Figure 3.2.10 Test Data Flow when the BYPASS Instruction is Selected

(4) CLAMP instruction

The CLAMP instruction outputs the value that boundary scan register is programmed according to the PRELOAD instruction, and execute Bypass operation.

The CLAMP instruction selects the bypass register between TDI and TDO.

(5) HIGHZ instruction

The HIGHZ instruction disables the output of the internal logical circuits. When the HIGHZ instruction is executed, it places the 3-state output pins in the high-impedance state.

The HIGHZ instruction also selects the bypass register between TDI and TDO.

- Notes

This section describes the cautions of the JTAG boundary-scan operations specific to the processor.

1) The PR2 pin serves as an I/O pin. However, the PR2 pin does not support the capture function by using SAMPLE/PRELOAD instructions because the BSR is connected to the pin.
2) The JTAG circuit can be released from the reset state by either of the following two methods:
   - Assert TRSTn, initialize the JTAG circuit, and then deassert TRSTn.
   - Supply the TCK signal for 5 or more clock pulses to TCK while pulling the TMS pin High.

3.3 Memory Map

The memory map of TMPA901CM is as follows:

Table 3.3.1 Outline of access to internal area

Item	Outline of access	
CPU address width	32 bit	
CPU data bus width	32 bit	
Internal operation frequency	Max 200MHz @ 0 to 70°C Max 150MHz @ -20 to 85°C	
Minimum bus cycle	1-f_{CLK} clock access (5ns at 200MHz)	
Internal RAM	32-bit 1-HCLK clock access	
Internal Boot ROM	32-bit 1-HCLK clock access	
Internal I/O	32-bit,1-HCLK clock access	LCDC, LCDDA, INTC, DMAC, USB Device, USB Host, I^2S, NANDFC, SSP,MPMC
	32-bit,2-PCLK clock access	A/D C, TSI, Timer/PWM, PMC, I^2C, UART, RTC, WDT, OFD, System C, PLL CG, GPIO

Start Address	Activation of the internal BOOT ROM		Activation of external memory
0x0000_0000	Internal ROM : 8KB+ 8KB	Remap area (8KB)	**SMCCS0n**
0x0000_2000			
0x0000_4000	**SMCCS0n**	External area (15.8MB)	
0x0100_0000 0x2000_0000	Unused area	External area (512MB)	Unused area
0x2100_0000	**SMCCS0n**	External area (496MB)	**SMCCS0n**
0x4000_0000	**DMCCSn**	External area (512MB)	**DMCCSn**
0x6000_0000	**SMCCS1n**	External area (512MB)	**SMCCS1n**
0x8000_0000	Unused area	External area (512MB)	Unused area
0xA000_0000	Unused area	External area (512MB)	Unused area
0xC000_0000	Unused area	External area (512MB)	Unused area
0xE000_0000	Unused area	External area (256MB)	Unused area
0xF000_0000	Internal IO-0 (APB) : 1MB	Internal I/O area (128MB)	Internal IO-0 (APB) : 1MB
0xF010_0000	Unused area		Unused area
0xF080_0000	Internal IO-1 (APB Port1/2) : 1MB		Internal IO-1 (APB Port1/2) : 1MB
0xF090_0000	Internal IO-2 (APB Port2/2) : 1MB		Internal IO-2 (APB Port2/2) : 1MB
0xF0A0_0000	Unused area		Unused area
0xF200_0000	Internal IO-3 (AHB+APB) : 16MB		Internal IO-3 (AHB+APB) : 16MB
0xF300_0000	Unused area		Unused area
0xF400_0000	Internal IO-4 (AHB) : 16MB		Internal IO-4 (AHB) : 16MB
0xF600_0000	Unused area		Unused area
0xF800_0000	Unused area	Internal memory area (128MB)	Unused area
0xF800_2000	Internal RAM-3 : 8KB(Remap)		Internal RAM-3 : 8KB(Remap)
0xF800_4000	Internal RAM-0 : 16KB		Internal RAM-0 : 16KB
0xF800_8000	Internal RAM-1 : 8KB		Internal RAM-1 : 8KB
0xF800_A000	Unused area		Unused area

0xFFFF_FFFF

Note1: Space between 0x0000_0000 and 0x0000_1FFF (8KB) is a Remap area, and the Internal RAM3 area will be accessed when Remap is set to Remap_ON (access to F8000_2000 also leads to the RAM3 area).

Note2: Access to unused area is prohibited.

Figure 3.3.1 Memory map (Details of start mode, external areas and internal area)

Address	Activation of the internal BOOT ROM		Bus Master and Slave connection ○ : Access available, × : Access unavailable — : Don't access						
			CPU(D) M1	CPU(I) M2	LCDC M3	LCDDA M4	DMA1 M5	DMA2 M6	USB M7
0x0000_0000 0x0000_2000	Internal ROM : 8KB+ 8KB	Remap area (8KB)	○	○	×	×	○	○	×
0x0000_4000	**SMCCS0n**	External area (15.8MB)	○	○	○	○	○	○	○
0x0100_0000 0x2000_0000	Unused area	External area (512MB)	–						
0x2100_0000	**SMCCS0n**	External area (496MB)	○	○	○	○	○	○	○
0x4000_0000	**DMCCSn**	External area (512MB)	○	○	○	○	○	○	○
0x6000_0000	**SMCCS1n**	External area (512MB)	○	○	○	○	○	○	○
0x8000_0000	Unused area	External area (1792MB)	–						
0xF000_0000	Internal IO-0 (APB) : 1MB	Internal I/O area (128MB)	**Please refer to next page.**						
0xF010_0000	Unused area								
0xF080_0000	Internal IO-1 (APB Port1/2) : 1MB								
0xF090_0000	Internal IO-2 (APB Port2/2) : 1MB								
0xF0A0_0000	Unused area								
0xF200_0000	Internal IO-3 (AHB+APB) : 16MB								
0xF300_0000	Unused area								
0xF400_0000	Internal IO-4 (AHB) : 16MB								
0xF600_0000	Unused area		–						
0xF800_0000	Unused area	Internal memory area(128MB)	–						
0xF800_2000	Internal RAM-3 : 8KB(Remap)		○	○	○	○	○	○	○
0xF800_4000	Internal RAM-0 : 16KB Dual port RAM share with LCDDA		○	○	○	○	○	○	○
0xF800_8000	Internal RAM-1 : 8KB share with USB Host		○	○	○	○	○	○	○
0xF800_A000	Unused area		–						

0xFFFF_FFFF

Note: USB Host can access the area of 0xF800_8000 to 0xF800_9FFF only.

Figure 3.3.2 Memory map (details of start mode and Bus Master and Slave connection)

Start address	End address	Details of Internal IO		Accessible Master	
0xF000_0000	0xF000_0FFF	Internal IO (APB) 1MB	SysCtrl	M1(CPU Data)	Internal IO area
0xF001_0000	0xF001_0FFF		WDT		
0xF002_0000	0xF002_0FFF		PMC		
0xF003_0000	0xF003_0FFF		RTC		
0xF004_0000	0xF004_0FFF		Timer01/PWM		
0xF004_1000	0xF004_1FFF		Timer23/PWM		
0xF004_2000	0xF004_2FFF		Timer45		
0xF005_0000	0xF005_0FFF		PLLCG		
0xF006_0000	0xF006_0FFF		TSI		
0xF007_0000	0xF007_0FFF		I^2C0		
0xF007_1000	0xF007_1FFF		Reserved		
0xF008_0000	0xF008_0FFF		ADC		
0xF009_0000	0xF009_0FFF		OFD		
0xF00A_0000	0xF00A_0FFF		EBI		
0xF00B_0000	0xF00B_0FFF		LCDOP		
0xF080_0000	0xF080_FFFF	Internal IO (APB) 1MB	PORT	M1(CPU Data)	
0xF200_0000	0xF200_1FFF	Internal IO (AHB+APB) 16MB	UART0,1 note2)	M1(CPU Data) M5(DMAC1) M6(DMAC2)	
0xF200_2000	0xF200_3FFF		SSP		
0xF200_4000	0xF200_4FFF		Reserved		
0xF201_0000	0xF201_0FFF		NANDFC		
0xF202_0000	0xF202_0FFF		Reserved		
0xF203_0000	0xF203_0FFF		Reserved		
0xF204_0000	0xF204_0FFF		I^2S		
0xF205_0000	0xF205_0FFF		LCDDA		
0xF400_0000	0xF400_0FFF	Internal IO (AHB) 16MB	INTC	M1(CPU Data)	
0xF410_0000	0xF410_0FFF		DMAC		
0xF420_0000	0xF420_0FFF		LCDC		
0xF430_0000	0xF430_0FFF		MPMC0		
0xF431_0000	0xF431_0FFF		MPMC1		
0xF440_0000	0xF440_0FFF		USB Device		
0xF450_0000	0xF450_F000		USB Host		

Note1: Addresses that are assigned to the above table are Reserved areas. Reserved addresses must not access.

Note2: UART1 don't support DMA Function.

Figure 3.3.3 Memory map (details of internal registers)

3.3.1 Boot mode

A few boot modes are available for choice to this microprocessor depending on the external pin setting.

1. Boot memory setting

Mode setting pin			Operation mode
RESETn	AM1	AM0	
	0	1	Start from the external 16-bit NOR Flash memory (Internal BOOT_TOM cannot be seen)
	1	0	Start from the external 32-bit NOR Flash memory (Internal BOOT_TOM cannot be seen)
	1	1	BOOT (start from the Internal boot ROM)
	0	0	TEST (this setting cannot be used)

2. External memory voltage setting (Except NANDF)

Mode setting pin	Operation mode
SELVCCM	
0	Memory-related control pins operate at 1.8±0.1V (DVCCM)
1	Memory-related control pins operate at 3.3±0.3V (DVCCM)

3. External memory controller setting

Mode setting pin	Operation mode
SELMEMC	
0	Only the SDR (Single Data Rate) and Mobile SDR types of SDRAM can be used.
1	Only the Mobile DDR (Mobile Double Data Rate) type of SDRAM can be used.

4. JTAG pin setting

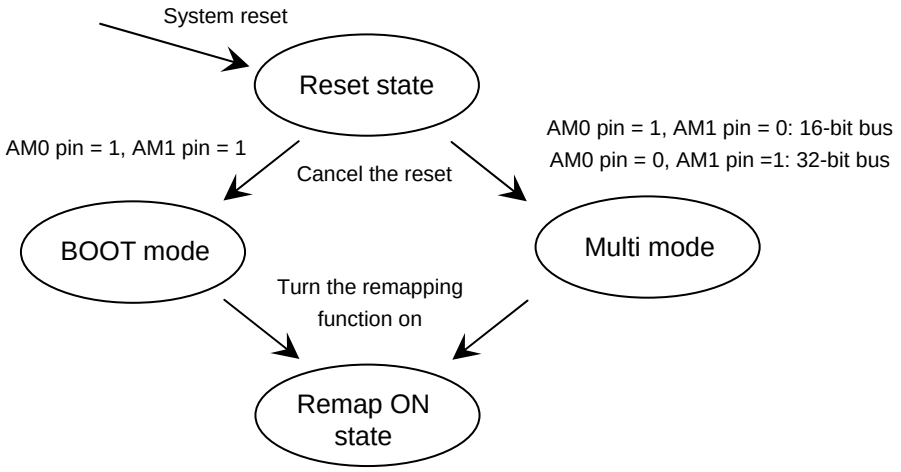
Mode setting pin	Operation mode
SELJTAG	
0	Set “0” to this pin except Boundary Scan Mode. This setting can be used as regular Debug Mode Note: Debugging cannot be carried out during internal BOOT with AM1 = 1 and AM0 = 1.
1	This setting can be used as Boundary Scan Mode

3.4 System Controller

3.4.1 Remapping function

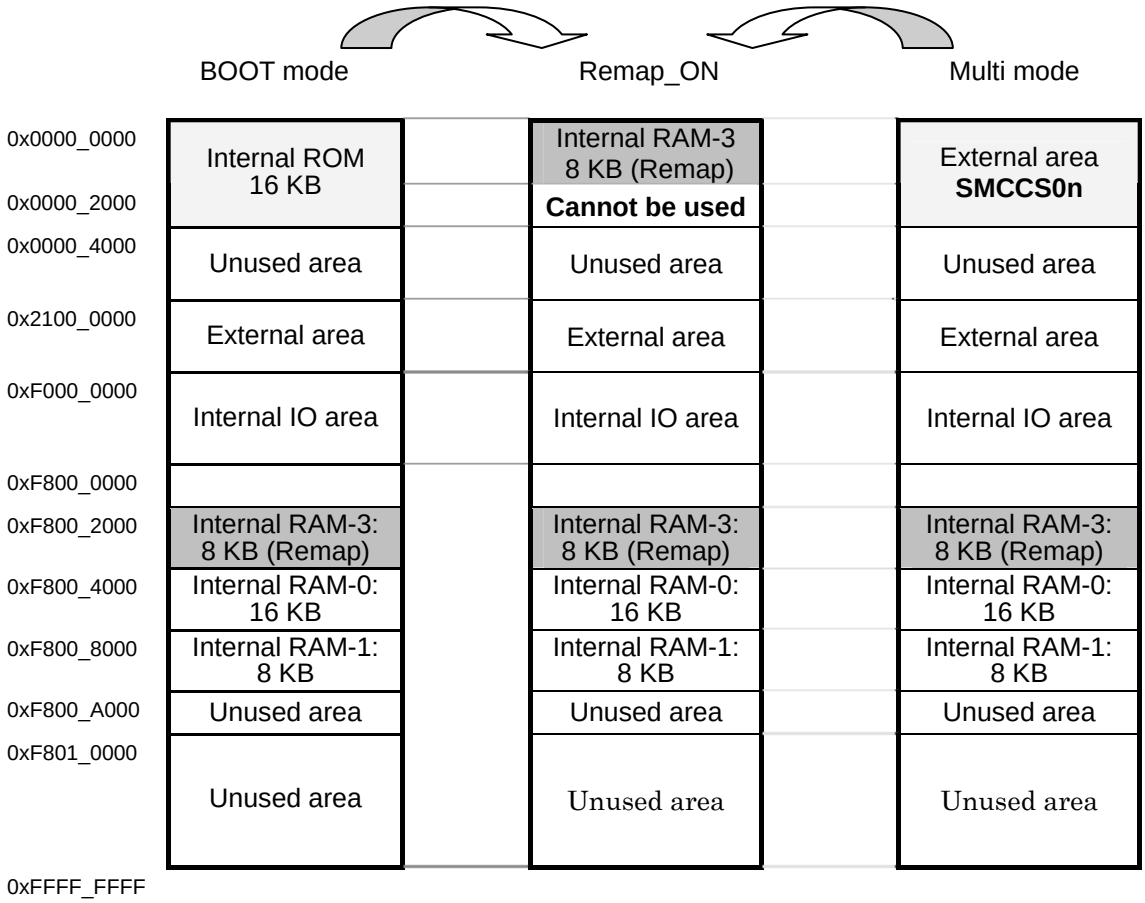
Using the remapping function, this LSI can access the 8K-byte area of the built-in RAM from two memory areas (0x0000_0000 to 0x0000_1FFF and 0xF800_2000 to 0xF800_3FFF).

It turns on the Remapping function by writing Remap<REMAP>.



Note: The Remap ON status is activated by the register setting, but it can only be deactivated by resetting the system or canceling it in the PCM status.

Figure 3.4.1 Transition of the memory space status



Note: Space between 0x0000_0000 and 0x0000_1FFF (8KB) is a Remap area, and the built-in RAM3 area will be accessed when Remap is set to Remap_ON (access to 0xF8000_2000 also leads to the RAM3 area).

Figure 3.4.2 Memory map (details of boot mode and external areas)

3.4.2 Register Descriptions

The system controller has the following register.

Base address = 0F000_0000

Register Name	Address (base+)	Description
Remap	0x0004	Reset memory map (REMAP)

1. Remap Register

Address = (0xF000_0000) + 0x0004

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read undefined. Write as zero.
[0]	REMAP	RW	0y0	REMAP setting

[Explanation]

a. <REMAP>

It is the register that enables the REMAP function.
By writing arbitrary data, the built-in RAM3 can be accessed from the beginning of the memory map. The register cannot turn off the remap status (to reset to the initial state).

3.5 Clock Controller

3.5.1 Overview

The clock controller is a circuit that controls the clock for the overall MCU. It has the following features:

a. By using a clock multiplication circuit (PLL), the clock controller supplies a clock of up to 200 MHz to the CPU. As a multiplied figure, x1, x6, or x8 can be dynamically selected.

b. The clock gear contributes to reduction of the consumption current.

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

Transition of clock operation modes is as follows:

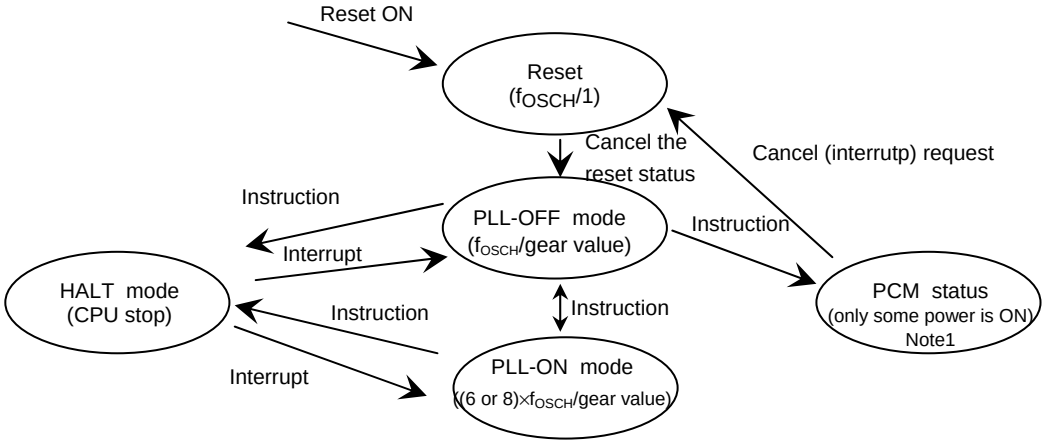
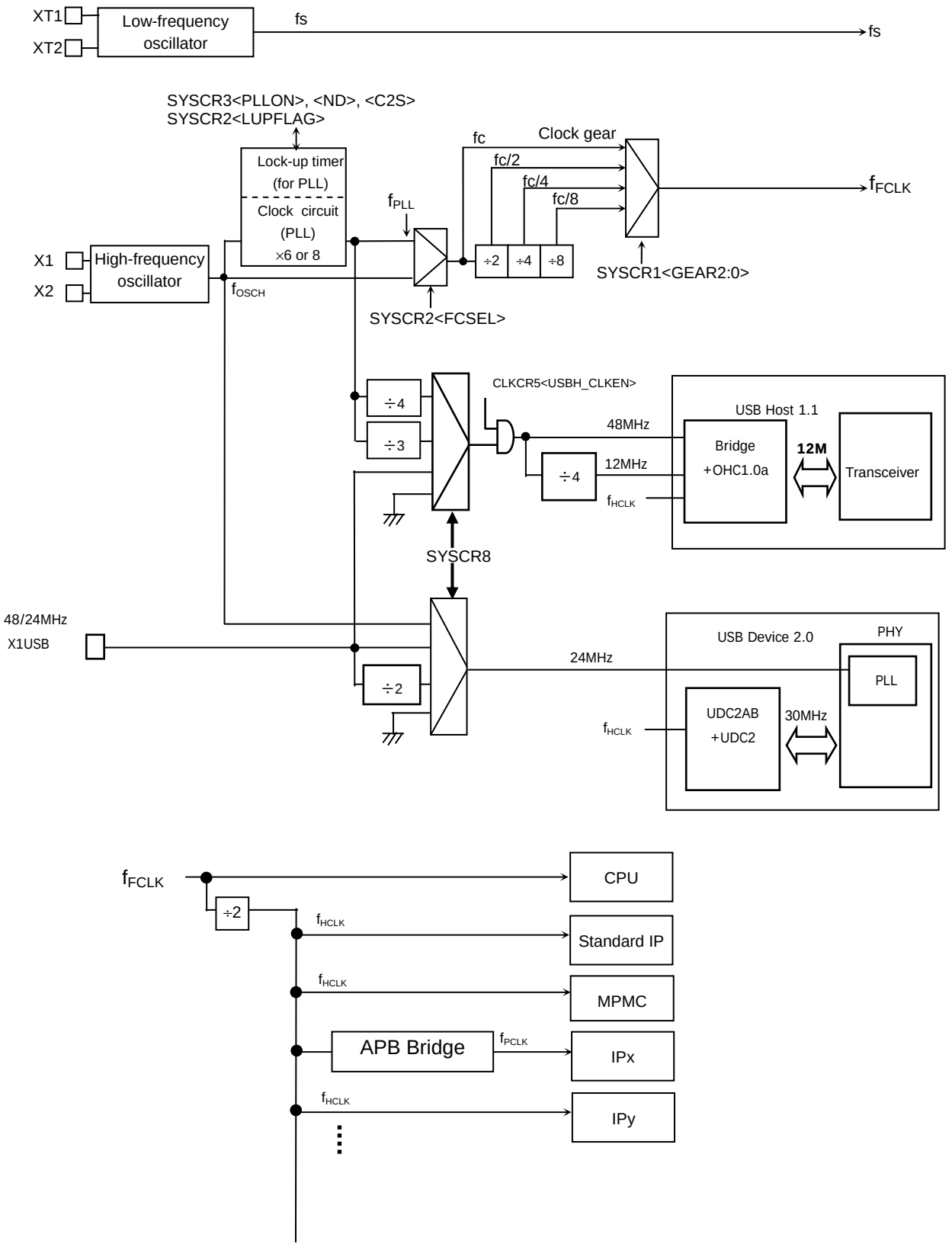


Figure 3.5.1 Clock mode status transition

Note1: About PCM mode, please refer to chapter 26 (Power Management Circuit).

3.5.2 Block Diagrams



Clock frequency input from the X1 and X2 pins is defined as f_{OSCH}, clock frequency input from the XT1 and XT2 pins is defined as f_S, and the clock selected in SYSCR1<GEAR2:0> is defined as clock f_{FCLK} for the CPU core. For peripheral IPs connected to the AHB bus, a clock obtained by dividing f_{FCLK} by 2 is defined as f_{HCLK} (Signal name: HCLK). For peripheral IPs connected to the APB bus, a clock obtained by dividing f_{FCLK} by 2 is defined as f_{PCLK} (Signal name: PCLK).

Also, two types of clock, for DRAM and for SRAM/NORF respectively, are input in the memory controller, and as a SRAM/NORF clock, f_{HCLK} or a clock obtained by dividing f_{HCLK} by 2 can be selected (Please refer to MPMC section).

Clock constraints are defined below. Select a clock that meets these criteria for intended applications.

Table 3.5.1 Clock constraints @ Ta = 0 to 70°C

	Lowest frequency	Highest frequency	Notes
(a) f_{OSCH} (High speed oscillator frequency)	10 MHz	27 MHz	
(b) f_{PLL} (PLL output frequency)	60 MHz	200 MHz	
(c) f_{FCLK} (Frequency for the CPU)	1.25 MHz	200 MHz	
(d) f_{USB} (Frequency for the USB)	24 MHz	24 MHz	Accuracy of 24MHz±100 ppm is required.
(e) f_{USB} (Frequency for the USB)	48MHz	48MHz	Accuracy of 48MHz±100 ppm is required.
(f) f_S (Low speed oscillator frequency)	30 kHz	34 kHz	

Table 3.5.2 Clock constraints @ Ta = -20 to 85°C

	Lowest frequency	Highest frequency	Notes
(b) f_{OSCH} (High speed oscillator frequency)	10 MHz	27 MHz	
(b) f_{PLL} (PLL output frequency)	60 MHz	150MHz	
(c) f_{FCLK} (Frequency for the CPU)	1.25 MHz	150 MHz	
(d) f_{USB} (Frequency for the USB)	24 MHz	24 MHz	Accuracy of 24MHz±100 ppm is required.
(e) f_{USB} (Frequency for the USB)	48MHz	48MHz	Accuracy of 48MHz±100 ppm is required.
(f) f_S (Low speed oscillator frequency)	30 kHz	34 kHz	

The table below shows the examples of recommended uses that meet the criteria listed above.

Table 3.5.3 Examples of recommended uses @ 0 to 70°C

	High speed oscillation: f_{OSCH}	PLL output clock: f_{PLL}	Clock for CPU: f_{FCLK}	Clock for USB: f_{USB}
(1) USB required, Maximum CPU: 192 MHz	24 MHz	Maximum of 192 MHz	Maximum of 192 MHz	24 MHz / 48MHz
(2) USB required, Maximum CPU: 200 MHz	25 MHz	Maximum of 200 MHz	Maximum of 200 MHz	24 MHz / 48MHz (Input from the X1USB pin is required)
(3) USB not required Maximum CPU: 200 MHz	25 MHz	Maximum of 200 MHz	Maximum of 200 MHz	–

Table 3.5.4 Examples of recommended uses @ -20 to 85°C

	High speed oscillation: f_{OSCH}	PLL output clock: f_{PLL}	Clock for CPU: f_{FCLK}	Clock for USB: f_{USB}
(1) USB required, Maximum CPU: 144 MHz	24 MHz	Maximum of 144 MHz	Maximum of 144 MHz	24 MHz / 48MHz
(2) USB required, Maximum CPU: 150 MHz	25 MHz	Maximum of 150 MHz	Maximum of 150 MHz	24 MHz / 48MHz (Input from the X1USB pin is required)
(3) USB not required Maximum CPU: 150 MHz	25 MHz	Maximum of 150 MHz	Maximum of 150 MHz	–

3.5.3 Operation Descriptions

3.5.3.1 Register Descriptions

The following lists the SFRs and their functions.

base address = 0xF005_0000

Register Name	Address (base+)	Description
Reserved	0x000	Reserved
SYSCR1	0x004	System Control Register 1
SYSCR2	0x008	System Control Register 2
SYSCR3	0x00C	System Control Register 3
SYSCR4	0x010	System Control Register 4
SYSCR5	0x014	System Control Register 5
SYSCR6	0x018	System Control Register 6
SYSCR7	0x01C	System Control Register 7
SYSCR8	0x020	System Control Register 8
Reserved	0x040	Reserved
Reserved	0x044	Reserved
Reserved	0x048	Reserved
Reserved	0x04C	Reserved
Reserved	0x050	Reserved
CLKCR5	0x054	Clock Control Register 5

1. SYSCR1 (System Control Register 1)

Address = (0xF005_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	GEAR	R/W	0y000	Clock gear programming (fc) 0y000: fc 0y001: fc/2 0y010: fc/4 0y011: fc/8 0y1xx: Reserved

[Description]

a. <GEAR>

Programs the clock gear.

0y000: fc

0y001: fc/2

0y010: fc/4

0y011: fc/8

0y1xx: Reserved

2. SYSCR2 (System Control Register-2)

Address = (0xF005_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[6:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	FCSEL	R/W	0y0	Selection of the PLL output clock 0y0: f_{OSCH} 0y1: f_{PLL}
[0]	LUPFLAG	RO	0y0	End flag of the PLL lockup counter Read: 0y0: Not end 0y1: End Write: Invalid

[Description]

a. <FCSEL>

Selects the clock to be output from the PLL.

0y0: f_{OSCH}

0y1: f_{PLL}

b. <LUPFLAG>

Indicates the state of the PLL lock-up counter.

0y0: Not end

0y1: End

3. SYSCR3 (System Control Register 3)

Address = (0xF005_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	PLLON	R/W	0y0	PLL operation control 0y0: OFF 0y1: ON
[6]	–	–	Undefined	Read as undefined. Write as zero.
[5]	C2S	R/W	0y1	PLL constant value setting1 **Always write 0**
[4:0]	ND	R/W	0y00111	PLL constant value setting 2 0y00101 for x6, 0y00111 for x8

[Description]

a. <PLLON>

Controls the operation of the PLL.

0y0: OFF

0y1: ON

b. <C2S>

PLL constant value setting 1

1 is set as default. Rewrite it to 0 before use.

c. <ND>

PLL constant value setting 2

0y0_0101 for x6, 0y0_0111 for x8

4. SYSCR4 (System Control Register 4)

Address = (0xF005_0000) + (0x010)

<table>
<tr><th>Bit</th><th>Bit Symbol</th><th>Type</th><th>Reset Value</th><th>Description</th></tr>
<tr><td>[31:8]</td><td>–</td><td>–</td><td>Undefined</td><td>Read as undefined. Write as zero.</td></tr>
<tr><td>[7:4]</td><td>RS</td><td>R/W</td><td>0y0111</td><td>PLL constant value setting 3
<table><tr><td colspan="2">x 8</td><td colspan="2">x 6</td></tr><tr><td>140MHz or more</td><td>Less than 140MHz</td><td>140MHz or more</td><td>Less than 140MHz</td></tr><tr><td>0y0110</td><td>0y1001</td><td>0y0110</td><td>0y0111</td></tr></table></td></tr>
<tr><td>[3:2]</td><td>IS</td><td>R/W</td><td>0y10</td><td>PLL constant value setting 4
Always write 0y01</td></tr>
<tr><td>[1:0]</td><td>FS</td><td>R/W</td><td>0y01</td><td>PLL constant value setting 5
<table><tr><td colspan="2">x 8</td><td colspan="2">x 6</td></tr><tr><td>140MHz or more</td><td>Less than 140MHz</td><td>140MHz or more</td><td>Less than 140MHz</td></tr><tr><td>0y01</td><td>0y10</td><td>0y01</td><td>0y10</td></tr></table></td></tr>
</table>

[Description]

a. <RS>

PLL constant value setting 3

Program the following values according to PLL multiplying factor and frequency to be multiplied.

x 8

140MHz or more: 0y0110

Less than 140MHz: 0y1001

x 6

140MHz or more: 0y0110

Less than 140MHz: 0y0111

b. <IS>

PLL constant value setting 4

0y10 is set as default. Rewrite it to 0y01 before use.

c. <FS>

PLL constant value setting 5

Program the following values according to the PLL multiplying factor and frequency to be multiplied.

x 8

140MHz or more: 0y01

Less than 140MHz: 0y10

x 6

140MHz or more: 0y01

Less than 140MHz: 0y10

5. SYSCR5 (System Control Register 5)

Address = (0xF005_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined.
[0]	PROTECT	RO	0y0	Protect Flag 0y0: OFF 0y1: ON

[Description]

By setting a dual key to the SYSCR6 and SYSCR7 registers, protection (write operation to certain SFRs in the clock controller) can be activated or released.

[Dual key]

1st-KEY : Consecutive writing of 0x5A to SYSCR6 and 0xA5 to SYSCR7

2nd-KEY : Consecutive writing of 0xA5 to SYSCR6 and 0x5A to SYSCR7

The protection status can be checked by reading SYSCR5<PROTECT>.

Reset operation turns protection OFF. If write operation is executed to certain SFRs shown below while protection is ON, written data will be invalidated.

The SFRs:

SYSCR1, SYSCR2, SYSCR3, SYSCR4, SYSCR5, SYSCR8

CLKCR5

6. SYSCR6 (System Control Register 6)

Address = (0xF005_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	P-CODE0	WO	0X00	Protect code setting-0

[Description]

a. <P-CODE0>

Used to set the protect code 0.

7. SYSCR7 (System Control Register 7)

Address = (0xF005_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	P-CODE1	WO	0X00	Protect code setting-1

[Description]

a. <P-CODE1>

Used to set the protect code 1.

8. SYSCR8 (System Control Register 8)

Address = (0xF005_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset value	Description
[31:8]	-	-	Undefined	Read as undefined. Write as zero.
[7:6]	-	-	Undefined	Read as undefined. Write as zero.
[5:4]	USBD_CLKSEL	R/W	0y00	Clock selection for USB Device Controller: 00 : fix to GND 01 :1/2 clock of X1USB 10 : clock of X1USB 11 : clock of X1
[3]	-	-	Undefined	Read as undefined. Write as zero.
[2:0]	USBH_CLKSEL	R/W	0y000	Clock selection for USB host Controller 000 : fix to GND 001 : clock of X1USB 010: 1/3 f_{PLL} 011 : fix to GND 100 : 1/4 f_{PLL} 101 : clock of X1 110 : fix to GND 111 : fix to GND

9. CLKCR5 (Clock Control Register-5)

Address = (0xF005_0000) + (0x0054)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined. Write as zero.
[6]	Reserved	R/W	0y1	Read as undefined. Write as one.
[5]	–	–	Undefined	Read as undefined. Write as zero.
[4]	USBH_CLKEN	R/W	0y1	Clock selection for USB HOST controller 0y0 : disable 0y1 : enable
[3]	Reserved	R/W	0y1	Write as one
[2]	SEL_TIM45	R/W	0y1	Selection of a prescaler clock for Timer45 0y0: fs (32.768 kHz) clock 0y1: $f_{PCLK}/2$
[1]	SEL_TIM23	R/W	0y1	Selection of a prescaler for Timer23 0y0: fs (32.768 kHz) clock 0y1: f_{PCLK} /2
[0]	SEL_TIM01	R/W	0y1	Selection of a prescaler for Timer01 0y0: fs (32.768 kHz) clock 0y1: $f_{PCLK}/2$

[Description]

a. < USBH_CLKEN >

Clock selection for USB Host controller

0y0: Disable

0y1: Enable

If the user desires to change a setting of the clock for USB Host, the user must disable the output of the clock first.

b. <SEL_TIM45>

Selects the prescaler clock for Timer45.

0y0: fs (32.768 kHz) clock

0y1: $f_{PCLK}/2$

c. <SEL_TIM23>

Selects the prescaler clock for Timer23.

0y0: fs (32.768 kHz) clock

0y1: $f_{PCLK}/2$

d. <SEL_TIM01>

Selects the prescaler clock for Timer01.

0y0: fs (32.768 kHz) clock

0y1: $f_{PCLK}/2$

3.5.4 System Clock Controller

The system clock controller generates a clock to be supplied to the CPU core (f_{FCLK}) and other built-in I/Os (f_{HCLK}). With the f_{OSCH} or f_{PLL} clock as an input, it is possible to use SYSCR1<GEAR2:0> to change the high speed clock gear to 1, 2, 4, or 8-speed (fc, fc/2, fc/4, or fc/8) to reduce power consumption.

Reset operation switches the mode to PLL-OFF, and <GEAR2:0> is initialized to 0y000; therefore, frequency of the CPU clock f_{FCLK} will be the same as f_{OSCH}. For example, when a 24 MHz oscillator is connected to the X1 and X2 pins, the frequency of f_{FCLK} becomes 24 MHz when reset operation is executed.

(1) Clock gear

By using the clock gear selection register SYSCR1<GEAR2:0>, the gear can be set to fc, fc/2, fc/4, or fc/8.

Changing f_{FCLK} by using the clock gear contributes to reduction of power consumption.

An example of clock gear switching is as follows:

```
[Setting example]
;
(SYSCR1)        ←        0x0000_0011        ; switch fFCLK to 1/8.
```

3.5.5 PLL Clock Multiplier

The PLL outputs f_{PLL} clock signals whose frequency is 6 or 8 times the f_{OSCH}. By using the PLL, it is possible to lower the oscillator frequency and make the internal clock faster.

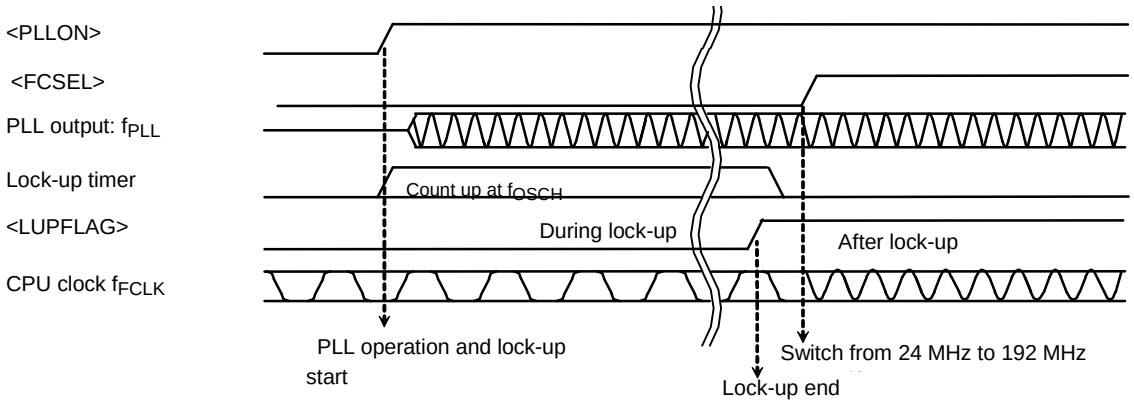
Since the PLL is initialized to the halt state when reset operation is executed, it is necessary to configure the SYSCR2, SYSCR3 and SYSCR4 registers when using the PLL.

As with an oscillator, this circuit requires time to stabilize the f_{PLL} clock signals after operation is enabled, and the time required is called lock-up time.

A 12-stage binary counter can be used to check the lock-up time. For example, lock-up time is approximately 164μs when f_{OSCH} = 25 MHz.

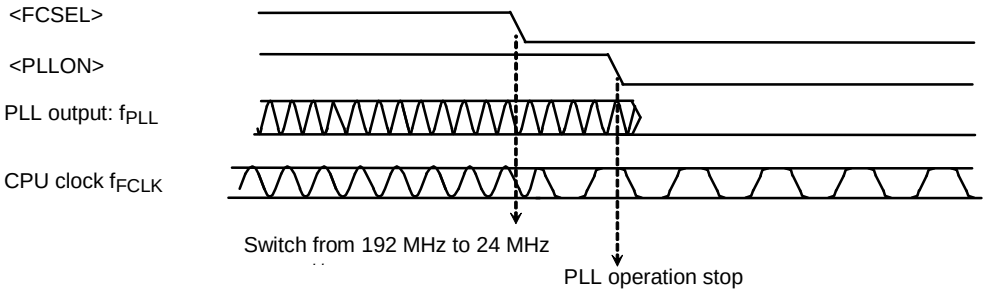
Examples of the PLL start and stop settings are as follows:

```
Setting example – 1: PLL start
            SYSCR4      ←    0x00000065     ; Set the constant of PLL x8
            SYSCR3      ←    0x00000087     ; Operation is activated with PLL x8
LOCKUP:
            SYSCR2      →    r0             ; <LUPFLAG> == 1?
            LDR   r1, = 0x01
            AND   r0,r0,r1
            LDR   r1, = 0x01                ;
            CMP  r0 ,r1
            BNE  LOCKUP                     ; r0 ≠r1 ,   jump to LOCKUP
            (SYSCR2)    ←    0x00000002     ; <FCSEL> = 1 (change from 24 MHz to 192 MHz)
```



Setting example – 2: PLL stop

```
        (SYSCR2)    ←   0x0000_0000    ; <FCSEL> = 0  (change from 192 MHz to 24 MHz)
LUP:    Dummy instruction execution (Note)
        (SYSCR3)    ←   0x0000_0007    ; <PLLON> = 0
```



Note: When switching <FCSEL> from 1 to 0, a few clock cycles are required before f_{FCLK} is changed to f_{OSCH} after the register write is completed. Therefore, it is necessary to first wait for the required clock cycles and then execute the next instruction. More specifically, execute 10 NOP instructions.

3.6 Boot ROM

TMPA901CM contains a boot ROM for loading a user program to the internal RAM.
The following loading methods are supported.

3.6.1 Operation Modes

TMPA901CM has two operation modes: external memory mode and internal boot ROM mode. Either mode is selected in accordance with the AM1 and AM0 pin status when RESETn is asserted.

(1) External memory mode: After reset, the CPU fetches instructions from external memory and executes them.

(2) Internal boot ROM mode: After reset, the CPU fetches instructions from the internal boot ROM and executes them. According to the program in the internal boot ROM, a user program is transferred to the internal RAM via USB communication and branches into the program in the internal RAM.
This triggers the user program to boot.
Table 3.6.2 shows the overview of boot operation.

Table 3.6.1 Operation Modes

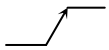
Mode Setting Pins			Operation Mode
RESETn	AM1	AM0	
	0	1	Start from the external bus memory (with 16-bit bus)
	1	0	Start from the external bus memory (with 32-bit bus)
	1	1	BOOT (start from the internal boot ROM)
	0	0	TEST (setting prohibited)

Table 3.6.2 Overview of boot operation

Priority	Loading			Operation after loading
	Source	I/F	Destination	
1	USB host such as a PC	USB	Internal RAM	Branch into the internal 8 KB_RAM 0x0000_0000

3.6.2 Hardware Specifications of the Internal Boot ROM

(1) Memory map

Figure 3.6.1 shows a memory map of BOOT mode.

The internal boot ROM consists of 16 KB ROM and is assigned to addresses from 0x0000_0000 to 0x0000_3FFF.

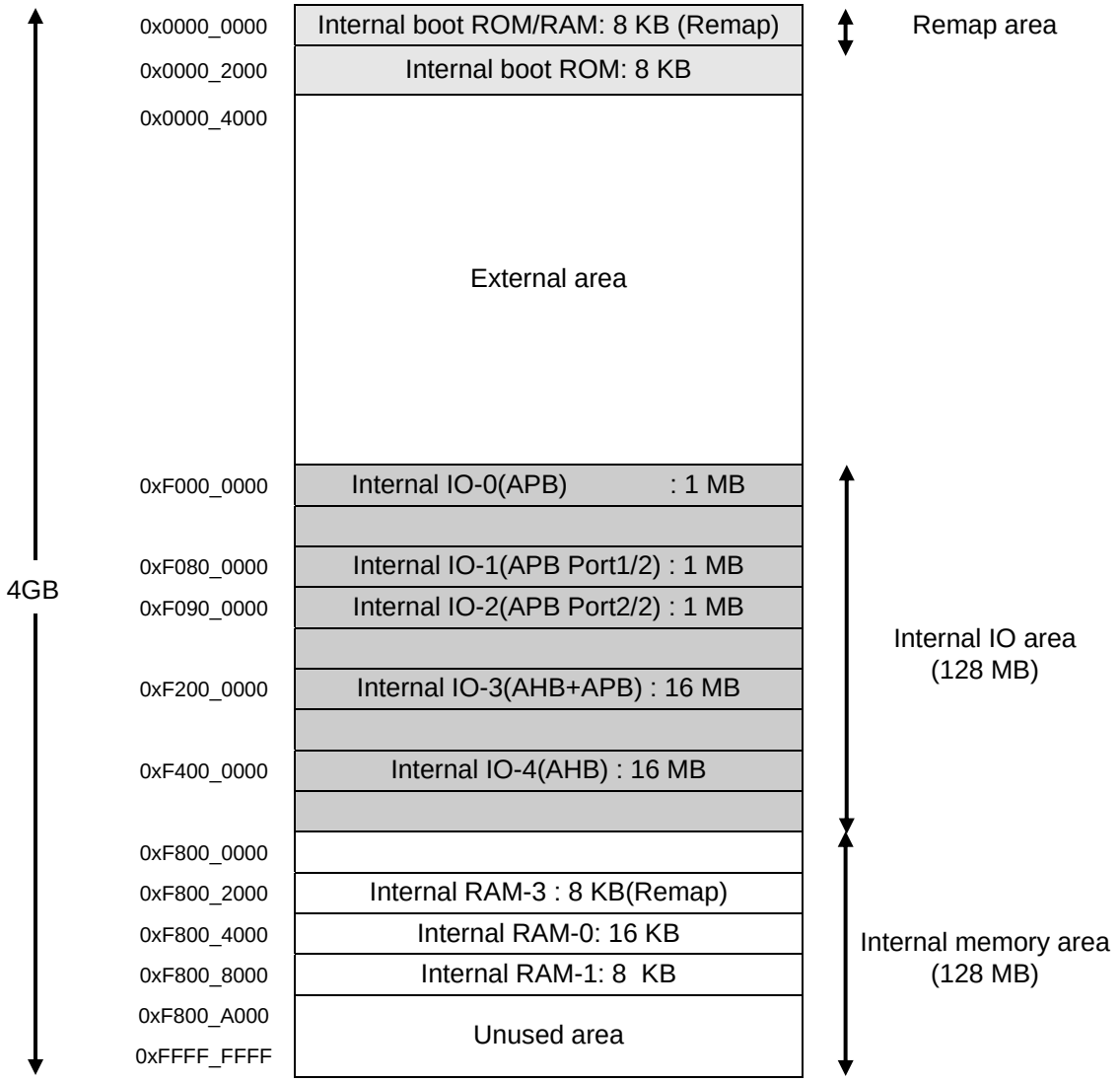
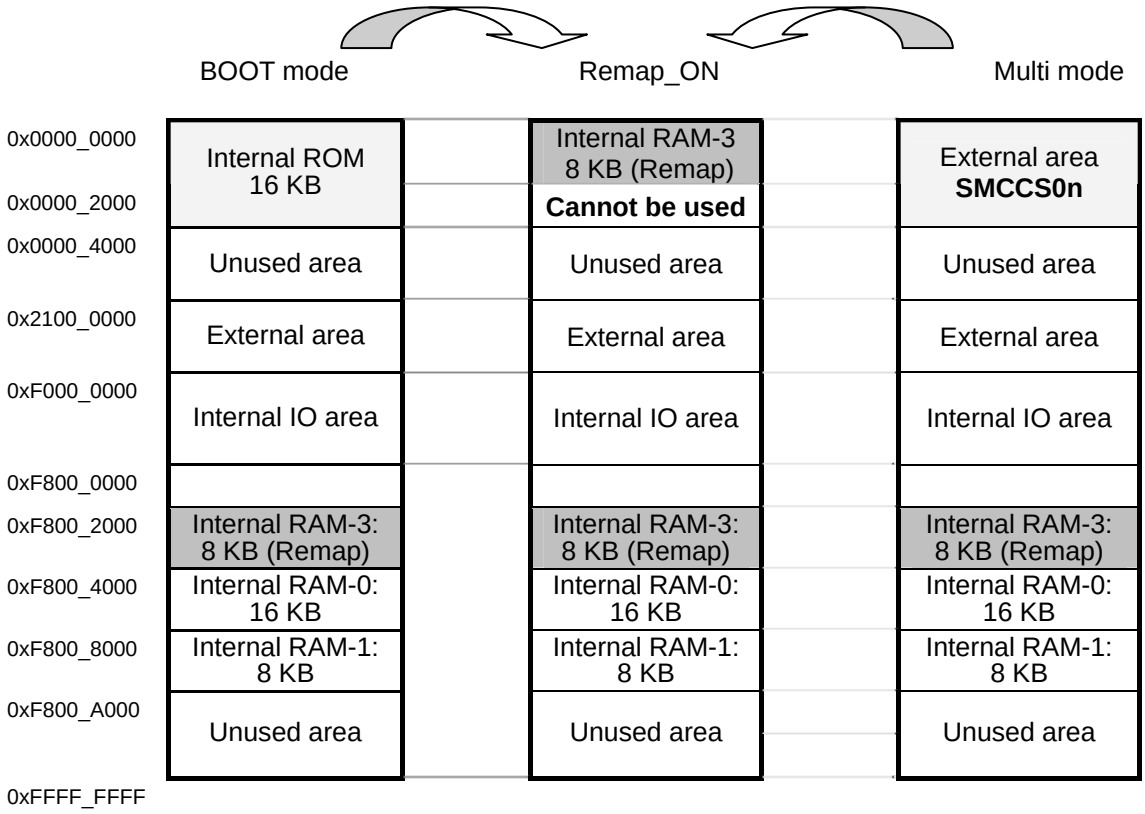


Figure 3.6.1 Memory map of BOOT mode

(2) The boot ROM elimination function

After the boot sequence is executed in BOOT mode, remapping is executed and the internal boot ROM area changes into RAM.



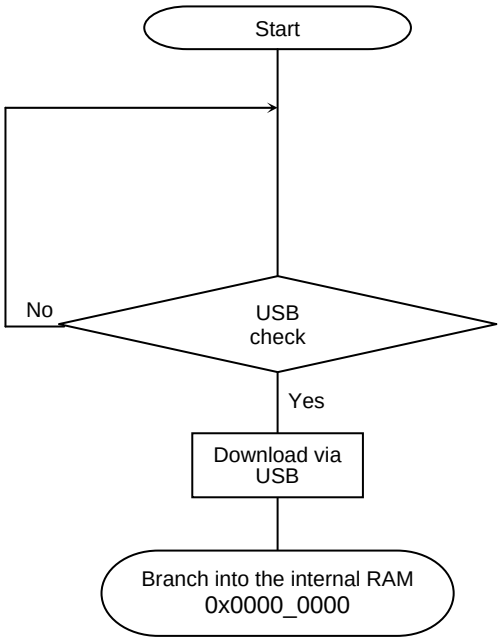
Note: Space between 0x0000_0000 and 0x0000_1FFF (8 KB) is a Remap area, and the internal RAM3 area will be accessed when Remap is set to Remap_ON (access to F8000_2000 also leads to the RAM3 area).

Figure 3.6.2 Memory map (details of boot mode and external area)

3.6.3 Outline of Boot Operation

USB can be selected as the transfer source of boot operation.

After reset, operation of the boot program on the internal boot ROM follows the flow chart shown in Figure 3.6.3. In any case, the user program is transferred from the source to the internal RAM, and branched into the internal RAM. The internal RAM is used in the same manner regardless of the transfer source as shown in Figure 3.6.4.



Note: When downloading the user program via USB, a USB device driver and special application software are needed on the PC.

Figure 3.6.3 Flow chart of internal boot ROM operation

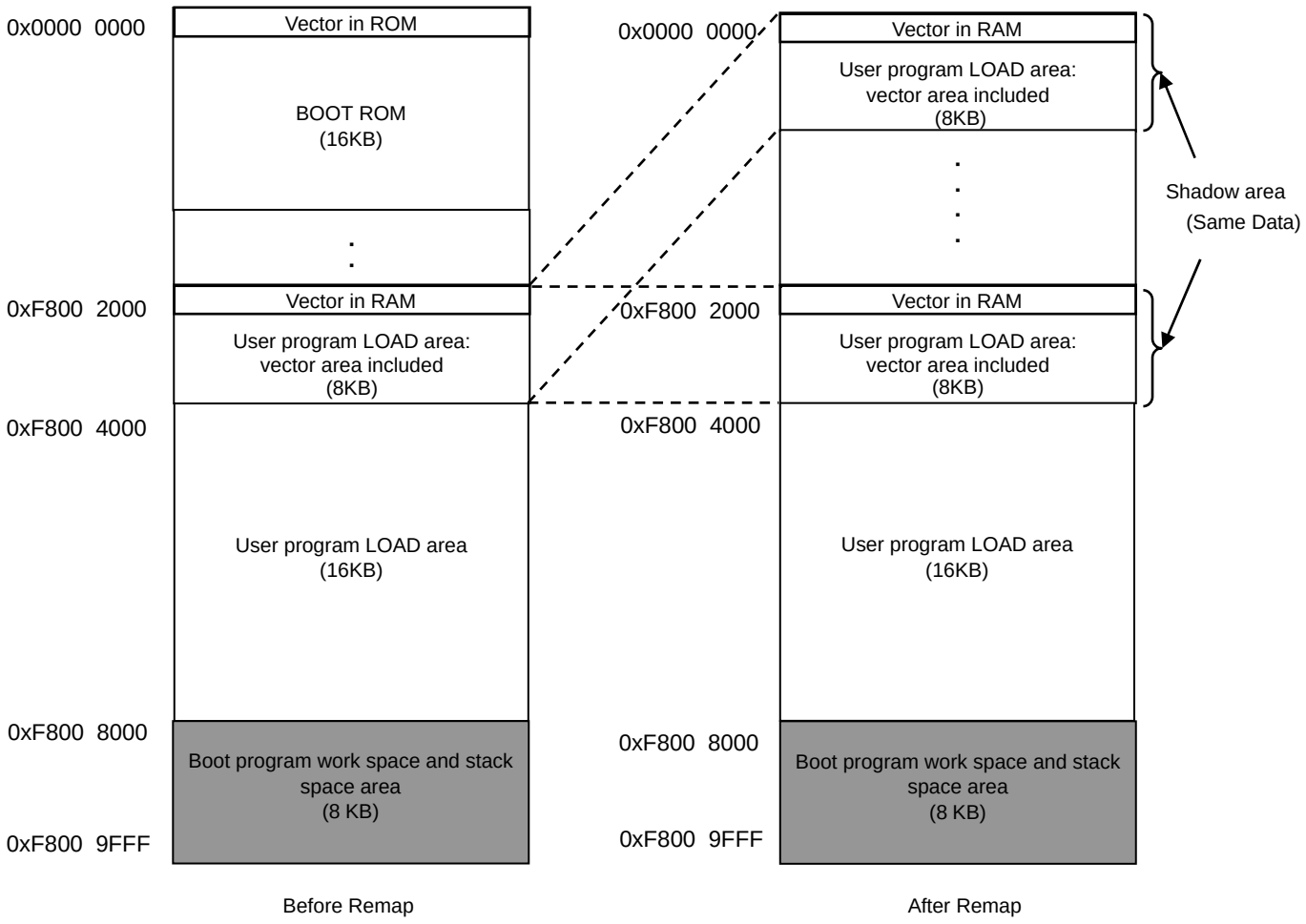


Figure 3.6.4 Use of the internal RAM of the boot program

Within the internal RAM, the area between 0xF800_8000 and 0xF800_9FFF is used as work and stack areas for executing the boot program. Therefore, the maximum size of the user program that can be loaded to the internal RAM is 24 KB.

Within 24 KB of the user program area between 0xF800_2000 and 0xF800_7FFF, the vector and program are written in an 8 KB space between 0xF800_2000 and 0xF800_3FFF.

The boot program loads user program into the user program area in the internal RAM. The boot program is loaded into the work space in the internal RAM. The loaded program executes remapping.

When the remap function is turned ON, the 8 KB space between 0xF800_2000 and 0xF800_3FFF can be accessed from the space between 0x0000_0000 and 0x0000_1FFF.

Refer to the chapter on the “system controller” for details of this function.

The boot program will branch to 0x0000_0000 of the last remapped RAM area (RESET vector).

As shown in Fig. 3.6.4, remapping assigns another vector addresses to the ROM area.

Ex. Before remapping 0xF800_2000 0xF800_2018

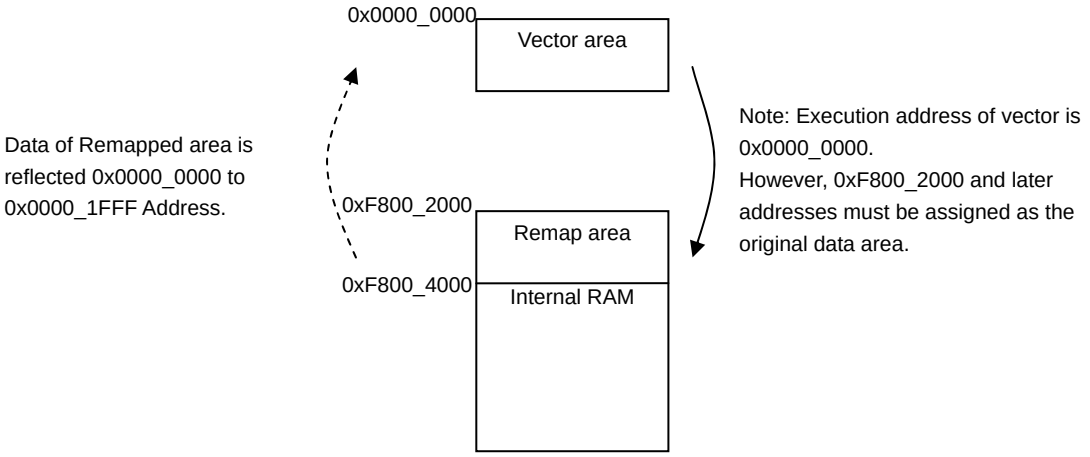
After remapping 0x0000_0000 (Reset vector) 0x0000_0018 (IRQ vector)

Therefore, the vector addresses to jump after running boot program must be assigned to 0xF800_2000 and later addresses.

3.6.3.1 Example of USB Boot

In boot from USB, user program vector is downloaded to 8KB of Remap area (0xF800_2000 to 0xF800_3FFF), program is downloaded to 16KB of internal RAM area (0xF800_4000 to 0xF800_7FFF).

Boot program remaps the area, and the data of Remap area is reflected to vector area (0x0000_0000 to 0x0000_1FFF). When the address jumps to 0x0000_0000 address, User program is started.



(1) CPU status and port settings

ARM926EJ™-S starts in supervisor mode after reset, and the boot program executes all programs in supervisor mode without any mode changes.

No port settings are required as ports used in the boot program are all dedicated pins.

Table 3.6.3 Port settings for the boot program

BOOT	PORT	I/O	Pin configuration by the boot program
USB	DDP	Input/Output	No settings required as dedicated pins are used.
	DDM	Input/Output	

(2) Control register settings by the boot program

Table 3.6.4 shows the control registers of internal circuits that are set by the boot program.

After the boot sequence, create a program while taking these setting values into account.

The stack pointer and the internal RAM including the area between 0xF800_8000 and 0xF800_9FFF remain in the state after execution of the boot program. Please reset them as appropriate before using.

Table 3.6.4 List of SFRs

Register name	Setting value	Description
SYSCR1	0x0002	Clock gear = 1/4
SYSCR2	0x0002	PLL clock is used (× 8)
SYSCR3	0x0087	
SYSCR4	0x0065	
SYSCR8	0x0030	Clock for USB Device Controller
REMAP	0x0001	Remap ON

【Important Notes】

Timer0 is used in the BOOT sequence. (Then Timer0control<TIM0EN> = 1: Timer0 operation is enable status.)It is possible that an interrupt of Timer0 may be generated when the program is running.

Before using Timer0, clear the interrupt by writing any values to Timer0IntClr<TIM0INTCLR>.

Note: The values to be set in the I/O registers for USB, INTC and DMAC are not described here. If these functions are needed in a user program, reconfigure each I/O register as necessary.

3.6.4 Download via USB

(1) Connection example

Figure 3.6.5 shows an example of USB connection (assuming that NOR Flash is program memory)

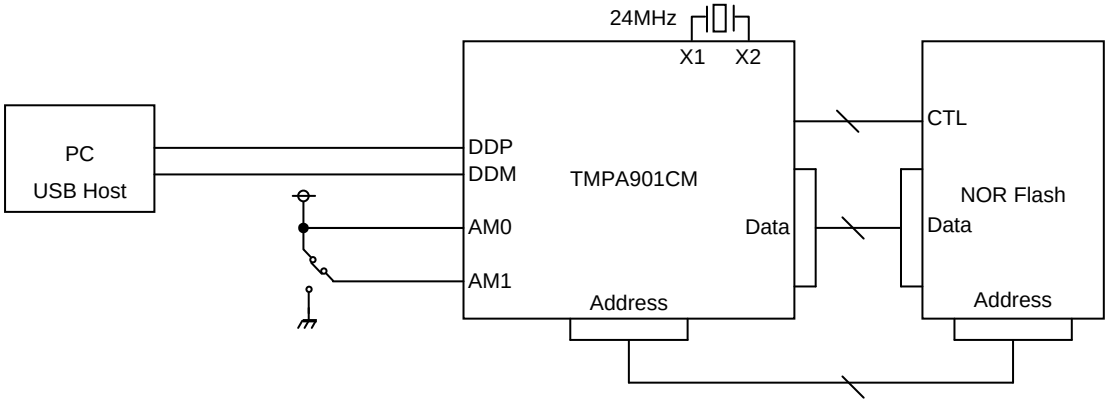


Figure 3.6.5 USB connection example

(2) Overview of the USB interface specifications

Set the oscillation frequency for the X1 and X2 pins to 24.00 MHz (± 100 ppm) when booting using USB.

The USB of this microcontroller supports high-speed communications. However, if the USB host does not support high-speed communications (USB 1.1 or older), full-speed communications will be carried out.

(The boot ROM function does not support clock supply from the USB clock pin X1USB.)

(For cautions on using the USB, refer to the chapter on the USB.)

Although there are four types of USB transfer, the following two types are used for the boot function.

Table 3.6.5 Transfer types used by the boot program

Transfer type	Description
Control Transfer	Used for transmitting standard requests and vendor requests.
Bulk Transfer	Used for responding to vendor requests and transmitting a user program.

The following shows an overview of the USB communication flow.

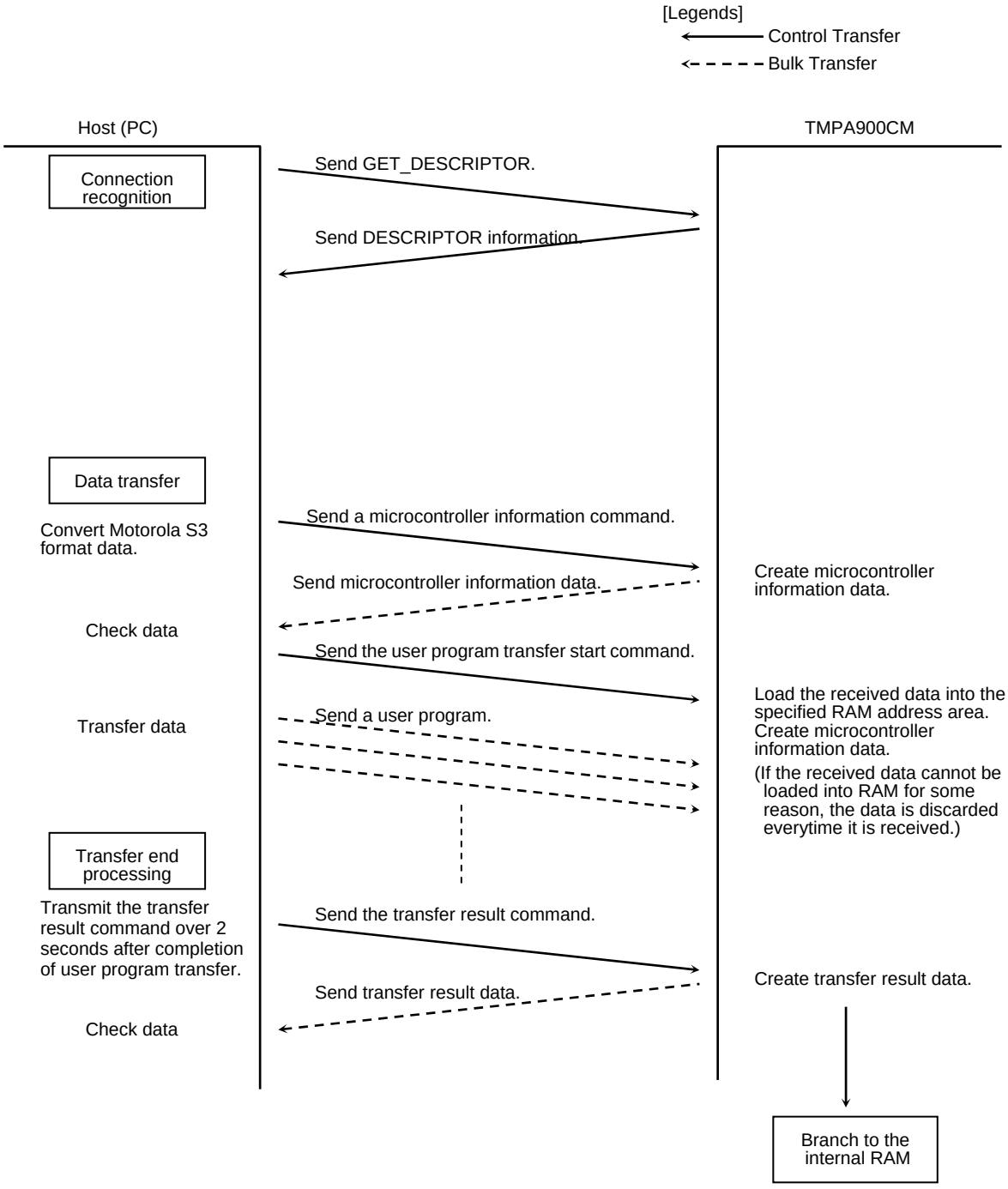


Figure 3.6.6 Overview of the overall flow

The following shows the connection of Vendor class request.

The table below shows the setup command data structure.

Table 3.6.6 Setup Command Data Structure

Field	Value	Description
bmRequestType	0x40	D7 0y0: Host to device D6-D5 0y10: Vendor D4-D0 0y00000: Device
bRequest	0x00, 0x02, 0x04	0x00: Microcontroller information 0x02: User program transfer start 0x04: User program transfer result
wValue	0x00~0xFFFF	Unique data number (Not used by the microcontroller)
wIndex	0x00~0xFFFF	Write size Used when starting user program transfer (user program transfer size)
wLength	0x0000	Fixed

The table below shows vendor request commands.

Table 3.6.7 Vendor Request Commands

Command	Vendor request value	Operation	Notes
Microcontroller information command	0x00	Device sends microcontroller information.	Microcontroller information data is sent by bulk IN transfer after the setup stage is completed.
User program transfer start command	0x02	Device starts receiving user program.	Set the transfer size of a user program in wIndex. The user program is received by bulk OUT transfer after the setup stage is completed.
User program transfer result command	0x04	Device sends the transfer result.	Transfer result data is transmitted as bulk data after the setup stage is completed.

The table below shows standard request commands.

Table 3.6.8 Standard request commands

Standard request	Response
GET_STATUS	Not supported
CLEAR_FEATURE	Not supported
SET_FEATURE	Not supported
SET_ADDRESS	Supported
GET_DESCRIPTOR	Supported
SET_DESCRIPTOR	Not supported
GET_CONFIGRATION	Not supported
SET_CONFIGRATION	Supported
GET_INTERFACE	Not supported
SET_INTERFACE	Not supported
SYNCH_FRAME	Ignored

The table below shows information to be returned by GET_DESCRIPTOR.

Table 3.6.9 Replies to GET_DISCRIPTOR

Device Descriptor

Field	Value	Description
Blength	0x12	18 bytes
BdescriptorType	0x01	Device descriptor
BcdUSB	0x0200	USB Version 2.0
BdeviceClass	0x00	Device class not in use
BdeviceSubClass	0x00	Sub command not in use
BdeviceProtocol	0x00	Protocol not in use
BmaxPacketSize0	0x40	EP0 maximum packet size is 64 bytes.
IdVendor	0x0930	Vendor ID
IdProduct	0x6504	Product ID (0)
BcdDevice	0x0001	Device version (v0.1)
Imanufacturer	0x00	Index value of string descriptor indicating the manufacturer name
Iproduct	0x00	Index value of string descriptor indicating the product name
IserialNumber	0x00	Index value of string descriptor indicating the product serial number
BnumConfigurations	0x01	There is one configuration.

* The descriptor information to be returned to the USB host should be modified as required by each application.

Configuration Descriptor

Field	Value	Description
bLength	0x09	9 bytes
bDescriptorType	0x02	Configuration descriptor
wTotalLength	0x0020	Total length (32 bytes) obtained by adding each configuration and endpoint descriptor
bNumInterfaces	0x01	There is one interface.
bConfigurationValue	0x01	Configuration number 1
iConfiguration	0x00	Index value of string descriptor indicating the configuration name (Not in use)
bmAttributes	0x80	Bus power
MaxPower	0x31	Maximum power consumption (49 mA)

Interface Descriptor

Field	Value	Description
bLength	0x09	9 bytes
bDescriptorType	0x04	Interface descriptor
bInterfaceNumber	0x00	Interface number 0
bAlternateSetting	0x00	Alternate setting number 0
bNumEndpoints	0x02	There are two endpoints.
bInterfaceClass	0xFF	Unique device
bInterfaceSubClass	0x00	
bInterfaceProtocol	0x50	BulkOnly protocol
iInterface	0x00	Index value of string descriptor indicating the interface name (Not in use)

* The descriptor information to be returned to the USB host should be modified as required by each application.

Endpoint Descriptor (When the USB host supports USB2.0)

Field	Value	Description
<Endpoint1>		
blength	0x07	7 bytes
bDescriptorType	0x05	Endpoint descriptor
bEndpointAddress	0x81	EP1 = IN
bmAttributes	0x02	Bulk transfer
wMaxPacketSize	0x0200	Payload 512 bytes
bInterval	0x00	(Ignored for bulk transfer)
<Endpoint2>		
bLength	0x07	7 bytes
bDescriptor	0x05	Endpoint descriptor
bEndpointAddress	0x02	EP2 = OUT
bmAttributes	0x02	Bulk transfer
wMaxPacketSize	0x0200	Payload 512 bytes
bInterval	0x00	(Ignored for bulk transfer)

Endpoint Descriptor (When the USB host supports USB1.1)

Field	Value	Description
<Endpoint1>		
blength	0x07	7 bytes
bDescriptorType	0x05	Endpoint descriptor
bEndpointAddress	0x81	EP1= IN
bmAttributes	0x02	Bulk transfer
wMaxPacketSize	0x0040	Payload 64 bytes
bInterval	0x00	(Ignored for bulk transfer)
<Endpoint2>		
bLength	0x07	7 bytes
bDescriptor	0x05	Endpoint descriptor
bEndpointAddress	0x02	EP2 = OUT
bmAttributes	0x02	Bulk transfer
wMaxPacketSize	0x0040	Payload 64 bytes
bInterval	0x00	(Ignored for bulk transfer)

* The descriptor information to be returned to the USB host should be modified as required by each application.

The table below shows information replied to the microcontroller information command.

Table 3.6.10 Information Replied to the Microcontroller Information Command

Microcontroller information	ASCII code
TMPA900CM	0x54,0x4D,0x50,0x41,0x39,0x30,0x30,0x43,0x4D,0x20,0x20,0x20,0x20,0x20,0x20

Note: produnct name in the Microcontroller information includes 6 spaces at the end of the product name.

Note: produnct name in the Microcontroller information is described TMPA900CM as TMPA900CMXBG series.

The table below shows information replied to the transfer result command.

Table 3.6.11 Information returned by the transfer result command

Transfer result	Value	Error condition
Normal termination	0x00	
User program not received	0x02	The user program transfer result is received without the user program transfer start command being received first.
Received file not in Motorola S3 format	0x04	The first data of a user program is not S (0x53).
Size of a received user program being larger than specified	0x06	The size of a received user program is larger than the value set in wIndex of the user program transfer start command.
Inadequate download address	0x08	The user program download address is not in the specified area.
Protocol error or errors other than above	0x0A	The user program transfer start or user program transfer result command is received first. A checksum error is detected in the Motorola S3 file. A record type error is detected in the Motorola S3 file. An error is detected in the DMA transfer.

(3) Description of the USB boot program operation

The boot program transfers data in Motorola S3 format sent from the PC to the internal RAM. The user program starts operating after data transfer is completed. The start address of the program is 0x0000_0000. Please refer to section 3.6.3 for details.
This function enables users to customize on-board programming control.

a. Operation procedure

1. Connect the USB cable.
2. Set both the AM0 and AM1 pins to 1 and reset the microcontroller.
3. After recognizing USB connection, the PC checks the information on the connected device using the GET_DESCRIPTOR command.
4. The PC sends the microcontroller information command by command transfer (vendor request).
5. Upon receiving the microcontroller information command, the boot program prepares microcontroller information in ASCII code.
6. The PC checks the microcontroller information data.
7. The PC sends the microcontroller transfer start command by command transfer (vendor request). After the setup stage is completed, the PC transfers the user program by bulk OUT transfer.
8. After the user program has been transferred, the PC waits for over two seconds and then sends the user program transfer result command by command transfer (vendor request).
9. Upon receiving the user program transfer result command, the boot program prepares for transmission of the transfer result value.
10. The PC checks the transfer result.
11. If the transfer results in failure, the boot program starts the error processing routine and will not automatically recover from it. In this case, terminate the device driver on the PC and retry from Step 2.

b. Notes on the user program format (binary)

1. After receiving the checksum of a record, the boot program waits for the start mark (0x53 for "S") of the next record. Even if data other than 0x53 is transmitted between records, it will be ignored.

Note: In USB transfers, the maximum object size that can be transferred is 64 KB since the write size is set by wIndex within the address range of 0x0000H to 0xFFFF.

3.6.5 Usage Note

Following are the note when use the BOOT ROM.

1. Using TIMER0

   Timer0 is used in the BOOT sequence. (Then Timer0control<TIM0EN> = 0y1: Timer0 operation is enable status.) It is possible that an interrupt of Timer0 may be generated when the program is running.

   Before using Timer0, clear the interrupt by writing any values to Timer0IntClr<TIM0INTCLR>.

2. USB connector

   The USB connector must not be connected or disconnected during USB boot.

3. Software on the PC

   A dedicated USB device driver and application software installed on the PC are needed for USB boot.

3.7 Interrupts

3.7.1 Functional Overview

- Supports 21 interrupt sources.
- Assigns 32 levels of fixed hardware (H/W) priorities to the interrupt sources (to be used if multiple interrupt requests of the same software priority level are made simultaneously).
- Enables to set 16 levels (0 to15) of software (S/W) interrupt priority for each interrupt source.
- Enables to mask hardware and software priority levels.
- Supports two types of interrupt requests: normal interrupt request (IRQ) and fast interrupt request (FIQ).
- Enables to generate software interrupts.

3.7.2 Block Diagram

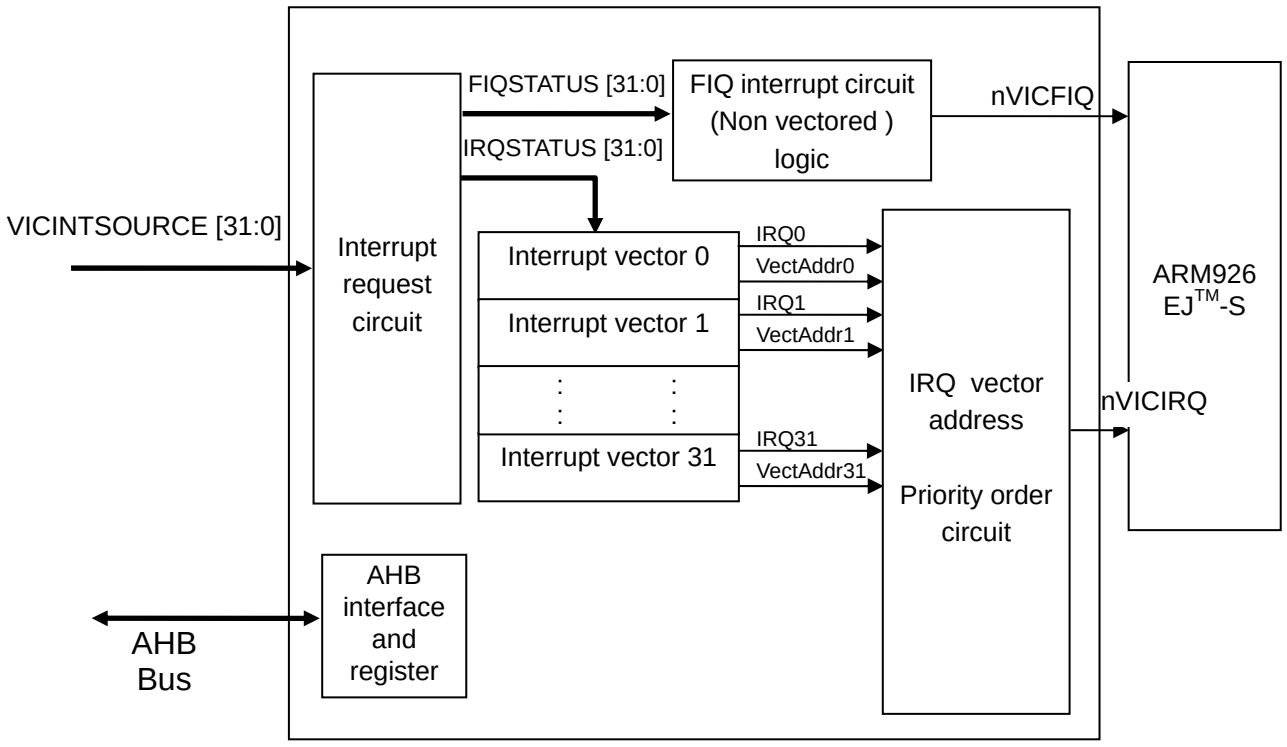


Figure 3.7.1 Block diagram

- Logic circuit of Interrupt request

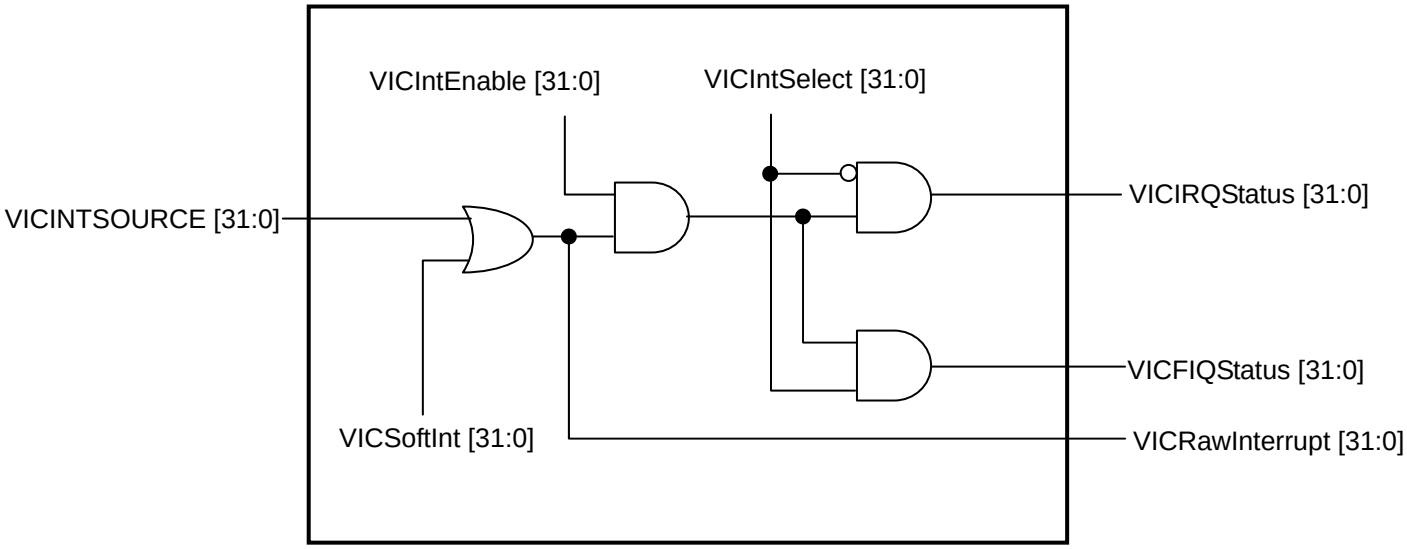


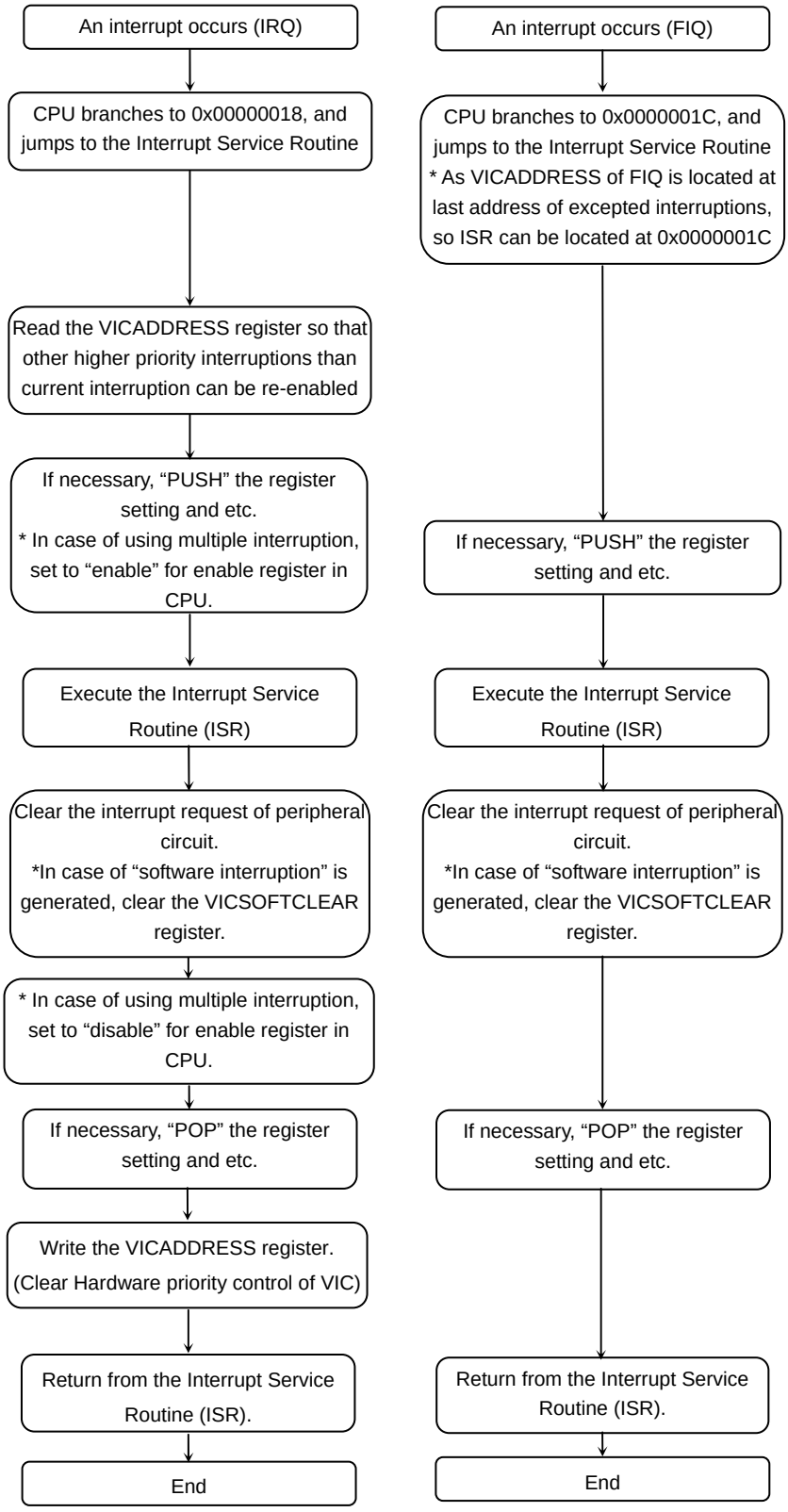
Figure 3.7.2 Status flag relation

3.7.3 Operational Description

For Interrupt Control(VIC), FIQ (Fast Interrupt Request) and IRQ (Interrupt Request) are available.

The TMPA901CM only has one FIQ source. FIQ is a low- latency interrupt and has the highest priority level. In handling FIQ, Interrupt Service Routine can be executed without checking which interrupt source is used.

- Interrupt vector flowchart



3.7.4 Interrupt Sources

Table 3.7.1 Interrupt sources

Interrupt source number (Note)	Interrupt source	Vector address
0	WDT	Vector Address 0
1	RTC	Vector Address 1
2	Timer01	Vector Address 2
3	Timer23	Vector Address 3
4	Timer45	Vector Address 4
5	GPIOD:INTA (TSI), INTB	Vector Address 5
6	I^2C ch0	Vector Address 6
7	Reserved	Vector Address 7
8	ADC	Vector Address 8
9	Reserved	Vector Address 9
10	UART ch0	Vector Address 10
11	UART ch1	Vector Address 11
12	SSP ch0	Vector Address 12
13	Reserved	Vector Address 13
14	NDFC	Vector Address 14
15	Reserved	Vector Address 15
16	DMA transfer error	Vector Address 16
17	DMA terminal count	Vector Address 17
18	LCDC	Vector Address 18
19	Reserved	Vector Address 19
20	LCDDA	Vector Address 20
21	USB Device	Vector Address 21
22	Reserved	Vector Address 22
23	I^2S	Vector Address 23
24	Reserved	Vector Address 24
25	Reserved	Vector Address 25
26	Reserved	Vector Address 26
27	USB Host	Vector Address 27
28	Reserved	Vector Address 28
29	Reserved	Vector Address 29
30	GPIOC (INT9)	Vector Address 30
31	GPIOA (KI0 to KI7)	Vector Address 31

Note: INTS[Num] shows the interrupt source signal. Ex: INTS[1]: RTC interrupt source signal.

3.7.5 SFRs

The following lists the SFRs:

Table 3.7.2 SFR (1/2) Base address = 0xF400_0000

Register Name	Address (base+)	Description
VICIRQSTATUS	0x0000	IRQ Status Register
VICFIQSTATUS	0x0004	FIQ Status Register
VICRAWINTR	0x0008	Raw Interrupt Status Register
VICINTSELECT	0x000C	Interrupt Select Register
VICINTENABLE	0x0010	Interrupt Enable Register
VICINTENCLEAR	0x0014	Interrupt Enable Clear Register
VICSOFTINT	0x0018	Software Interrupt Register
VICSOFTINTCLEAR	0x001C	Software Interrupt Clear Register
VICPROTECTION	0x0020	Protection Enable Register
VICSWPRIORITYMASK	0x0024	Software Priority Mask Register
–	0x0028	Reserved
VICVECTADDR0	0x0100	Vector Address 0 Register
VICVECTADDR1	0x0104	Vector Address 1 Register
VICVECTADDR2	0x0108	Vector Address 2 Register
VICVECTADDR3	0x010C	Vector Address 3 Register
VICVECTADDR4	0x0110	Vector Address 4 Register
VICVECTADDR5	0x0114	Vector Address 5 Register
VICVECTADDR6	0x0118	Vector Address 6 Register
–	0x011C	Reserved
VICVECTADDR8	0x0120	Vector Address 8 Register
–	0x0124	Reserved
VICVECTADDR10	0x0128	Vector Address 10 Register
VICVECTADDR11	0x012C	Vector Address 11 Register
VICVECTADDR12	0x0130	Vector Address 12 Register
–	0x0134	Reserved
VICVECTADDR14	0x0138	Vector Address 14 Register
–	0x013C	Reserved
VICVECTADDR16	0x0140	Vector Address 16 Register
VICVECTADDR17	0x0144	Vector Address 17 Register
VICVECTADDR18	0x0148	Vector Address 18 Register
–	0x014C	Reserved
VICVECTADDR20	0x0150	Vector Address 20 Register
VICVECTADDR21	0x0154	Vector Address 21 Register
–	0x0158	Reserved
VICVECTADDR23	0x015C	Vector Address 23 Register
–	0x0160	Reserved
–	0x0164	Reserved
–	0x0168	Reserved
VICVECTADDR27	0x016C	Vector Address 27 Register
–	0x0170	Reserved
–	0x0174	Reserved
VICVECTADDR30	0x0178	Vector Address 30 Register
VICVECTADDR31	0x017C	Vector Address 31 Register

Table 3.7.3 SFR (2/2)

Register Name	Address (base+)	Description
VICVECTPRIORITY0	0x0200	Vector Priority 0 Register
VICVECTPRIORITY1	0x0204	Vector Priority 1 Register
VICVECTPRIORITY2	0x0208	Vector Priority 2 Register
VICVECTPRIORITY3	0x020C	Vector Priority 3 Register
VICVECTPRIORITY4	0x0210	Vector Priority 4 Register
VICVECTPRIORITY5	0x0214	Vector Priority 5 Register
VICVECTPRIORITY6	0x0218	Vector Priority 6 Register
–	0x021C	Reserved
VICVECTPRIORITY8	0x0220	Vector Priority 8 Register
–	0x0224	Reserved
VICVECTPRIORITY10	0x0228	Vector Priority 10 Register
VICVECTPRIORITY11	0x022C	Vector Priority 11 Register
VICVECTPRIORITY12	0x0230	Vector Priority 12 Register
–	0x0234	Reserved
VICVECTPRIORITY14	0x0238	Vector Priority 14 Register
–	0x023C	Reserved
VICVECTPRIORITY16	0x0240	Vector Priority 16 Register
VICVECTPRIORITY17	0x0244	Vector Priority 17 Register
VICVECTPRIORITY18	0x0248	Vector Priority 18 Register
–	0x024C	Reserved
VICVECTPRIORITY20	0x0250	Vector Priority 20 Register
VICVECTPRIORITY21	0x0254	Vector Priority 21 Register
–	0x0258	Reserved
VICVECTPRIORITY23	0x025C	Vector Priority 23 Register
–	0x0260	Reserved
–	0x0264	Reserved
–	0x0268	Reserved
VICVECTPRIORITY27	0x026C	Vector Priority 27 Register
–	0x0270	Reserved
–	0x0274	Reserved
VICVECTPRIORITY30	0x0278	Vector Priority 30 Register
VICVECTPRIORITY31	0x027C	Vector Priority 31 Register
VICADDRESS	0x0F00	Vector Address Register

1. VICIRQSTATUS (IRQ Status Register)

Address = (0xF400_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	IRQStatus	RO	0x00000000	IRQ interrupt status after masked (for each bit) 0y0: Interrupt is inactive. 0y1: Interrupt is active.

[Description]

a. <IRQStatus>

This bit shows IRQ interrupt status after masked. Refer the Figure 3.7.2 Status flag relation .

IRQStatus [31:0] correspond to interrupt numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please.

Example: When bit 0 of this register is set to 1, a WDT interrupt (interrupt source number 0) has been requested.

2. VICFIQSTATUS (FIQ Status Register)

Address = (0xF400_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	FIQStatus	RO	0x00000000	FIQ interrupt status after masked (for each bit) 0y0: Interrupt is inactive. 0y1: Interrupt is active.

[Description]

a. <FIQStatus>

This bit shows FIQ interrupt status after masked. Refer the Figure 3.7.2 Status flag relation.

FIQStatus [31:0] correspond to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

Example: When bit 0 of this register is set to 1, a WDT interrupt (interrupt source number 0) has been requested.

3. VICRAWINTR (Raw Interrupt Status Register)

Address = (0xF400_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	RawInterrupt	RO	Undefined	IRQ interrupt status before masked (for each bit) 0y0: Interrupt is inactive. 0y1: Interrupts is active.

[Description]

a. <RawInterrupt>

This bit shows IRQ interrupt status before masked. Refer the Figure 3.7.2 Status flag relation .

RawInterrupt [31:0] correspond to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

Example: When bit 0 of this register is set to 1, a WDT interrupt (interrupt source number 0) has been requested.

4. VICINTSELECT (Interrupt Select Register)

Address = (0xF400_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	IntSelect	R/W	0x00000000	Selects interrupt type (for each bit) 0y0: IRQ 0y1: FIQ

[Description]

a. <IntSelect>

This bit controls Selects interrupt type.

IntSelect bit must set before interrupt generation.

IntSelect [31:0] correspond to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

Example: When bit 0 of this register is set to 1, the WDT interrupt (interrupt source number 0) is set to be of the FIQ type.

Note: Since this LSI supports only one FIQ source, only one of the bits in this register can be set to 1. Before changing the setting of this register, be sure to disable the relevant interrupts. Do not change the setting of this register while the interrupt is active and enabled.

5. VICINTENABLE (Interrupt Enable Register)

Address = (0xF400_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	IntEnable	RO	0x00000000	Interrupt enable (for each bit) 0y0: Disable 0y1: Enable

Address = (0xF400_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	IntEnable	WO	0x00000000	Interrupt enable (for each bit) 0y0: Invalid 0y1: Enable

[Description]

a. <IntEnable>

READ: Status read register of Interrupt Enable/Disable

WRITE: Setting register of Interrupt Enable

This register can be set only from disable to enable. Disable setting is controlled by VICINTENCLEAR register.

IntEnable [31:0] correspond to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

Example: When bit 0 of this register is set to 1, the WDT interrupt (interrupt source number 0) is enabled.

6. VICINTENCLEAR (Interrupt Enable Clear Register)

Address = (0xF400_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	IntEnable Clear	WO	Undefined	Interrupt disable (for each bit) 0y0: Invalid 0y1: Disable

[Description]

a. <IntEnable Clear>

This bit controls interrupt disable.

Enable setting of VICINTENABLE register can be cleared, and interruption is disabled.

IntEnable Clear [31:0] corresponds to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

7. VICSOFTINT (Software Interrupt Register)

Address = (0xF400_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	SoftInt	WO	0x00000000	Software interrupt (for each bit) 0y0: Invalid 0y1: Generate a software interrupt

Address = (0xF400_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	SoftInt	RO	0x00000000	Software interrupt (for each bit) 0y0: Inactive 0y1: Active

[Description]

a. <SoftInt>

READ: Status register for Active/Inactive of software interruption.

WRITE: Software interruption Active/Inactive control register

Set to “1” to each bit, and then software interruption is generated.

SoftInt[31:0] correspond to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

8. VICSOFTINTCLEAR (Software Interrupt Clear Register)

Address = (0xF400_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	SoftIntClear	WO	Undefined	Software interrupt disable (for each bit) 0y0: Invalid 0y1: Disable

[Description]

a. <SoftIntClear>

This bit controls “disable” for software interruption.

Software interruption of VICSOFTINT register can be disabled.

SoftIntClear [31:0] correspond to interrupt source numbers 31 to 0, respectively.

About the information for interrupt sources of each circuit, refer the Table 3.7.1 Interrupt sources please

9. VICPROTECTION (Protection Enable Register)

Address = (0xF400_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	Protection	R/W	0y0	Protect mode enable : 0y0: Disable 0y1: Enable

[Description]

a. <Protection>

This bit controls protect mode enable.

When protection is enabled, the registers of the interrupt controller can only be accessed in privileged mode.

Read/ write operations are available only in privilege mode.

10. VICSWPRIORITYMASK (Software Priority Mask Register)

Address = (0xF400_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	SWPriorityMask	R/W	0xFFFF	Masks software priority level 0y0: Mask 0y1: Do not mask

[Description]

a. <SWPriorityMask>

This register can be set the software priority level.

SWPriorityMask [15:0] correspond to priority levels 15 to 0, respectively.

Example: When SWPriorityMask [15:0] = 0xFF7F, interrupts of priority level 7 are masked.

11. VICVECTADDR0 (Vector Address 0 Register)

Address = (0xF400_0000) + (0x0100)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	VectorAddr 0	R/W	0x00000000	ISR address for interrupt source 0

ISR: Interrupt Service Routine

[Description]

a. <VectorAddr 0>

This register can be set the address for Interrupt Service Routine of interrupt sources.
Before changing the setting of this register, be sure to disable the relevant interrupts.

- VICVECTADDRn (Vector Address n Register)(n = 0 to 6, 8, 10 to12, 14, 16 to 18, 20, 21,23, 27, 30, 31)

  The structure and description of these registers are same as VICVECTADDR0.

  Please refer to the description of VICVECTADDR0.

  For the names and addresses of these registers, please refer to Table 3.7.2.

12. VICVECTPRIORITY0 (Vector Priority 0 Register)

Address = (0xF400_0000) + (0x0200)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	VectPriority	R/W	0y1111	S/W Priority level for interrupt source 0: 0y0000 to 0y1111

[Description]

a. <VectPriority>

This register can be set the software priority level of IRQ.

0y0000 is highest level, and can set 16 level (0y0000 to 0y1111).

If multiple interrupt requests of the same software priority level occur simultaneously, the hardware priority is used to determine the interrupt to be generated.

The hardware priority is assigned according to interrupt source numbers: interrupt source number 0 has the highest priority and interrupt source number 31 has the lowest priority.

- VICVECTPRIORITYn (Vector Priority n Register)(n = 0 to 6, 8, 10 to12, 14, 16 to 18, 20, 21,23, 27, 30, 31)

  The structure and description of these registers are same as VICVECTPRIORITY0.

  Please refer to the description of VICVECTPRIORITY0.

  The name and address of these registers, please refer to Table 3.7.2.

13. VICADDRESS (Vector Address Register)

Address = (0xF400_0000) + (0x0F00)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	VectAddr	R/W	0x00000000	Address of the currently active Interrupt Service Routine (ISR)

[Description]

a. <VectAddr>

This register can be read current active address of Interrupt Service Routine.

And current interruption can be clear.

Read: return the address of the currently active Interrupt Service Routine (ISR.)

Write: Writing any data to this register clears the current interrupt.

Note: A read of this register must only be performed when there is an active interrupt.

A write of this register must only be performed at the end of an ISR.

3.8 DMAC (DMA Controller)

3.8.1 Functional Overview

The DMA controller has the following features:

Table 3.8.1 DMA controller functions

Item	Function		Description
Number of channels	8 ch		
DMA start	Hardware request		16 types of DMA requests for peripheral IPs. Refer to Table 3.8.2.
	Software request		Activated by writing values into DMACSoftBReq
Bus master	32 bits × 2 (AHB)		DMA1, DMA2
Priority	DMA channel 0 (high) to DMA channel 7 (low)		Hardware-fixed
FIFO	4 words × 8 ch		
Bus width	8/16/32 bits		Source and destination can be programmed separately.
Burst size	1/4/8/16/32/64/128/256		
transfer count	~4095		
Address	Source address	incr / no-incr	Address wrapping is not supported.
	Destination address	incr / no-incr	
Endian	Only little endian is supported.		
Transfer type	Peripheral circuit (register) to peripheral circuit (register)		DMA cannot start by hardware request in memory to memory transfer. Refer to the description of DMACCxConfiguration register for details.
	Peripheral circuit (register) to memory		
	Memory to peripheral circuit (register)		
	Memory to memory		
Interrupt	Terminal count interrupt Transfer error interrupt		
Special function	Scatter/gather function		

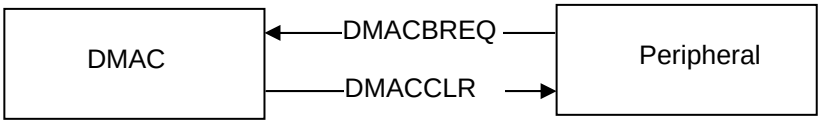
- DMA Transfer Types

	DMA Transfer Direction	DMA Request Generator	DMA Request Used	Description		
1	Memory-to-Peripheral	Peripheral	Burst request	1: Use bust request in all transactions 2: When the single request, set DMAC busrt to 1		
2	Peripheral-to-Memory	Peripheral	Burst request/ Single request (Note 1)	For transactions that are not an integral multiple of the burst size, use both the burst and single request signals. The amount of data left to transfer ≥ Burst size :Use burst transfer The amount of data left to transfer < Burst size : Use single transfer		
3	Memory-to-Memory (Note 2)	DMAC	None	Start condition: When enabled, the DMA channel commences transfers without DMA requests. Stop conditions: All transfer data has finished transfer. Disable DMAC channel (Note 2)		
4	Peripheral-to-Peripheral	Source peripheral	Burst request/ Single request (Note 1)	Transfer size	Source side	Destination side
				1)Integral multiple of the burst size	Burst request	Burst request
				2) Single transfer	Single request	
		Destination peripheral	Burst request	3) Not imtegral multiple of the burst size	Burst request Single request	

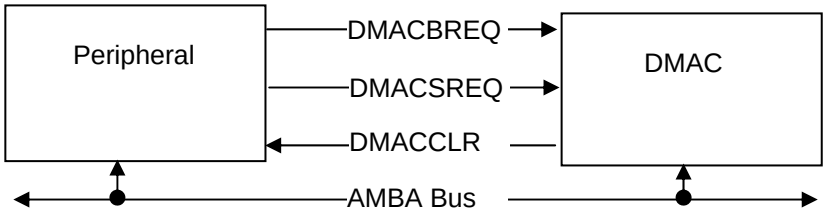
Note 1: Peripheral that can use the single request: UART and LCDDA.

Note 2:You must program memory-to-memory transfers with a low channel priority , otherwise the other DMA channels cannot access the bus until the huge memory-to-memory transfer has finished.

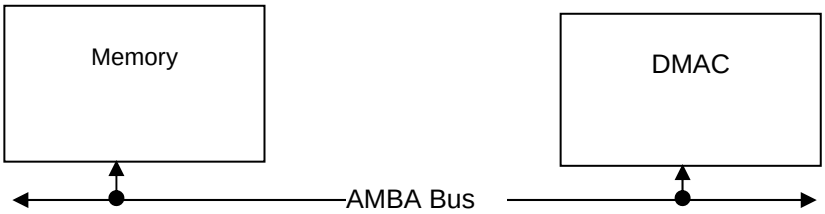
1. Memory-to-peripheral



2. Peripheral-to-Memory

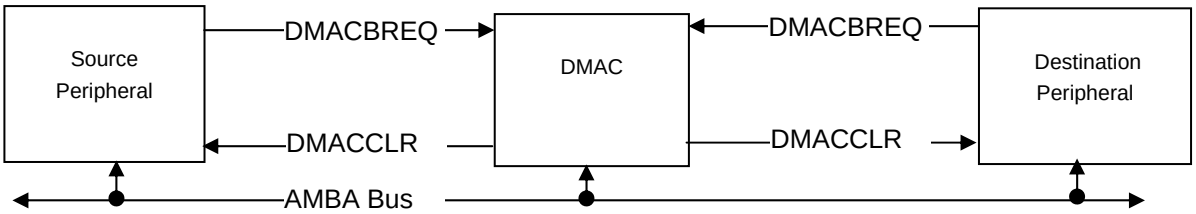


3. Memory-to-Memory

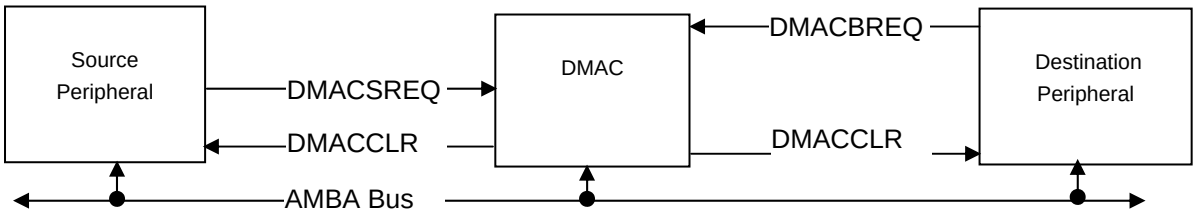


4. Peripheral-to-peripheral

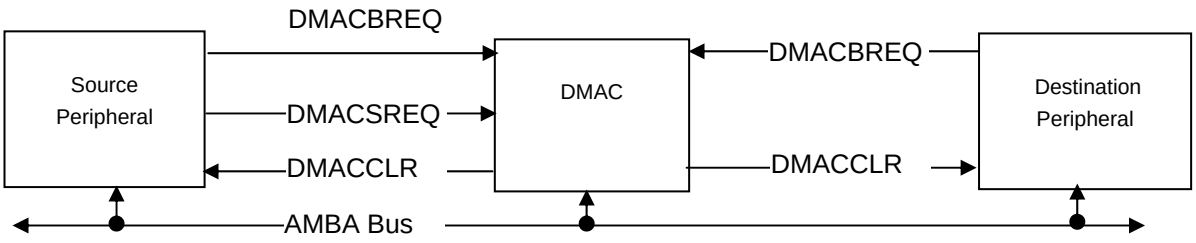
1) Integral multiple of the burst size



2) Single transfer



3) Not Integral Multiple of the burst size



3.8.2 Block Diagram

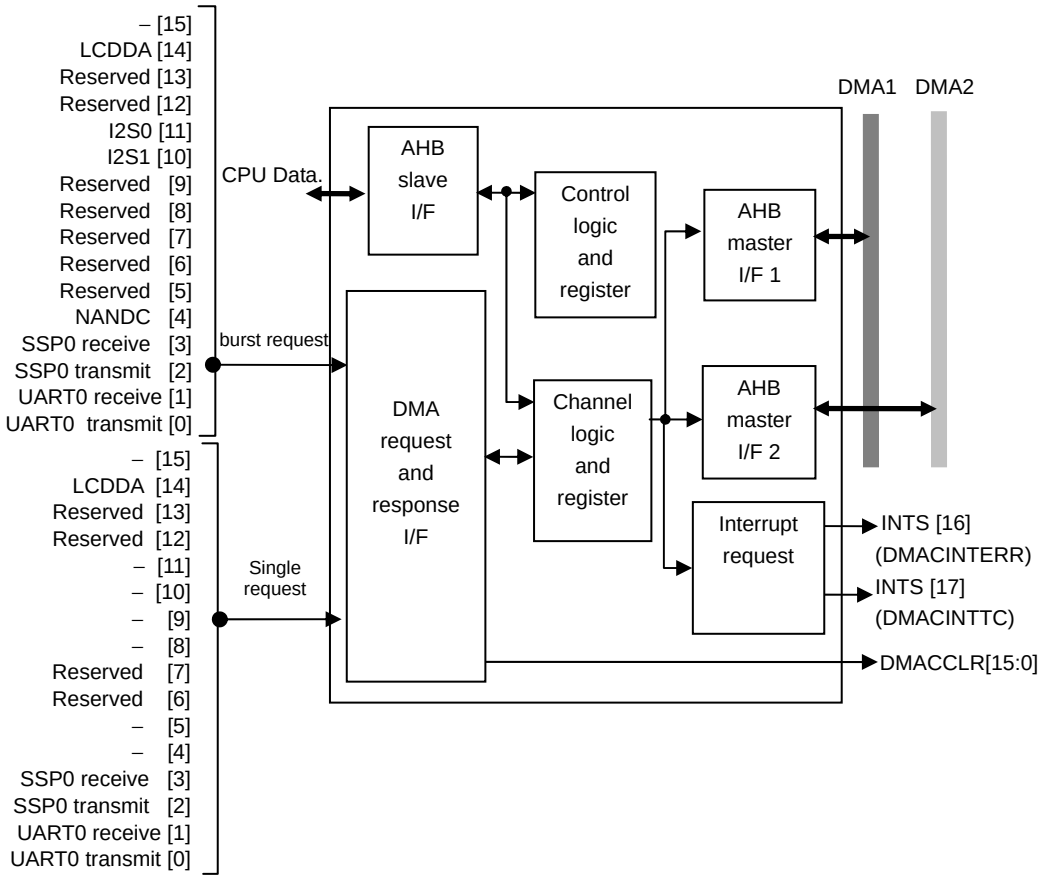


Table 3.8.2 DMA request number chart

DMA Request Number	Peripheral	
	Burst	Single
0	UART0 transmit	UART0 transmit
1	UART0 receive	UART0 receive
2	SSP0 transmit	SSP0 transmit
3	SSP0 receive	SSP0 receive
4	NANDC	–
5	Reserved	–
6	Reserved	Reserved
7	Reserved	Reserved
8	Reserved	–
9	Reserved	–
10	I2S1	–
11	I2S0	–
12	Reserved	Reserved
13	Reserved	Reserved
14	LCDDA	LCDDA
15	–	–

3.8.3 Register descriptions

The following lists the SFRs:

Table 3.8.3 SFR Base address = 0xF410_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

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.

1. DMACIntStatus (DMAC Interrupt Status Register)

Address = (0xF410_0000) + (0x0000)

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

[Description]

a. <IntStatus[7:0]>

Indicates the status of the DMAC interrupt after reflecting the status of the terminal count interrupt enable register and error interrupt enable register. An interrupt is requested when a transfer error occurs or the counter completes counting.

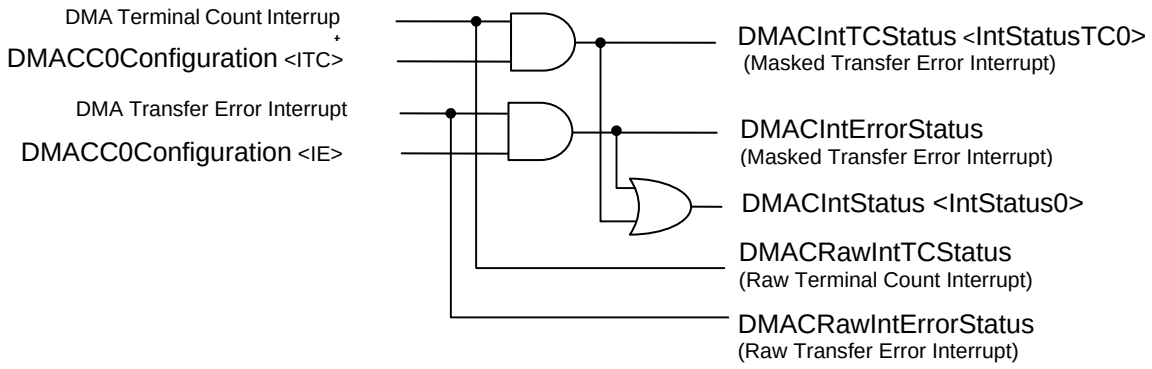


Figure 3.8.1 Block diagram for Interrupt

2. DMACIntTCStatus (DMAC Interrupt Terminal Count Status Register)

Address = (0xF410_0000) + (0x0004)

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

[Description]

a. <IntStatusTC[7:0]>

Indicates the enabled state of the terminal count interrupt.

3. DMACIntTCClear (DMAC Interrupt Terminal Count Clear Register)

Address = (0xF410_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	IntTCClear7	WO	0y0	DMAC channel 7 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[6]	IntTCClear6	WO	0y0	DMAC channel 6 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[5]	IntTCClear5	WO	0y0	DMAC channel 5 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[4]	IntTCClear4	WO	0y0	DMAC channel 4 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[3]	IntTCClear3	WO	0y0	DMAC channel 3 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[2]	IntTCClear2	WO	0y0	DMAC channel 2 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[1]	IntTCClear1	WO	0y0	DMAC channel 1 terminal count interrupt clear 0y0 Invalid 0y1 Clear
[0]	IntTCClear0	WO	0y0	DMAC channel 0 terminal count interrupt clear 0y0 Invalid 0y1 Clear

[Description]

a. <IntTCClear[7:0]>

Writing 1 to each bit of this register clears the corresponding bit in the DMACIntTCStatus register.

4. DMACIntErrorStatus (DMAC Interrupt Error Status Register)

Address = (0xF410_0000) + (0x000C)

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

[Description]

a. <IntErrStatus[7:0]>

These bits shows status of Raw Error interrupt.

i

5. DMACIntErrClr (DMAC Interrupt Error Clear Register)

Address = (0xF410_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	IntErrClr7	WO	0y0	DMAC channel 7 error interrupt clear 0y0: Invalid 0y1: Clear
[6]	IntErrClr6	WO	0y0	DMAC channel 6 error interrupt clear 0y0: Invalid 0y1: Clear
[5]	IntErrClr5	WO	0y0	DMAC channel 5 error interrupt clear 0y0: Invalid 0y1: Clear
[4]	IntErrClr4	WO	0y0	DMAC channel 4 error interrupt clear 0y0: Invalid 0y1: Clear
[3]	IntErrClr3	WO	0y0	DMAC channel 3 error interrupt clear 0y0: Invalid 0y1: Clear
[2]	IntErrClr2	WO	0y0	DMAC channel 2 error interrupt clear 0y0: Invalid 0y1: Clear
[1]	IntErrClr1	WO	0y0	DMAC channel 1 error interrupt clear 0y0: Invalid 0y1: Clear
[0]	IntErrClr0	WO	0y0	DMAC channel 0 error interrupt clear 0y0: Invalid 0y1: Clear

[Description]

a. <IntErrClr[7:0]>

0y1: Clear Error interrupt request.

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

Address = (0xF410_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7]	RawIntTCS7	RO	0y0	DMAC channel 7 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[6]	RawIntTCS6	RO	0y0	DMAC channel 6 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[5]	RawIntTCS5	RO	0y0	DMAC channel 5 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[4]	RawIntTCS4	RO	0y0	DMAC channel 4 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[3]	RawIntTCS3	RO	0y0	DMAC channel 3 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[2]	RawIntTCS2	RO	0y0	DMAC channel 2 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[1]	RawIntTCS1	RO	0y0	DMAC channel 1 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested
[0]	RawIntTCS0	RO	0y0	DMAC channel 0 terminal count interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested

[Description]

a. <RawIntTCS[7:0]>

The status of raw interrupt terminal count before an interrupt enable

7. DMACRawIntErrorStatus (DMAC Raw Error Interrupt Status Register)

Address = (0xF410_0000) + (0x0018)

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

[Description]

a. <RawIntErrS[7:0]>

The status of raw error interrupt before an interrupt enable

8. DMACEnbldChns (DMAC Enabled Channel Register)

Address = (0xF410_0000) + (0x001C)

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

[Description]

a. <EnabledCH[7:0]>

0y0: Applicable channel bit is cleared when DMA transfer has finished.

0y1: Applicable channel DMA is in the enable state.

9. DMACSoftBReq (DMAC Software Burst Request Register)

Address = (0xF410_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:15]	–	–	Undefined	Read as undefined. Write as zero.
[14]	SoftBReq14	R/W	0y0	DMA burst request of LCDDA by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[13:10]	Reserved	R/	Undefined	Read as undefined. Write as zero.
[11]	SoftBReq11	WR/W	0y0	DMA burst request of I^2S0 by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[10]	SoftBReq10	R/W	0y0	DMA burst request of I^2S1 by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[9:5]	Reserved	R/	Undefined	Read as undefined. Write as zero.
[4]	SoftBReq4	WR/W	0y0	DMA burst request of NANDC0 by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[3]	SoftBReq3	R/W	0y0	DMA burst request of SSP0 receive by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[2]	SoftBReq2	R/W	0y0	DMA burst request of SSP0 transmit by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[1]	SoftBReq1	R/W	0y0	DMA burst request of UART0 receive by software 0y0: Invalid when data write 0y1: Generate a DMA burst request
[0]	SoftBReq0	R/W	0y0	DMA burst request of UART0 transmit by software 0y0: Invalid when data write 0y1: Generate a DMA burst request

[Description]

a. <SoftBReq[14:0]>

This register is used to set DMA burst transfer requests by software. Upon completion of a DMA burst transfer, the corresponding bit of SoftBReq [14:0] is cleared.

The state of the burst-request is led when leading. (The demand by the peripheral circuitry is included.).

Note: Making DMA request by software and a hardware peripheral simultaneously is prohibited.

10. DMACSoftSReq (DMAC Software Single Request Register)

Address = (0xF410_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	Undefined	Read as undefined. Write as zero.
[14]	SoftSReq14	R/W	0y0	DMA single request by software for LCDDA 0y0: Invalid when data write 0y1: Generate a DMA single request
[13:4]	Reserved	R/W	Undefined	Read as undefined. Write as zero.
[3]	SoftSReq3	R/W	0y0	DMA single request of SSP0 recevie by software 0y0: Invalid when data write 0y1: Generate a DMA single request
[2]	SoftSReq2	R/W	0y0	DMA single request of SSP0 transimit by software 0y0: Invalid when data write 0y1: Generate a DMA single request
[1]	SoftSReq1	R/W	0y0	DMA single request by software for UART0 receive 0y0: Invalid when data write 0y1: Generate a DMA single request
[0]	SoftSReq0	R/W	0y0	DMA single request b software for UART0 transmit 0y0: Invalid when data write 0y1: Generate a DMA single request

[Description]

a. <SoftSReq[14:0]>

This register is used to configure the DMA single transfer requests by software. Upon completion of a DMA single transfer, the corresponding bit of SoftSReq [14:0] is cleared.

The state of a single request is led when leading. (The demand by the peripheral circuitry is included.).

Note: Making DMA request by software and a hardware peripheral simultaneously is prohibited.

11. DMACConfiguration (DMAC Configuration Register)

Address = (0xF410_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2]	M2	R/W	0y0	DMA2 endianness 0y0 Little endian mode 0y1 Reserved
[1]	M1	R/W	0y0	DMA1 endianness 0y0 Little endian mode 0y1 Reserved
[0]	E	R/W	0y0	DMA circuit control 0y0: Stopped 0y1: Active

[Description]

a. <E>

Write/read operation can be executed to any of the DMAC registers only when the DMA circuit is active. To perform DMA operation, the DMA circuit must always be active.

12. DMACC0SrcAddr (DMAC Channel0 Source Address Register)

Address = (0xF410_0000) + (0x0100)

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

[Description]

a. <SrcAddr>

Software configures each register directly before the channel is enabled. When the DMAchannel is enabled, the register is updated as the destination address is incremented and by following the linked list when a complete packet of data has been transferred. Reading the register when the channel is active does not provide useful information. This is because by the time the software has processed the value read, the channel might have progressed. It is intended to be read-only when a channel has stopped. In this case, it shows the destination address of the last item read.

When transfer is taking place, don't update this register. If you want to change the channel configurations, you must disable the channel first with the DMACCxConfiguration register and then reconfigure the relevant register.

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

  The DMACCxSrcAddr regusters have the same structure as DMACC0SrcAddr.

  Please refer to the descriptions of DMACC0SrcAddr.

  For the names and addresses of these registers, please refer to Table 3.8.3.

13. DMACC0DestAddr (DMAC Channel0 Destination Address Register)

Address = (0xF410_0000) + (0x0104)

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

[Description]

a. <DestAddr>

When transfer is taking place, don't update this register. If you want to change the channel configuration, you must disable the channel first with the DMACCxConfiguration register and then reconfigure the relevant registers.

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

The DMACCxDestAddr registers have the same structure as DMACC0DestAddr.

Please refer to the description of DMACC0DestAddr.

The name and addresses of these registers, please refer to Table 3.8.3.

14. DMACC0LLI (DMAC Channel0 Linked List Item Register)

Address = (0xF410_0000) + (0x0108)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	LLI	R/W	0x00000000	Set the start address of the next transfer information
[1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	LM	R/W	0y0	AHB master for storing LLI: 0y0: DMA1 0y1: DMA2

[Description]

a. <LLI>

The value set to <LLI> must be within 0xFFFF_FFF0.

If the LLI is 0, then the current LLI is the last in the chain, and the DMA channel is disabled after all DMA transfers associated with it are completed.

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

The DMACCxLLI registers have the same structure as DMACC0LLI.

Please refer to the description of DMACC0LLI.

The names and addresses of these registers, please refer to Table 3.8.3.

15. DMACC0Control (DMAC Channel0 Control Register)

Address = (0xF410_0000) + (0x010C)

Bit	Bit Symbol	Type	Reset Value	Description
[31]	I	R/W	0y0	Terminal count interrupt enable register when using the scatter/gather function 0y0: Disable 0y1: Enable
[30]	Prot[3]	R/W	0y0	Control cache permission HPROT[3] 0y0: Noncacheable 0y1: Cacheable
[29]	Prot[2]	R/W	0y0	Control buffer permission HPROT[2] 0y0: Nonbufferable 0y1: Bufferable
[28]	Prot[1]	R/W	0y0	Control privileged mode HPROT[1] 0y0: User mode 0y1: Privileged mode
[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 transfer count

[Description]

The description below applies to all channels.

a. <I>

It is enable register of terminal count interrupt.

Terminal count interrupt is generated by setting <I>=1 and DAMCCxConfiguration Register<ITC>=1.This bit is set to enable in DMAC configuration flow of the final transfer when using Scatter/gather function, and it is possible to generate terminal count interrupt when the only final transfer is performed.

To generate an interrupt at transferring, this bit also must be set to enable by setting to "1".

b. Prot[3]

Control cache permission HPROT[3]

0y0: Noncacheable

0y1: Cacheable

c. Prot[2]

Control buffer permission HPROT[2]

0y0: Nonbufferable

0y1: Bufferable

d. Prot[1]

Control privileged mode HPROT[1]

0y0: User mode

0y1: Privileged mode

e. DI

Increment the transfer destination address

0y0: Do not increment

0y1: Increment

f. SI

Increment the transfer source address

0y0: Do not increment

0y1: Increment

g. D

Transfer destination AHB Master

0y0: DMA1

0y1: DMA2

h. S

Transfer source AHB Master

0y0: DMA1

0y1: DMA2

i. Dwidth[2:0]

Transfer destination bit width

0y000: Byte (8 bits)

0y001: Half-word (16 bits)

0y010: Word (32 bits)

other: Reserved

j. <Swidth[2:0]>

The transfer source bit width must be an integral multiple of the transfer destination bit width.

k. <DBSize[2:0]>

Note: The burst size set in DBsize is unrelated to HBURST of the AHB bus.

l. <SBSize[2:0]>

Note: The burst size set in SBsize is unrelated to HBURST of the AHB bus.

m. <TransferSize>

Specifies the total number of transfers when the DMAC is operating as a flow controller.

The <TransferSize> value decrements with respect to each DMA transfer until it reaches 0.

On read, the number of transfers yet to be performed is read.

The total transfer count should be specified in units of the transfer source bit width.

Examples:

<Swidth>	Transfer count unit
8 bits	byte
16 bits	half-word
32 bits	word

Note: If the transfer source bit length is smaller than the transfer destination bit length, caution is required in specifying the total transfer count. Make sure that the following equation is satisfied.

Transfer source bit length × Total transfer count = Transfer destination bit length × N

N: Integer

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

  The DMACCxControl registers have the same structure as DMACC0Control.
  Please refer to the description of DMACC0Control.
  For the names and addresses of these registers, please refer to Table 3.8.3.

16. DMACC0Configuration (DMAC Channel0 Configuration Register)

Address = (0xF410_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 as undefined. Write as zero.</td></tr>
<tr><td>[18]</td><td>Halt</td><td>R/W</td><td>0y0</td><td>0y0: DMA requests accepted
0y1: DMA requests ignored</td></tr>
<tr><td>[17]</td><td>Active</td><td>RO</td><td>0y0</td><td>Read: 0y0: No data in the FIFO
0y1: The FIFO has data
Write: Invalid</td></tr>
<tr><td>[16]</td><td>Lock</td><td>R/W</td><td>0y0</td><td>0y0: Disable lock transfers
0y1: Enable lock transfers</td></tr>
<tr><td>[15]</td><td>ITC</td><td>R/W</td><td>0y0</td><td>Terminal count interrupt enable register
0y0: Disable interrupts
0y1: Enable interrupts</td></tr>
<tr><td>[14]</td><td>IE</td><td>R/W</td><td>0y0</td><td>Error interrupt enable register
0y0: Disable interrupts
0y1: Enable interrupts</td></tr>
<tr><td>[13:11]</td><td>FlowCntrl</td><td>R/W</td><td>0y000</td><td><table><tr><th>FlowCntrl set value</th><th>Transfer Mode</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 as 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 (Note1)
0y000-0y1111</td></tr>
<tr><td>[5]</td><td>–</td><td>–</td><td>Undefined</td><td>Read as 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 (Note1)
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: Please refer to Table 3.8.2 DMA request number chart.

[Description]

a. < ITC >

It is an enable register of transfer end interrupt.

Transfer end interrupt is generated by setting <ITC>=1 and DMACCxControl Register<I>=1.

b. < FlowCntrl>

This bit sets the transfer mode.

0y000: Memory to Memory

0y001: Memory to Peripheral

0y010: Peripheral to Memory

0y011: Peripheral to Peripheral

0y100 to 0y111: Reserved

Note: When you selected Memory-to-Memory, hardware start triggered by DMA is not supported. Transfer is started by writing <E>= 1.

c. <DestPeripheral>

This is a DMA request peripheral number in binary.

This setting will be ignored if memory is specified as the transfer destination.

d. <SrcPeripheral>

This is a DMA request peripheral number in binary.

This setting will be ignored if memory is specified as the transfer source.

e. <E>

This bit is used to enable or disable the channel. If the channel is disabled during a transfer, the data in the channel's FIFO will be lost. To re-start, the channel must be reset.

To temporarily stop DMA transfer, use the <Halt> bit to disable DMA requests, poll the <Active> bit until it becomes 0, and then clear the <E> bit to disable the channel.

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

The structure and description of these registers are same as DMACC0Configuration.

Please refer to the description of DMACC0Configuration.

The names and addresses of these registers, please refer to Table 3.8.3.

- DMAC configuration flow

  Ex: using DMAC ch1, transfer from Memory to built-in FIFO of I²S

  Total transfer data size: 32 words

  Transfer count unit: Swidth = Word

  Total transfer count: 32 counts

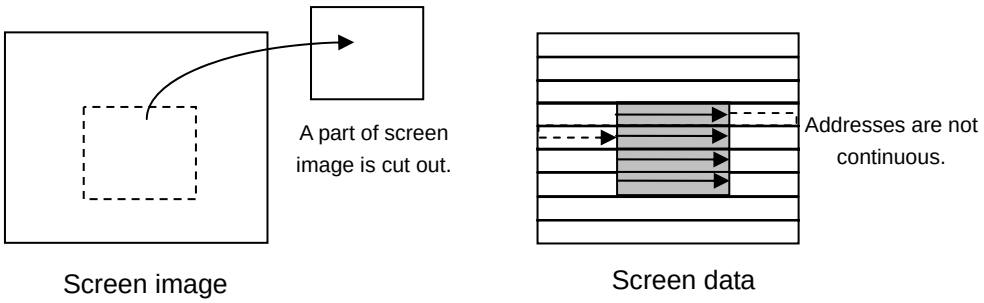
```
DMACConfiguration      ←  0x00000001      ; Set DMAC Active
DMACC1SrcAddr          ←  Memory address  ; Source address (DMAC ch1)
DMACC1DestAddr         ←  I2STDAT         ; Destination address
DMACC1Control          ←  0x04492020      ; Destination address fixed
                                          ; Source address increment
                                          ; Swidth = word, Dswidth = word
                                          ; DBSize= 8 bursts, SBSize= 8 bursts
                                          (Note)
                                          ; TransferSize = 32 counts
DMACC1Configuration    ←  0x00000a81      ; channel1 Enable,
                                          ; Memory to Peripheral (I2S1)
        .                     .
        .                     .           ; I2S configuration and Preparation
        .                     .
I2STDMA1               ←  0x00000001      ; I2S DMA Ready and request DMA
                                          transfer
```

Note: Please set Burst size equivalent to the FIFO size of Peripheral.

3.8.4 Special Function

1) Scatter/gather function

When a part of image data is cut off and transferred, the image data is not be handled as consecutive data. The addresses of the image data to be transferred are scattered according to specific rule. Since DMA can only transfer data to consecutive addresses, the transfer settings must be reconfigured each time a gap occurs in the sequence of transfer addresses.



The scatter/gather function enables a continuous DMA operation without involving the CPU by allowing the transfer settings (source address, destination address, transfer count, transfer bus width) to be re-loaded each time a specified number of DMA transfers have been completed. This is done by using the linked lists (LLI)..

The scatter/gather function is controlled by setting the DMACCxLLI register to 1.

A linked list includes information comprised of the following four words:

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

It is also possible to generate interrupts in conjunction with the scatter/gather function.

Terminal count interrupt is generated by setting both DMACCxControl Register<I>=1 and DMACCxConfiguration Register<ITC>=1. In case that terminal count interrupt is generated only when the final DMA transfer using scatter/gather function is performed, set DMACCxControl Register<I>=0 and DMACCxConfiguration Register<ITC>=1 to start transfer, and set <I>=1 in the final DMA transfer configuration flow to generate e terminal count interrupt only during the final DMA transfer.

If enabled, additional operations (e.g. adding conditions, branch etc.) during transfer using linked lists can be executed. An interrupt can be cleared by configuring the corresponding bit of the DMACIntTCClear register.

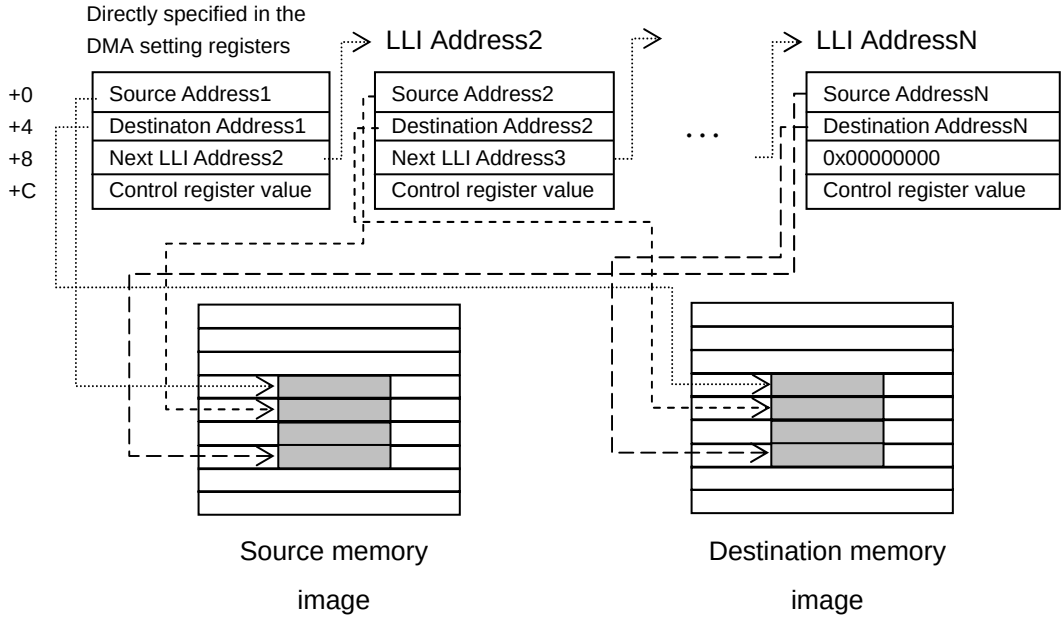
2) Linked list operation

To use the scatter/gather function, a series of linked lists should be created to define source and destination data areas.

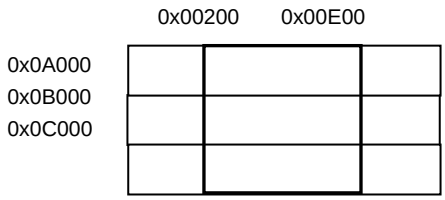
LLI enables to transfer unordered multiple blocks sequentially. Each LLI transfers data based on the configuration of normal DMA continuous transfer. Upon completion of each DMA transfer, the next LLI is loaded to continuously perform DMA operation (daisy-chained operation).

The following shows a setting example:

1. Set the information for the first DMA transfer to the DMA registers.
2. Write the information for the second and subsequent transfers to the memory space of the address specified by "next LLI AddressX".
3. To finish the linked list operation with the Nth DMA transfer, set "next LLI AddressX" to 0x00000000.

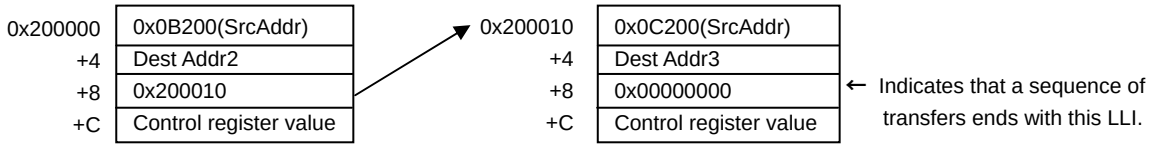


Example: When transferring data in the area enclosed by the square



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

Linked List



3.9 Port Functions

The list of the port pin functions and input-output port programming show how to configure each pin.

Information on power sources is also provided as different power sources are used for individual external pins.

Table 3.9.1 TMPA901CM pin assignment (dedicated pins)

Power Supply	Destination	Alias	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	No. of pads	No. of external INT pins	No. of internal INT vectors	I/O port	Open-drain port
DVCCM	Memory	SA	D[7:0]								8				
		SB	D[15:8]								8				
		–													
		–													
		SE	A[7:0]								8				
		SF	A[15:8]								8				
		SG	A[23:16]								8				
		SH	DMCCSn	–	–	SMCCS1n	SMCCS0n	SMCBE0n	–	–	4				
		SJ	–	DMCCKE	DMCBA1	DMCBA0	DMCCASn	DMCRASn	DMCWEn	SMCOEn	7				
		SK	–	–	SMCBE1n	SMCWEn	–	–	DMCSDQM1 DMCDDM1	DMCSDQM0 DMCDDM0	4				
		SL	–	DMCCLKIN	DMCDDQS1	DMCDDQS0	–	DMCAP	DMCDCLKN	DMCDCLKP DMCSCLK	6				
DVCC3IO DVCC1C,1B	Clock, Mode	SM	AM1	AM0	–	RESETn	XT2	XT1	X2	X1	7				
DVCC3IO, AVCC3H	USB Host Mode	SN	HDM	HDP	–	–	–	SELJTAG	SELDVCCM	SELMEMC	5				
DVCC3IO	JTAG	SP	–	–	TDO	RTCK	TRSTn	TDI	TMS	TCK	6				
AVDD3T/3C	USB2	SR	–	–	–	VSENS	REXT	–	DDM	DDP	4				

Note 1: Dedicated pins (with no port function).
Note 2: The alias "Sx" in the table above is only a symbol and does not have any general-purpose port function.

Table 3.9.2 TMPA901CM pin assignment (dual-purpose pins)

Power Supply	Destination	Alias	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	No. of pads	No. of external INT pins	No. of internal INT vectors	I/O port	Open-drain port
DVC3IO	Key I2C0 INT Other	PA	–	–	–	–	KI3	KI2	KI1	KI0	4	4	1	4(I)	
		PB	–	–	–	–	KO3 LCLLP	KO2 LCLFP	KO1 LCLAC	KO0 LCLCP	4			4(O)	4
		PC	I2C0DA INT9	I2C0CL	–	FSOUT PWM2OUT	MLDALM PWM0OUT	PWE	–	–	5	1	1	3(O) 2(I/O)	2
AVCC3AD	ADC TSI,INT	PD	AN7 / PY INTB	AN6 / PX INTA(INTTSI)	AN5 MY	AN4 MX	–	–	–	–	4	2	1	4(I)	
DVCC3IO	UART0	PN	–	–	–	–	–	–	U0RXD SIR0IN	U0TXD SIR0OUT	2	0	0	2(I/O)	
DVCC3IO	SSP0, USB H/D UART1	PT	X1USB	U1CTSn I2S1DATO	U1RXD USBOCn	U1TXD USBPON	SP0DI I2S0MCLK	SP0DO I2S0DATI	SP0CLK I2S0CLK	SP0FSS I2S0WS	8			8(I/O)	
DVCC3IO	NAND LCDC	PU	NDD7 LD7	NDD6 LD6	NDD5 LD5	NDD4 LD4	NDD3 LD3	NDD2 LD2	NDD1 LD1	NDD0 LD0	8			8(I/O)	
DVCC3IO	NAND LCDC	PV	LD15	NDRB LD14	NDCE1n LD13	NDCE0n LD12	NDCLE LD11	NDALE LD10	NDWEn LD9	NDREn LD8	8			8(I/O)	

Note 1: Dual-purpose pins (they have the port function.)

Note 2: The alias "Px" in the table above indicates the general-purpose port function.

Table 3.9.3 TMPA901CM address and initial value table

Register Name	R/W	Address	Description	Meaning		PortA	PortB	PortC	PortD	PortN	PortT	PortU	PortV
				0	1								
GPIOnDATA	R/W	0x000 -0x3FC	Data register	–	–	0xF	0x0	0xEF	0xFF	0xFF	0XFF	0xFF	0XFF
GPIOnDIR	R/W	0x400	Data direction register	Input port	Output port	Note	Note	0x1F	Note	0x00	0x00	0x00	0x00
GPIOnFR1	R/W	0x424	Function register 1	GPIO	Function 1 input or Output enable	Note	0x0	0x00	0xFF	0x00	0x00	0x00	0x00
GPIOnFR2	R/W	0x428	Function register 2	GPIO	Function 2 input or Output enable	Note	0x0	0x00	0x00	0x00	0x00	0x00	0x00
GPIOnIS	R/W	0x804	Interrupt sensitivity register	Edge	Level	0x0		0x00	0x00				
GPIOnIBE	R/W	0x808	Interrupt-both-edge register	Single edge	Both-edge	0x0		0x00	0x00				
GPIOnIEV	R/W	0x80C	Interrupt event register	Falling edge or Low level	Rising edge or High level	0x0		0x00	0x00				
GPIOnIE	R/W	0x810	Interrupt enable register	Disable	Enable	0x0		0x00	0x00				
GPIOnRIS	RO	0x814	Raw interrupt status register	No interrupt requested	Interrupt requested	0x0		0x00	0x00				
GPIOnMIS	RO	0x818	Masked interrupt status register	No interrupt requested	Interrupt requested	0x0		0x00	0x00				
GPIOnIC	WO	0x81C	Interrupt clear register	–	Clear	0x0		0x00	0x00				
GPIOnODE	R/W	0xC00	Open-drain output enable register	3-state output	Open-drain output		0x0	0x00					

Writes are prohibited depending on the bits.

No register exists.

Note: Reserved: Don't access this register.

3.9.1 Data Registers

[Notes on data registers]

All data registers allow all the 8 bits to be read or written simultaneously. It is also possible to mask certain bits in reading from or writing to the data registers.

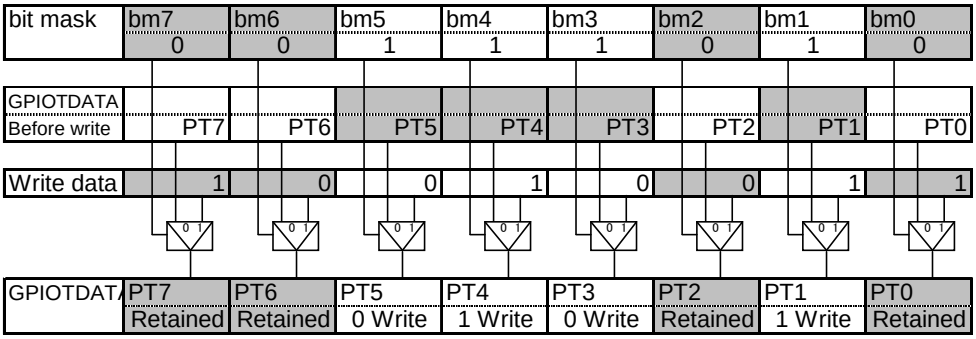
Data registers allow accesses to a 256-address space (0x0000 to 0x03FC). (Assume that addresses are shifted to the high-order side by 2 bits. The lower 2 bits have no meaning. Valid addresses exist at every 4 addresses, such as 0x000, 0x0004, and so on.)

Accesses to the 256-address space are done through the same data register. Valid bits vary according to the address to be accessed.

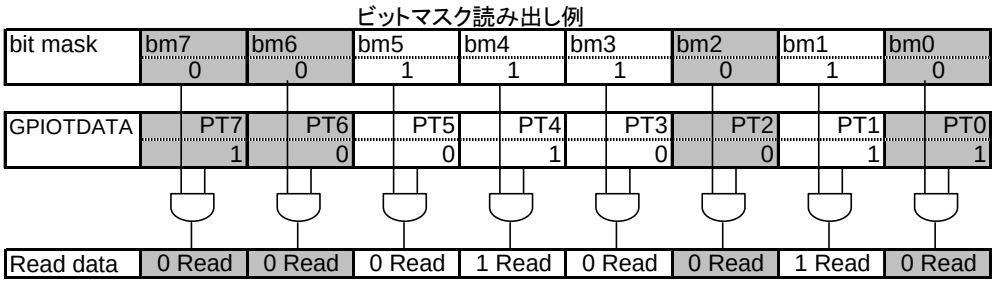
Bits [9:2] of the address to be accessed correspond to bits [7:0] of the data register. Address bits that are 1 are accessed in the data register and address bits that are 0 are masked.

Address[9:2]	Bit9	Bit8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2
Bit mask	bm7	bm6	bm5	bm4	bm3	bm2	bm1	bm0

- Example: Writing 0x93 to address 0x00E8 of Port T by using bit masks



- Example: Reading 0x12 from address 0x00E8 of Port T by using bit masks



Note: All the bits are valid in accessing 0x03FC, and no bits are valid in accessing 0x0000.

3.9.2 Port Function Settings

This section describes the settings of Port A through Port V that can also function as general-purpose ports. Each port should basically be accessed in word (32-bit) units.

3.9.2.1 Port A

Port A can be used not only as a general-purpose input pin with pull up but also as key input pin.

By enabling interrupts, Port A is used as key input pins (KI3-KI0).

Port A can be used without pull up. Please refer to Section 3.26 PMC.

General-purpose input setting

Function	Data Value	Interrupt Enable
General-purpose input	GPIOADATA	GPIOAIE
	*	0

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

Note: All bits are provided with pull up resisters.

Key input function setting

Function	Interrupt Enable
Key input	GPIOAIE
	1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				KI3	KI2	KI1	KI0

Note: All bits support the interrupt function. All bits are provided with pull up resisters.

Base address = 0xF080_0000

Register Name	Address (base+)	Description
GPIOADATA	0x03FC	PortA Data Regisiter
–	0x0400	Reserved
–	0x0424	Reserved
–	0x0428	Reserved
GPIOAIS	0x0804	Port A Interrupt Select Register (Level and Edge)
GPIOAIBE	0x0808	Port A Interrupt Select Register (Single edge and Both edge)
GPIOAIEV	0x080C	Port A Interrupt Select Register (Falling edge/Low level and Rising edge/High level)
GPIOAIE	0x0810	Port A Interrupt Enable Register
GPIOARIS	0x0814	Port A Interrupt Status Register (Raw)
GPIOAMIS	0x0818	Port A Interrupt Status Register (Masked)
GPIOAIC	0x081C	Port A Interrupt Clear Register
–	0x0C00	Reserved

1. GPIOADATA (Port A Data Register)

Address = (0xF080_0000) + (0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit Mask	Description
[31:4]	–	–	Undefined	–	Read as undefined. Write as zero.
[3:0]	PA[3:0]	RO	0xF	Bm3:0	Port A data register

[Description]

a. <PA[3:0]>

Data register: Stores data.

See notes on data registers for the bit mask function.

2. GPIOAIS (Port A Interrupt Select Register (Level and Edge))

Address = (0xF080_0000) + (0x0804)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Written as zero.
[3:0]	PA3IS to PA0IS	R/W	0x0	Port A interrupt sensitivity register (for each bit) 0y0: Edge-sensitive 0y1: Level-sensitive

[Description]

a. <PA3IS to PA0IS>

Interrupt sensitivity register: Selects edge-sensitive or level-sensitive.

0y0: Edge-sensitive

0y1: Level-sensitive

3. GPIOAIBE (Port A Interrupt Select Register (Single edge and Both edge))

Address = (0xF080_0000) + (0x0808)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Written as zero.
[3:0]	PA3IBE to A0IBE	R/W	0x0	Port A interrupt both-edge register (for each bit) 0y0: Single edge 0y1: Both-edge

[Description]

a. <PA3IBE to PA0IBE>

Interrupt both-edge register: Selects single edge or both-edge.

0y0: Single edge

0y1: Both-edge

4. GPIOAIEV (Port A Interrupt Select Register (“Falling edge/Low level” and “Rising edge/High level”))

Address = (0xF080_0000) + (0x080C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Written as zero.
[3:0]	PA3IEV to PA0IEV	R/W	0x0	Port A interrupt event register (for each bit) 0y0: Falling edge/Low level 0y1: Rising edge/High level

[Description]

a. <PA3IEV to PAIEV>

Interrupt event register: Selects falling edge or rising edge for edge-sensitive interrupts, and Low level or High level for level-sensitive interrupts.

0y0: Falling edge/Low level

0y1: Rising edge/High level

5. GPIOAIE (Port A Interrupt Enable Register)

Address = (0xF080_0000) + (0x0810)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Written as zero.
[3:0]	PA3IE to PA0IE	R/W	0x0	Port A interrupt enable register (for each bit) 0y0: Disable 0y1: Enable

[Description]

a. <PA3IE to PA0IE>

Interrupt enable register: Enables or disables interrupts.

0y0: Disable

0y1: Enable

6. GPIOARIS (Port A Interrupt Status Register (Raw))

Address = (0xF080_0000) + (0x0814)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined.
[3:0]	PA3RIS to PA0RIS	RO	0x0	Port A interrupt raw status register (for each bit) 0y0: Not requested 0y1: Requested

[Description]

a. <PA3RIS to PA0RIS>

Interrupt raw status register: Monitors the interrupt status before being masked by the interrupt enable register.

0y0: Not requested

0y1: Requested

7. GPIOAMIS (Port A Interrupt Status Register (Masked))

Address = (0xF080_0000) + (0x0818)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined.
[3:0]	PA3MIS to PA0MIS	RO	0x0	Port A masked interrupt status register (for each bit) 0y0: Not requested 0y1: Requested

[Description]

a. <PA3MIS to PA0MIS>

Masked interrupt status register: Monitors the interrupt status after masking.

0y0: Not requested

0y1: Requested

8. GPIOAIC (Port A Interrupt Clear Register)

Address = (0xF080_0000) + (0x081C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Written as zero.
[3:0]	PA3IC to PA0IC	WO	0x0	Port A interrupt clear register (for each bit) 0y0: Invalid 0y1: Clear

[Description]

a. <PA3IC to PA0IC>

Interrupt clear register: Clears edge-sensitive interrupts.

0y0: Invalid

0y1: Clear

3.9.2.2 Port B

Port B can be used not only as general-purpose output pins but also as key output pins. By enabling open-drain output, Port B is used as key output (KO3-KO0). And this port has a LCDC control signal.

General-Purpose Output Setting

Function	Data Value	Open-Drain Enable	Function Select 2
General-purpose output	GPIOBDATA	GPIOBODE	GPIOBFR2
	*	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				Output	Output	Output	Output

Key Output Setting

Function	Data Value	Open-Drain Enable
Key output	GPIOBDATA	GPIOBODE
	*	1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				KO3	KO2	KO1	KO0

Note: 3 to 0 bits support open-drain mode.

LCD control Output Setting

Function	Data Value	Open-Drain Enable	Function Select 2
Key output	GPIOBDATA	GPIOBODE	GPIOBFR2
	*	0	1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				LCLLP	LCLFP	LCLAC	LCLCP

Base address = 0xF080_1000

Register Name	Address (base+)	Description
GPIOBDATA	0x03FC	PortB Data Regisiter
–	0x0400	Reserved
–	0x0424	Reserved
GPIOBFR2	0x0428	PortB Function Register2
–	0x0804	Reserved
–	0x0808	Reserved
–	0x080C	Reserved
–	0x0810	Reserved
–	0x0814	Reserved
–	0x0818	Reserved
–	0x081C	Reserved
GPIOBODE	0x0C00	Port B Open-drain Output Enable Register

1. GPIOBDATA (Port B Data Register)

Address = (0xF080_1000) + (0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:4]	–	–	Undefined	–	Read as undefined. Written as zero.
[3:0]	PB[3:0]	R/W	0x0	Bm3:0	Port B data register

[Description]

a. <PB[3:0]>

Data register: Stores data.

See notes on data registers for bit masking.

2. GPIOBFR2 (Port B Function Register2)

Address = (0xF080_1000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	PB3F2 to PB0F2	R/W	0x0	Port B function register 2

(個別説明)

a. <PB3F2 to PB0F2>

Function register 2: Controls the function setting.

3. GPIOBODE (Port B Open-drain Output Enable Register)

Address = (0xF080_1000) + (0x0C00)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Written as zero.
[3:0]	PB3ODE to PB0ODE	R/W	0x0	Port B open-drain output enable register (for each bit) 0y0: Push-Pull output 0y1: Open-drain (Pch disabled) output

[Description]

a. <PB3ODE to PB0ODE>

Open-drain output enable register: Selects Push-Pull output or open-drain output.

0y0: Push-Pull output

0y1: Open-drain (Pch disabled) output

3.9.2.3 Port C

The upper 2 bits (bits [7:6]) of Port C can be used as general-purpose input/output pins and the lower 3 bits (bits [4:2]) can be used as general-purpose output pins.

Port C can also be used as interrupt (INT9), I²C (I2C0DA, I2C0CL), low-frequency clock output (FSOUT), melody output (MLDALM), PWM output function (PWM0OUT, PWM2OUT). And with regard to PWE pin, please refer to NOTE described later for details.

General-purpose input and Interrupt settings

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2	Interrupt Enable	Open-Drain Enable
General-purpose input	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2	GPIOCIE	GPIOCODE
Interrupt	*	0	0	0	0/1	*

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Input/INT9	Input						

Note: Only bit 7 support the interrupt function.

General-purpose output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2	Interrupt Enable	Open-Drain Enable
General-purpose output	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2	GPIOCIE	GPIOCODE
	*	1	0	0	0	0/1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Output	Output		Output	Output	Output		

Note: Bits 7 to 6 support open-drain mode.

PWE setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2	Interrupt Enable	Open-Drain Enable	PMC register
PWE	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2	GPIOCIE	GPIOCODE	PMCCTL <PMCPWE>
	*	*	*	*	*	*	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
					PWE		

I^2C setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2	Interrupt Enable	Open-Drain Enable
I^2C	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2	GPIOCIE	GPIOCODE
	*	*	1	0	0	1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2C0DA	I2C0CL						

MLDALM, FSOUT output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2	Interrupt Enable	Open-Drain Enable
MLDALM	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2	GPIOCIE	GPIOCODE
	*	*	1	0	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
*	*		FSOUT	MLDALM			

PWM output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2	Interrupt Enable	Open-Drain Enable
PWM output	GPIOCDATA	GPIOCDIR	GPIOCFR1	GPIOCFR2	GPIOCIE	GPIOCODE
	*	*	0	1	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-		PWM2OUT	PWM0OUT			

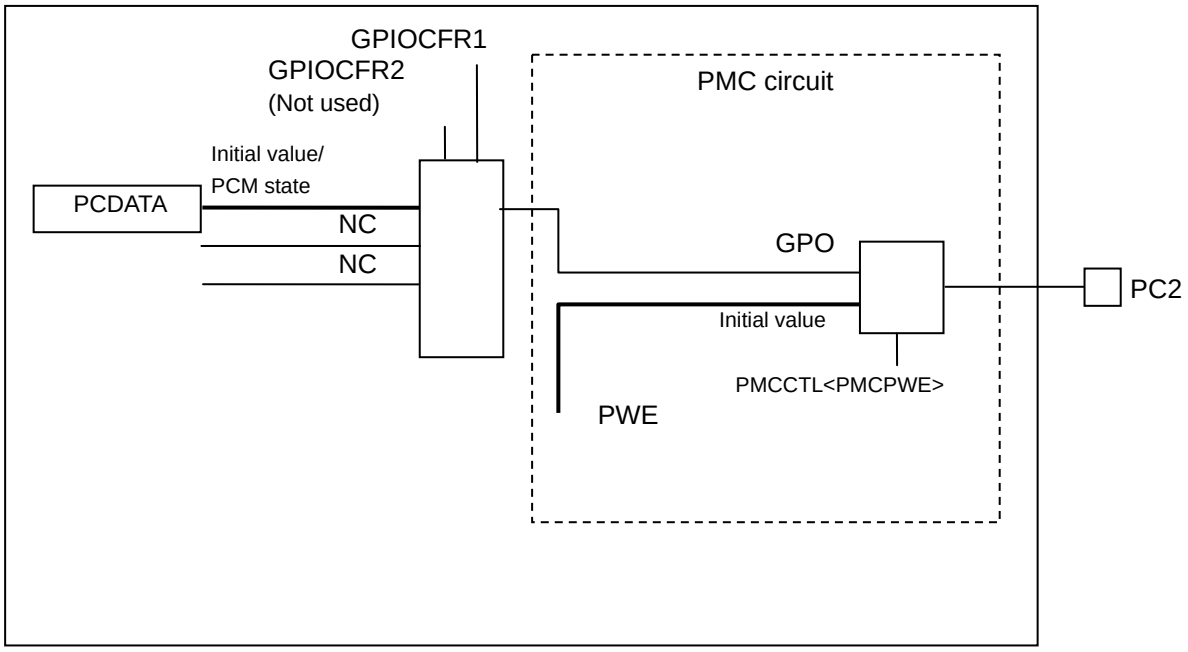
Note: about PC2 setting

This MCU implements power management circuit that can cut off power supply to circuit blocks other than some special circuits and I/O pins. For details, please refer to PMC chapter).

Even if the power of some internal circuits is cut off, the statuses of external IO can be held.

Care should be taken when controlling ports. Furthermore, please pay special attention to the PC2 port control due to its particular circuit configuration. The below chart shows an internal circuit connection diagram.

TMPA901CMXBG



- In PCM (Power Cut mode) mode, the general Port Function of PC2 can't be used.
- To use PC2 as a general port, please set GPIOCFR1<PC2F1>= 0y0, GPIOCFR2<PC2F2> = 0y0 and PMCCTL<PMCPWE> = 0y0, PMCWV1<PMCCTLV>=0y1 in the PMC function and then the PMCCTL<PMCPWE> will be valid as "0" after to be read the bit of PMCWV1<PMCCTLV> to "1".

```
Set:      GPIOCFR1<PC2F1>=0y0
Set:      GPIOCFR2<PC2F2>=0y0
Set:      PMCCTL<PMCPWE>=0y0
Set:      PMCWV1<PMCCTLV>=0y1
                  |
                  | 2.5 XT1 cycles, approximately 77μs
                  |
Read:     PMCWV1<PMCCTLV>=0y1
Valid:    PMCCTL<PMCPWE>=0y0
```

Note: when no use the low frequency oscillator, the PC2 can not use as a general port.

Base address = 0xF080_2000

Register Name	Address (base+)	Description
GPIOCDATA	0x03FC	Port C Data Register
GPIOCDIR	0x0400	Port C Data Direction Register
GPIOCFR1	0x0424	Port C Function Register 1
GPIOCFR2	0x0428	Port C Function Register 2
GPIOCIS	0x0804	Port C Interrupt Select Register (Level and Edge)
GPIOCIBE	0x0808	Port C Interrupt Select Register (Single edge and Both edge)
GPIOCIEV	0x080C	Port C Interrupt Select Register (Falling edge/Low level and Rising edge/High level)
GPIOCIE	0x0810	Port C Interrupt Enable Register
GPIOCRIS	0x0814	Port C Interrupt Status Register (Raw)
GPIOCMIS	0x0818	Port C Interrupt Status Register (Masked)
GPIOCIC	0x081C	Port C Interrupt Clear Register
GPIOCODE	0x0C00	Port C Open-drain Output Enable Register

1. GPIOCDATA (Port C Data Register)

Address = (0xF080_2000) + (0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read as undefined. Write as zero.
[7:6]	PC[7:6]	R/W	0y11	Bm7:6	Port C data register
[5]	–	–	Undefined	–	Read as undefined. Write as zero.
[4:2]	PC[4:2]	R/W	0y011	Bm4:2	Port C data register
[1:0]	–	–	Undefined	–	Read as undefined. Write as zero.

[Description]

a. <PC7,PC6,PC4,PC3,PC2>

Data register: Stores data.

See notes on data registers for bit masking.

2. GPIOCDIR (Port C Data Direction Register)

Address = (0xF080_2000) + (0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:6]	PC7C to PC6C	R/W	0y00	Port C data direction register (for each bit) 0y0: Input 0y1: Output
[5]	–	–	Undefined	Read as undefined. Write as zero.
[4:2]	PC4C to PC0C	–	0y111	Must be written as 1. Read as 1.
[1:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <PC7C to PC6C>

Data direction register: Selects input or output when Port C is used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOCFR1 (Port C Function Register 1)

Address = (0xF080_2000) + (0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:6]	PC7F1 to PC6F1	R/W	0y00	Port C function register 1
[5]	Reserved	R/W	0y0	Must be written as 0. Read as 0.
[4:2]	PC4F1 to PC2F1	R/W	0y000	Port C function register 1
[1:0]	Reserved	R/W	0y00	Must be written as 0. Read as 0.

[Description]

a. <PC7F1 to PC6F1, PC4F1 to PC2F1>

Function register 1: Controls the function setting.

4. GPIOCFR2 (Port C Function Register 2)

Address = (0xF080_2000) + (0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:5]	Reserved	R/W	0y000	Must be written as 0. Read as 0.
[4:3]	PC4F2 to PC3F2	R/W	0y00	Port C function register 2
[2:0]	Reserved	R/W	0y000	Must be written as 0. Read as 0.

[Description]

a. < PC4F2 to PC3F2 >

Function register 2: Controls the function setting.

Note: 1 can be set to only one of the function register 1 or the function register 2 at a time. These registers must not be written as 1 simultaneously even for an instant.

Table 3.9.4 Function register setting table

Mode	GPIOCFR1	GPIOCFR2
General-purpose	0	0
Function 1	1	0
Function 2	0	1
Prohibited	1	1

5. GPIOCIS (Port C Interrupt Select Register (Level and Edge))

Address = (0xF080_2000) + (0x0804)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7]	PC7IS	R/W	0y0	Port C interrupt sensitivity register 0y0: Edge-sensitive 0y1: Level-sensitive
[6:0]	–	–	Undefined	Read as undefined. Written as zero.

[Description]

a. <PC7IS>

Interrupt sensitivity register: Selects edge-sensitive or level-sensitive.

0y0: Edge-sensitive

0y1: Level-sensitive

6. GPIOCIBE (Port C Interrupt Select Register (Single edge and Both edge))

Address = (0xF080_2000) + (0x0808)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7]	PC7IBE	R/W	0y0	Port C interrupt both-edge register 0y0: Single edge 0y1: Both-edge
[6:0]	–	–	Undefined	Read as undefined. Written as zero.

[Description]

a. <PC7IBE>

Interrupt both-edge register: Selects the trigger mode from single edge and both-edge.

0y0: Single edge

0y1: Both-edge

7. GPIOCIEV (Port C Interrupt Select Register ("Falling edge/Low level" and "Rising edge/High level"))

Address = (0xF080_2000) + (0x080C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7]	PC7IEV	R/W	0y0	Port C interrupt event register 0y0: Falling edge/Low level 0y1: Rising edge/High level
[6:0]	–	–	Undefined	Read as undefined. Written as zero.

[Description]

a. <PC7IEV>

Interrupt event register: Select falling edge or rising edge for edge-sensitive interrupts, and Low level or High level for level-sensitive interrupts.

0y0: Falling edge/Low level

0y1: Rising edge/High level

8. GPIOCIE (Port C Interrupt Enable Register)

Address = (0xF080_2000) + (0x0810)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7]	PC7IE	R/W	0y0	Port C interrupt enable register 0y0: Disabled 0y1: Enabled
[6:0]	Reserved	R/W	0y0000000	Must be written as 0. Read as 0.

[Description]

a. <PC7IE>

Interrupt enable register: Enables or disables interrupts.

0y0: Disabled

0y1: Enabled

9. GPIOCRIS (Port C Interrupt Status Register (Raw))

Address = (0xF080_2000) + (0x0814)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7]	PC7RIS	RO	0y0	Port C interrupt raw status register 0y0: Not requested 0y1: Requested
[6:0]	–	–	Undefined	Read as undefined.

[Description]

a. <PC7RIS>

Interrupt raw status register: Monitors the interrupt status before being masked by the interrupt enable register.

0y0: Not requested

0y1: Requested

10. GPIOCMIS (Port C Interrupt Status Register (Masked))

Address = (0xF080_2000) + (0x0818)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7]	PC7MIS	RO	0y0	Port C masked interrupt status register 0y0: Not requested 0y1: Requested
[6:0]	–	–	Undefined	Read as undefined.

[Description]

a. <PC7MIS>

Masked interrupt status register: Monitors the interrupt status after being masked by the interrupt enable.

0y0: Not requested

0y1: Requested

Following table is an example configurations of interrupt register.
The configurations of each register and bit are shown below.

Table 3.9.5 An example configurations of interrupt register

(GPOxIS, GPIOxIBE, GPIOxIEV, GPIOxIE, GPIOxRIS, GPIOxMIS: x = A, C, D)

Register setting				Trigger Mode	Output		
GPIOxIS (Port x Interrupt Select Register (Level and Edge))	GPIOxIBE (Port x Interrupt Select Register (Single edge and Both edge)	GPIOxIEV (Port x Interrupt Select Register (Falling edge/Low level and Rising edge/High level))	GPIOxIE (Port x Interrupt Enable Register)		GPIOxRIS (Port x Interrupt Status Register (Raw))	GPIOxMIS (Port x Interrupt Status Register (Masked))	INTS [Num]
0	0	0	0	Falling edge detection	Detection enabled	Detection disabled (0x00)	Detection disabled
		1		Rising edge detection			
	1	0		Both edge detection			
		1					
	0	0	1	Falling edge detection	Detection enabled	Detection enabled	Detection enabled
		1		Rising edge detection			
	1	0		Both edge detection			
		1					
1	0	0	0	Low level detection	Detection enabled	Detection disabled (0x00)	Detection disabled
		1		High level detection			
	1	0		Low level detection			
		1		High level detection			
	0	0	1	Low level detection	Detection enabled	Detection enabled	Detection enabled
		1		High level detection			
	1	0		Low level detection			
		1		High level detection			

11. GPIOCIC (Port C Interrupt Clear Register)

Address = (0xF080_2000) + (0x081C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7]	PC7IC	WO	0y0	Port C interrupt clear register 0y0: Invalid 0y1: Request cleared
[6:0]	–	–	Undefined	Read as undefined. Written as zero.

[Description]

a. <PC7IC>

Interrupt clear register: Clears edge-sensitive interrupts.

0y0: Invalid

0y1: Request cleared

12. GPIOCODE (Port C Open-drain Output Enable Register)

Address = (0xF080_2000) + (0x0C00)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:6]	PC7ODE to PC6ODE	R/W	0x00	Port C open-drain output enable register (for each bit) 0y0: Push-Pull output 0y1: Open-drain (Pch disabled) output
[5]	–	–	Undefined	Read as undefined. Write as zero.
[4:2]	Reserved	R/W	0y000	Must be written as 0. Read as 0.
[1:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <PC7ODE, PC6ODE>

Open-drain output enable register: Selects the output mode from Push-Pull output and Open-drain output.

0y0: Push-Pull output

0y1: Open-drain (Pch disabled) output

3.9.2.4 Port D

Port D can be used as general-purpose input.

Port D can also be used as interrupt (INTB, INTA), ADC (AN7-AN4), and touch screen control (PX, PY, MX, MY) pins.

General-purpose input and Interrupt settings

Function	Data Value	Function Select 1	Function Select 2	Interrupt Enable
General-purpose input	GPIODDATA	GPIODFR1	GPIODFR2	GPIODIE
Interrupt	*	0	0	0/1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Input/INTB	Input/INTA	Input	Input	Input	Input	Input	Input

Note: Only bits 7 and 6 support the interrupt function.

ADC settings

Function	Data Value	Function Select 1	Function Select 2	Interrupt Enable
ADC	GPIODDATA	GPIODFR1	GPIODFR2	GPIODIE
	*	1	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AN7	AN6	AN5	AN4	-	-	-	-

TSI settings

Function	Data Value	Function Select 1	Function Select 2	Interrupt Enable
TSI	GPIODDATA	GPIODFR1	GPIODFR2	GPIODIE
	*	0	1	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PY	PX/INTA(INTTSI)	MY	MX	–	–	–	–

Base address = 0xF080_3000

Register Name	Address (base+)	Description
GPIODDATA	0x03FC	Port D Data Register
–	0x0400	Reserved
GPIODFR1	0x0424	Port D Function Register1
GPIODFR2	0x0428	Port D Function Register2
GPIODIS	0x0804	Port D Interrupt Select Register (Level and Edge)
GPIODIBE	0x0808	Port D Interrupt Select Register (Single edge and Both edge)
GPIODIEV	0x080C	Port D Interrupt Select Register (Falling edge/Low level and Rising edge/High level)
GPIODIE	0x0810	Port D Interrupt Enable Register
GPIODRIS	0x0814	Port D Interrupt Status Register (Raw)
GPIODMIS	0x0818	Port D Interrupt Status Register (Masked)
GPIODIC	0x081C	Port D Interrupt Clear Register
–	0x0C00	Reserved

1. GPIODDATA (Port D Data Register)

Address = (0xF080_3000) + (0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read as undefined.
[7:4]	PD[7:4]	RO	0xF	Bm7:4	Port D data register
[3:0]	–	–	Undefined	–	Read as undefined.

[Description]

a. <PD[7:4]>

Data register: Stores data.

See notes on data registers for bit masking.

2. GPIODFR1 (Port D Function Register 1)

Address = (0xF080_3000) + (0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:4]	PD7F1 to PD4F1	R/W	0xF	Port D function register 1
[3:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <PD7F1 to PD4F1>

Function register 1: Controls the function setting.

3. GPIODFR2 (Port D Function Register 2)

Address = (0xF080_3000) + (0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:4]	PD7F2 to PD4F2	R/W	0y0000	Port D function register 2
[3:0]	Reserved	R/W	0y0000	Must be written as 0. Read as 0.

[Description]

a. <PD7F2 to PD4F2>

Function register 2: Controls the function setting.

Note: 1 can be set to only one of the function register 1 or the function register 2 at a time. These registers must not be written as 1 simultaneously even for an instant.

Table 3.9.6 Function register setting table

Mode	GPIODFR1	GPIODFR2
General-purpose	0	0
Function 1	1	0
Function 2	0	1
Prohibited	1	1

4. GPIODIS (Port D Interrupt Select Register (Level and Edge))

Address = (0xF080_3000) + (0x0804)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:6]	PD7IS to PD6IS	R/W	0y00	Port D interrupt sensitivity register (for each bit) 0y0: Edge-sensitive 0y1: Level-sensitive
[5:0]	Reserved	R/W	0y000000	Must be written as 0. Read as 0.

- [Description]

a. <PD7IS to PD6IS>

Interrupt sensitivity register: Selects the interrupt trigger mode from edge-sensitive and level-sensitive.

0y0: Edge-sensitive

0y1: Level-sensitive

5. GPIODIBE (Port D Interrupt Select Register (Single edge and Both-edge))

Address = (0xF080_3000) + (0x0808)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:6]	PD7IBE to D6IBE	R/W	0y00	Port D interrupt both-edge register (for each bit) 0y0: Single edge 0y1: Both-edge
[5:0]	Reserved	R/W	0y000000	Must be written as 0. Read as0.

[Description]

a. <PD7IBE to PD6IBE>

Interrupt both-edge register: Selects the trigger edge from single edge or both-edge.

0y0: Single edge

0y1: Both-edge

6. GPIODIEV (Port D Interrupt Select Register (“Falling edge/Low level” and “Rising edge/High level”))

Address = (0xF080_3000) + (0x080C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:6]	PD7IEV to PD6IEV	R/W	0y00	Port D interrupt event register (for each bit) 0y0: Falling edge/Low level 0y1: Rising edge/High level
[5:0]	Reserved	R/W	0y000000	Must be written as 0. Read as 0.

[Description]

a. <PD7IEV to PD6IEV>

Interrupt event register: Selects falling edge or rising edge for edge-sensitive interrupts, and Low level or High level for level-sensitive interrupts.

0y0: Falling edge/Low level

0y1: Rising edge/High level

7. GPIODIE (Port D Interrupt Enable Register)

Address = (0xF080_3000) + (0x0810)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:6]	PD7IE to PD6IE	R/W	0y00	Port D interrupt enable register (for each bit) 0y0: Disable 0y1: Enable
[5:0]	Reserved	R/W	0y000000	Must be written as 0. Read as 0.

[Description]

a. <PD7IE to PD6IE>

Interrupt enable register: Enables or disables interrupts.

0y0: Disable

0y1: Enable

8. GPIODRIS (Port D Interrupt Status Register (Raw))

Address = (0xF080_3000) + (0x0814)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7:6]	PD7RIS to PD6RIS	RO	0y00	Port D interrupt raw status register (for each bit) 0y0: Not requested 0y1: Requested
[5:0]	–	–	Undefined	Read as undefined.

[Description]

a. <PD7RIS to PD6RIS>

Interrupt raw status register: Monitors the interrupt status before being masked by the interrupt enable register.

0y0: Not requested

0y1: Requested

9. GPIODMIS (Port D Interrupt Status Register (Masked))

Address = (0xF080_3000) + (0x0818)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7:6]	PD7MIS to PD6MIS	RO	0y00	Port D masked interrupt status register (for each bit) 0y0: Not requested 0y1: Requested
[5:0]	–	–	Undefined	Read as undefined.

[Description]

a. <PD7MIS to PD6MIS>

Masked interrupt status register: Monitors the interrupt status after being masked by the interrupt enable register.

0y0: Not requested

0y1: Requested

Note: Refer to Table 3.9.5 for the configurations of each external interrupt register.

10. GPIODIC (Port D Interrupt Clear Register)

Address = (0xF080_3000) + (0x081C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:6]	PD7IC to PD6IC	WO	0y00	Port D interrupt clear register (for each bit) 0y0: Invalid 0y1: Request cleared
[5:0]	–	–	Undefined	Read as undefined. Written as zero.

[Description]

a. <PD7IC to PD6IC>

Interrupt clear register: Clears edge-sensitive interrupts.

0y0: Invalid

0y1: Request cleared

3.9.2.5 Port N

Port N can be used as general-purpose input/output pins.

Port N can also be used as UART/IrDA function (U0RXD, U0TXD, SIR0IN, SIR0OUT) pins.

General-purpose input setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose input	GPIONDATA	GPIONDIR	GPIONFR1	GPIONFR2
	*	0	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	-	Input	Input

General-purpose output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose output	GPIONDATA	GPIONDIR	GPIONFR1	GPIONFR2
	*	1	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	-	Output	Output

UART (ch0) setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
UART(ch0)	GPIONDATA	GPIONDIR	GPIONFR1	GPIONFR2
	*	*	1	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	–	–	–	–	–	–	U0TXD

UART/IrDA (ch0) setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
UART (ch0/IrDA)	GPIONDATA	GPIONDIR	GPIONFR1	GPIONFR2
	*	*	0	1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
–	–	–	–	–	–	SIR0IN /U0RXD	SIR0OUT

Base address = 0xF080_C000

Register Name	Address (base+)	Description
GPIONDATA	0x03FC	Port N Data Register
GPIONDIR	0x0400	Port N Data Direction Register
GPIONFR1	0x0424	Port N Function Register1
GPIONFR2	0x0428	Port N Function Register2
Reserved	0x0804	
Reserved	0x0808	
Reserved	0x080C	
Reserved	0x0810	
Reserved	0x0814	
Reserved	0x0818	
Reserved	0x081C	
Reserved	0x0C00	

1. GPIONDATA (Port N Data Register)

Address = (0xF080_C000) + (0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:2]	–	–	Undefined	–	Read as undefined. Written as zero.
[1:0]	PN[1:0]	R/W	0y11	Bm1:0	Port N data register

[Description]

a. <PN[1:0]>

Data register: Stores data.

See notes on data registers for the bit mask function.

2. GPIONDIR (Port N Data Direction Register)

Address = (0xF080_C000) + (0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Written as zero.
[1:0]	PN1C to PN0C	R/W	0y00	Port N data direction register (for each bit) 0y0: Input 0y1: Output

[Description]

a. <PN1C to PN0C>

Data direction register: Selects input or output for each pin used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIONFR1 (Port N Function Register 1)

Address = (0xF080_C000) + (0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Written as zero.
[1]	Reserved	–	0y0	Must be written as 0. Read as 0.
[0]	PN0F1	R/W	0y0	Port N function register 1

[Description]

a. <PN0F1>

Function register 1: Controls the function setting.

4. GPIONFR2 (Port N Function Register 2)

Address = (0xF080_C000) + (0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:2]	Reserved	–	0y000000	Must be written as 0. Read as 0.
[1:0]	PN1F2 to PN0F2	R/W	0y00	Port N function register 2

[Description]

a. <PN1F2 to PN0F2>

Function register 2: Controls the function setting.

Note: The function register 1 and function register 2 can only be set exclusively of each other. These registers must not simultaneously be written as 1 even for an instant.

Table 3.9.7 Function register setting table

Mode	GPIONFR1	GPIONFR2
General-purpose	0	0
Function 1	1	0
Function 2	0	1
Prohibited	1	1

3.9.2.6 Port T

Port T can be used as general-purpose input/output pins.

Port T can also be used as USB external clock input (X1USB), UART function (U1CTSn, U1RXD, U1TXD), SPI function (SP0DI, SP0DO, SP0CLK, SP0FSS), I2S control function, USBOCn and USBONn pins.

General-purpose input setting

Function	Data Value	Input/Output Select	Function Select 1
General-purpose input	GPIOTDATA	GPIOTDIR	GPIOTFR1
	*	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Input	Input	Input	Input	Input	Input	Input	Input

General-purpose output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose output	GPIOTDATA	GPIOTDIR	GPIOTFR1	GPIOTFR2
	*	1	0	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Output	Output	Output	Output	Output	Output	Output	Output

UART, SPI settings

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
UART	GPIOTDATA	GPIOTDIR	GPIOTFR1	GPIOTFR2
SPI	*	*	1	0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X1USB	U1CTSn	U1RXD	U1TXD	SP0DI	SP0DO	SP0CLK	SP0FSS

USB Host control, I2S setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
USB Host	GPIOTDATA	GPIOTDIR	GPIOTFR1	GPIOTFR2
I2S	*	*	0	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	I2S1DATO	USBOCn	USBPON	I2S0MCLK	I2S0DATI	I2S0CLK	I2S0WS

Base address = 0xF080_F000

Register Name	Address (base+)	Description
GPIOTDATA	0x03FC	PortT Data Register
GPIOTDIR	0x0400	PortT Data Direction Register
GPIOTFR1	0x0424	PortT Function Register1
GPIOTFR2	0x0428	PortT Function Register2
Reserved	0x0804	
Reserved	0x0808	
Reserved	0x080C	
Reserved	0x0810	
Reserved	0x0814	
Reserved	0x0818	
Reserved	0x081C	
Reserved	0x0C00	

1. GPIOTDATA (Port T Data Register)

Address = (0xF080_F000) + (0x03FC)

Bit	Bit Symbol	Type	Reset Value	Bit mask	Description
[31:8]	–	–	Undefined	–	Read as undefined. Written as zero.
[7:0]	PT7 to PT0	R/W	0xFF	Bm7:0	Port T data register

[Description]

a. <PT7 to PT0>

Data register: Stores data.

See notes on data registers for the bit mask function.

2. GPIOTDIR (Port T Data Direction Register)

Address = (0xF080_F000) + (0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:0]	PT7C to PT0C	R/W	0x00	Port T data direction register (for each bit) 0y0: Input 0y1: Output

[Description]

a. <PT7C to PT0C>

Data direction register: Selects input or output for each pin used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOTFR1 (Port T Function Register 1)

Address = (0xF080_F000) + (0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Written as zero.
[7:0]	PT7F1 to PT0F1	R/W	0x00	Port T function register 1

[Description]

a. <PT7F1 to PT0F1>

Function register 1: Controls the function setting.

4. GPIOTFR2 (Port T Function Register2)

Address = (0xF080_F000) +(0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	PT6F2 to PT0F2	R/W	0y0000000	Port T function register 2

(個別説明)

b. <PT6F2 to PT0F2>

Function register 2: Controls the function setting.

Note: The function register 1 and function register 2 can only be set exclusively of each other. These registers must not simultaneously be written as 1 even for an instant.

Table 3.9.8 Function register setting table

Mode	GPIOTFR1	GPIOTFR2
General-purpose	0	0
Function 1	1	0
Function 2	0	1
Prohibited	1	1

3.9.2.7 PORTU

Port U can be used as general-purpose input/output pins pins.

Port U can also be used as NAND controller function (NDD7 to NDD0) and, LCDC (LD7 to LD0).

General-purpose input setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose input	GPIOUDATA	GPIOUDIR	GPIOUFR1	GPIOUFR2
	*	0	0	0

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

General-purpose output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose output	GPIOUDATA	GPIOUDIR	GPIOUFR1	GPIOUFR2
	*	1	0	0

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

NANDC setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
NAND	GPIOUDATA	GPIOUDIR	GPIOUFR1	GPIOUFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NDD7	NDD6	NDD5	NDD4	NDD3	NDD2	NDD1	NDD0

LCDC function setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
LCDC	GPIOUDATA	GPIOUDIR	GPIOUFR1	GPIOUFR2
	*	*	0	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0

base address = 0xF080_4000

Register Name	Address (base+)	Description
GPIOUDATA	0x03FC	PortU Data Register
GPIOUDIR	0x0400	PortU Data Direction Register
GPIOUFR1	0x0424	PortU Function Register1
GPIOUFR2	0x0428	PortU Function Register2
–	0x0804	Reserved
–	0x0808	Reserved
–	0x080C	Reserved
–	0x0810	Reserved
–	0x0814	Reserved
–	0x0818	Reserved
–	0x081C	Reserved
–	0x0C00	Reserved

1. GPIOUDATA (Port U Data Register)

Address = (0xF080_4000) + (0x03FC)

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

[Description]

a. <PU[7:0]>

Data register: Stores data.

See notes on data registers for the bit mask function.

2. GPIOUDIR (Port U Data Direction Register)

Address = (0xF080_4000) + (0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	PU7C to PU0C	R/W	0x00	Port U data direction register (for each bit) 0y0: Input 0y1: Output

[Description]

a. <PU7C to PU0C>

Data direction register: Selects input or output for each pin used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOUFR1 (Port U Function Register1)

Address = (0xF080_4000) +(0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	PU7F1 to PU0F1	R/W	0x00	Port U function register 1

[Description]

a. <PU7F1 to PU0F1>

Function register 1: Controls the function setting.

4. GPIOUFR2 (Port U Function Register2)

Address = (0xF080_4000) + (0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	PU7F2 to PU0F2	R/W	0x00	Port U function register 2

[Description]

a. <PU7F2 to PU0F2>

Function register 2: Controls the function setting.

Note: The function register 1 and function register 2 can only be set exclusively of each other. These registers must not simultaneously be written as 1 even for an instant.

Table 3.9.9 Function register setting table

Mode	GPIOUFR1	GPIOUFR2
General-purpose	0	0
Function 1	1	0
Function 2	0	1
Prohibited	1	1

3.9.2.8 PORTV

Port V can be used as general-purpose input/output pins pins.

Port V can also be used as NAND controller function (NDRBn, NDCE1n, NDCE0n, NDCLE, NDALE, NDWEn and NDREn) and LCDC function (LD15 to LD8).

General-purpose input setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose input	GPIOVDATA	GPIOVDIR	GPIOVFR1	GPIOVFR2
	*	0	0	0

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

General-purpose output setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
General-purpose output	GPIOVDATA	GPIOVDIR	GPIOVFR1	GPIOVFR2
	*	1	0	0

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

NANDC setting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
NAND	GPIOVDATA	GPIOVDIR	GPIOVFR1	GPIOVFR2
	*	*	1	0

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
–	NDRB	NDCE1n	NDCE0n	NDCLE	NDALE	NDWEn	NDREn

LCDCsetting

Function	Data Value	Input/Output Select	Function Select 1	Function Select 2
LCDC	GPIOVDATA	GPIOVDIR	GPIOVFR1	GPIOVFR2
	*	*	0	1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LD15	LD14	LD13	LD12	LD11	LD10	LD9	LD8

base address = 0xF080_7000

Register Name	Address (base+)	Description
GPIOVDATA	0x03FC	PortV Data Register
GPIOVDIR	0x0400	PortV Data Direction Register
GPIOVFR1	0x0424	PortV Function Register1
GPIOVFR2	0x0428	PortV Function Register2
–	0x0804	Reserved
–	0x0808	Reserved
–	0x080C	Reserved
–	0x0810	Reserved
–	0x0814	Reserved
–	0x0818	Reserved
–	0x081C	Reserved
–	0x0C00	Reserved

1. GPIOVDATA (Port V Data Register)

Address = (0xF080_7000) + (0x03FC)

Bit	Bit Symbol	Type	Reset ValVe	Bit mask	Description
[31:8]	–	–	Undefined	–	Read as undefined. Write as zero.
[7:0]	PV[7:0]	R/W	0xFF	Bm7:0	Port V data register

[Description]

a. <PV[7:0]>

Data register: Stores data.

See notes on data registers for the bit mask function.

2. GPIOVDIR (Port V Data Direction Register)

Address = (0xF080_7000) + 0x0400)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	PV7C to PV0C	R/W	0x00	Port V data direction register (for each bit) 0y0: Input 0y1: Output

[Description]

a. <PV7C to PV0C>

Data direction register: Selects input or output for each pin used as a general-purpose port.

0y0: Input

0y1: Output

3. GPIOVFR1 (Port V Function Register1)

Address = (0xF080_7000) + (0x0424)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	PV6F1 to PV0F1	R/W	0y0000000	Port V function register 1

[Description]

a. <PV6F1 to PV0F1>

Function register 1: Controls the function setting.

4. GPIOVFR2 (Port V Function Register2)

Address = (0xF080_7000) + (0x0428)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	PV7F2 to PV0F2	R/W	0x00	Port V function register 2

[Description]

a. <PV7F2 to PV0F2>

Function register 2: Controls the function setting.

Note: The function register 1 and function register 2 can only be set exclusively of each other. These registers must not simultaneously be written as 1 even for an instant.

Table 3.9.10 Function register setting table

Mode	GPIOVFR1	GPIOVFR2
General-purpose	0	0
Function 1	1	0
Function 2	0	1
Prohibited	1	1

3.9.3 Notes

- Procedure for using the interrupt function

  Interrupts can be detected in various modes depending on the sensitivity setting. The following procedure should be observed when the interrupt function is enabled (GPIOxIE = 1) or the interrupt mode settings are modified by GPIOxDIR, GPIOxIS, GPIOxIBE, GPIOxIEV.

  1. Disable interrupts in a relevant bit of the GPIOnDIR register (GPIOnDIR = 0).
  2. Disable interrupts in a relevant bit of the GPIOxIE register (GPIOxIE = 0).
  3. Set a relevant bit of the interrupt mode setting registers (GPIOxIS, GPIOxIBE and GPIOxIEV).
  4. Clear the interrupt in a relevant bit of the GPIOxIC register (GPIOxIC = 1).
  5. Enable interrupts in a relevant bit of the GPIOxIE register (GPIOxIE = 1).

3.10 MPMC

This LSI contains two types of memory controller with different specifications.

Depending on the connected external memory, one of two types of controllers (MPMC0/MPMC1) can be selected by setting the external pin SELMEMC (port SN0).

By setting the external pin SELDVCCM (port SN1) and the internal PMCDRV register, the power supply voltage of memory interface DVCCM can be selected to correspond to 1.8 V or 3.3 V. In the case of using SDRAM, special settings for special pins and registers are required. Required settings are shown in the table below.

Table 3.10.1 Memory controller and Voltage Configurations

Memory controller configuration \ Supply Voltage for External memory				1.8 V ± 0.1 V	3.3 V ± 0.3 V
MPMC0	Pin configuration	SELMEMC (Note1)		0 input	
		SELDVCCM (Note1)		0 input	1 input
		DMCCLKIN		0 input	
	Register configuration	PMCDRV<DRV_MEM1:0>		0y11	0y01
SDRAM is used		16bit_bus	dmc_user_config3	0x00000000	
MPMC1	Pin configuration	SELMEMC (Note1)		1 input	N/A
		SELDVCCM (Note1)		0 input	
		DMCCLKIN		DMCDCLKP connect to External Memory (Note2)	
	Register configuration	PMCDRV<DRV_MEM1:0>		0y11	
SDRAM is used		16bit_bus	dmc_user_config_5	0x00000058	

Note1: The SELMEMC and SELDVCCM pins derive power from DVCC3IO. Therefore, 0 input voltage must be 0 V and 1 input voltage must be 3.3 V.

Note 2: When using MPMC1 to control DDR SDRAM, the feedback clock DMCCLKP for MPMC1 data latch must be input. DMCCLKP must be connected to DMCCLKIN (input pin) as short as possible in designing a board. When using MPMC0 to control SDR SDRAM or not using SDRAM, take a precaution to avoid leak current (e. g. fixing DMCCLKIN pin to GND).

The following shows differences in supported memory between MPMC0 and MPMC1. Select MPMC0 or MPMC1 depending on SDRAM to use.

MPMC0: 16-bit Standard type SDR SDRAM
16-bit Mobile type SDR SDRAM
16-bit NOR Flash (Asynchronous, Separate bus only)
16-bit SRAM (Asynchronous, Separate bus only)

MPMC1: 16-bit LVCMOS type DDR SDRAM
16-bit NOR Flash (Asynchronous, Separate bus only)
16-bit SRAM (Asynchronous, Separate bus only)

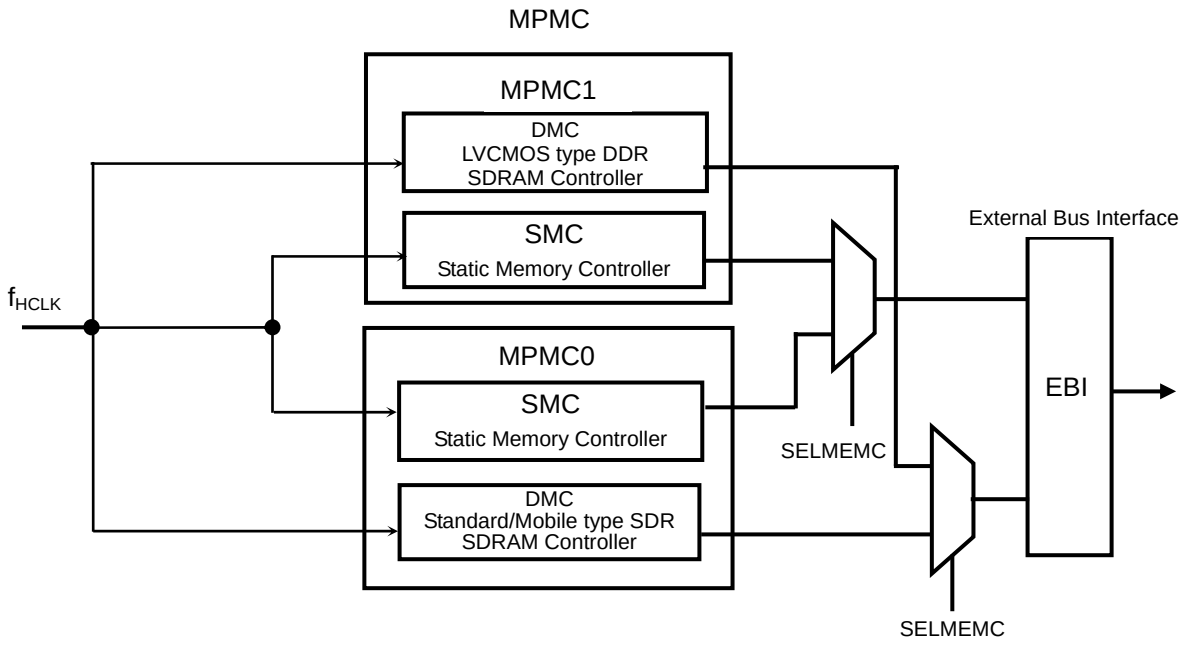
Mode setting pin	Operation mode
SELMEMC	
0	Use MPMC0
1	Use MPMC1

Note 1: SDR SDRAM and DDR SDRAM cannot be used concurrently.

Note 2: The two memory controllers cannot be used by dynamically switching between them. The memory controller to be used must be fixed.

Refer to chapters on respective circuits for details.

The following shows the MPMC block diagram.



According to the voltage of the connected external memory, set pin and register as follows.

Note: The two memory controllers cannot be used by dynamically switching between them. The memory controller to be used must be fixed.

Mode setting pin	Operation mode
SELDVCCM	
0	Control pin of external memory except NAND Flash operate in the DVCCM = 1.8 ± 0.1 V.
1	Control pin of external memory operate in the DVCCM = 3.3 ± 0.3 V.

According to power voltage, adjust drive power of related ports. In the case of using SDRAM, related pin connections and the constant value setting register need be set.

The following table shows the required setting.

Port drive power set register	Operation mode
PMCDRV<DRV_MEM1:0>	
0y11	control pin of external memory except NAND Flash operate in the DVCCM = 1.8 ± 0.1 V
0y01	control pin of external memory operate in the DVCCM = 3.3 ± 0.3 V

Note: The PMCDRV register should be set during low-speed operation (PLL = OFF) after reset is released.

[SDR SDRAM]

Bus width setting register	Operation mode
dmc_user_config_3	
0x00000000	16bit bus in SDR SDRAM (MPMC0)

Note: The dmc_user_config_3 register should be set after reset is released and before SDRAM is initialized. This also applies after HOT_RESET by the PMC is released.

Pin treatment	Operation mode
DMCCLKIN	
This pin isn't used. (Fix DMCCLKIN to GND)	16 bit bus in SDR SDRAM (MPMC0)

[DDR SDRAM]

Bus width setting register	Operation mode
dmc_user_config_5	
0x00000058	16bit bus in DDR SDRAM (MPMC1)

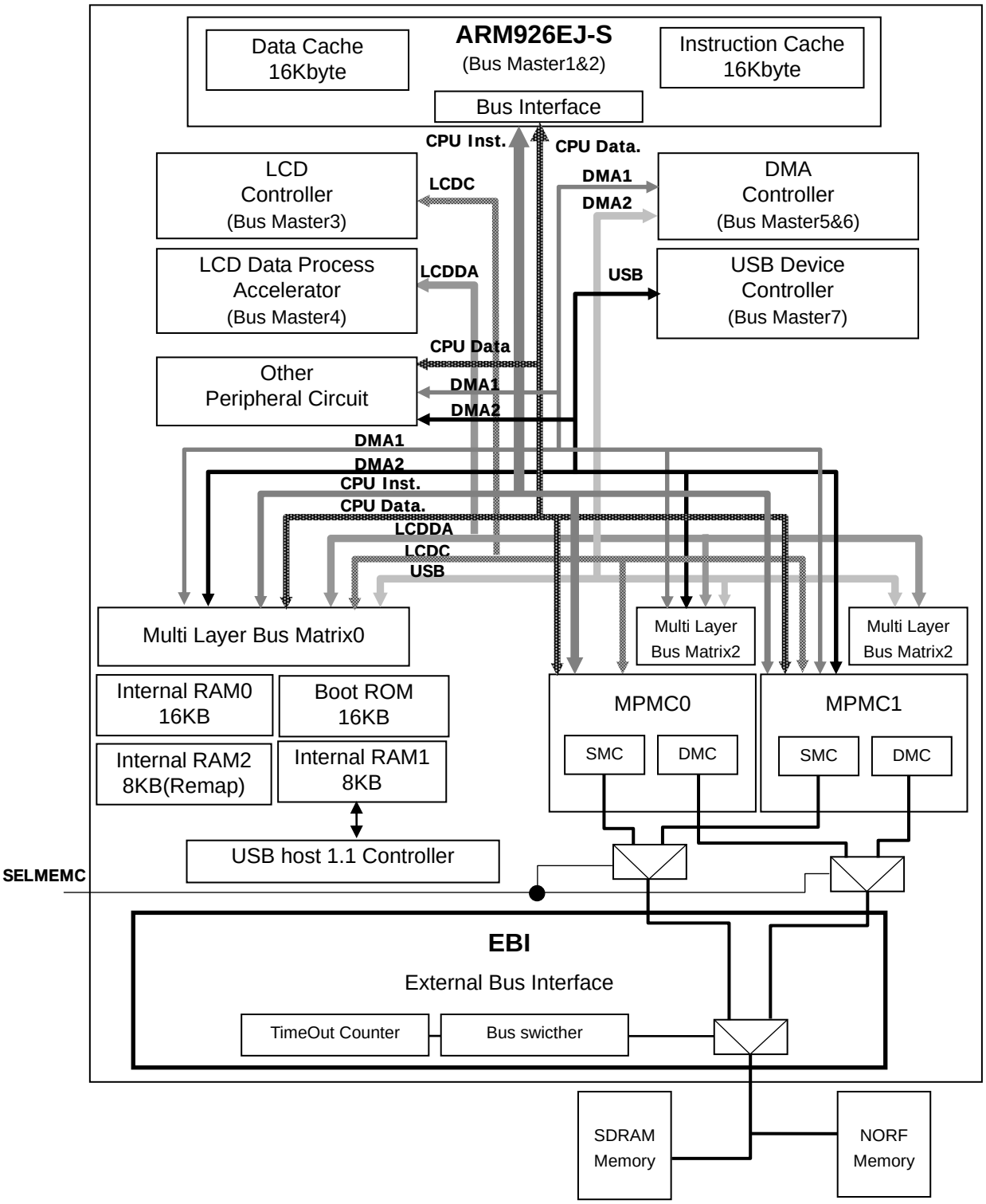
Note: The dmc_user_config_5 register should be set after reset is released and before SDRAM is initialized. This also applies after HOT_RESET by the PMC is released.

pin treatment	Operation mode
DMCCLKIN	
Connect DMCCLKIN to DMCCLKP	16bit bus in DDR SDRAM (MPMC1)

3.10.1 EBI (External Bus Interface)

Memory controllers (MPMC0 and MPMC1) have a built-in SMC (Static Memory Controller) circuit and DMC (Dynamic Memory Controller) circuit.

The external bus of SMC is used also as the external bus of DMC in the TMPA901CM. However, SMC and DMC function as independent circuits in memory controller. DMC and SMC circuits are controlled by EBI (External Bus Interface).



EBI shifts the bus according to the access request from memory controller (DMC and SMC). If two Access requests of DMC and SMC are generated, EBI keeps the one Access request wait, when the other is accessing.

To avoid the one Access request is made to wait for a long time when one Access request is generated continuously, EBI manage the overlapped time, also it has a "Timeout counter"; the bus is released forcibly.

In the TMPA901CM, the higher the access speed and the frequency become, the higher the priority of the DMC becomes..

Therefore, it has function to prioritize DMC request by setting Timeout cycle of SMC side to register.

Table 3.10.2 Timeout for EBI

DMC time out cycle	SMC time out cycle
1024 clocks (Fixed)	to 1024 clocks (configurable with register)

SMC timeout cycle setting register

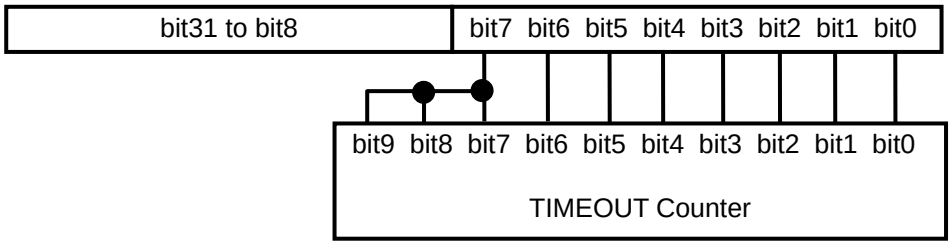
Base address = 0xF00A_0000

Register Name	Address (base+)	Type	Reset value	Description
smc_timeout	0x0050	R/W	0x000000FF	SMC Timeout Register

Note: "0x00000000" cannot be set. "0x00000001 to 0x000000FF" only is effective.

The smc_timeout cycle is controlled by a 10 bit counter, however, the effective bits in control register are Low-order 8bits only. The most significant bit (bit 7) of effective bits controls High-order 3bits of the 10 bit counter.

smc_timeout register



Note: To avoid an underflow in LCDC when setting DMC memory (SDRAM) to VRAM of LCDC,It is recommended to set this register to 0y01.Please use this function together with the QOS function (refer to "DMC" section)

3.10.2 Overview of MPMC0

MPMC0 contains both a DMC (Dynamic Memory Controller) that controls SDRAM and SMC (Static Memory Controller) that controls NOR Flash and SRAM.

Features of a DMC (Dynamic Memory Controller):

a. Supports 16-bit SDR SDRAM

b. Supports 1 channel Chip Select

c. Supports clock-basis adjusting function for SDRAM request timing.

Features of an SMC (Static Memory Controller):

(a) Supports asynchronous, 16-bit SRAM and NOR Flash (only separate buses are supported, and multiplex buses are not supported)

(b) Supports 2 channels Chip Select

(c) Cycle timings and memory data bus widths can be programmed for each Chip Select

3.10.3 Functions of MPMC0

Figure 3.10.1 is a simplified block diagram of MPMC0 circuits.

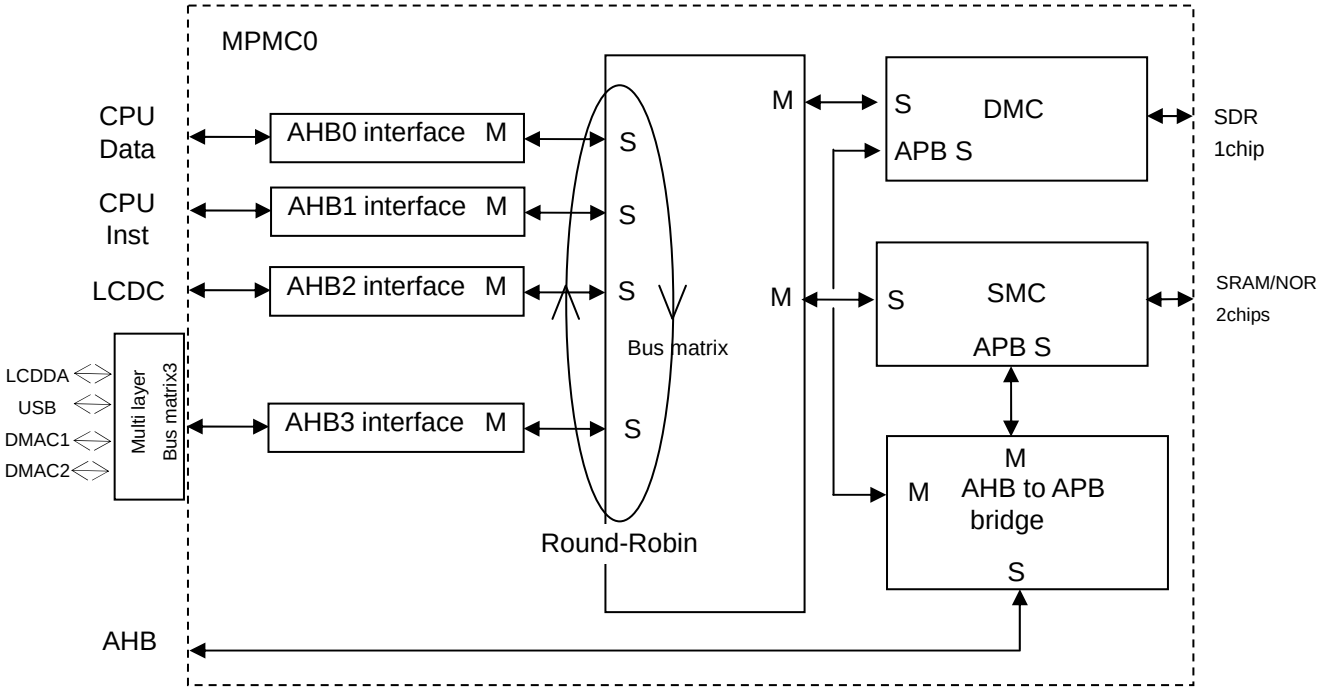
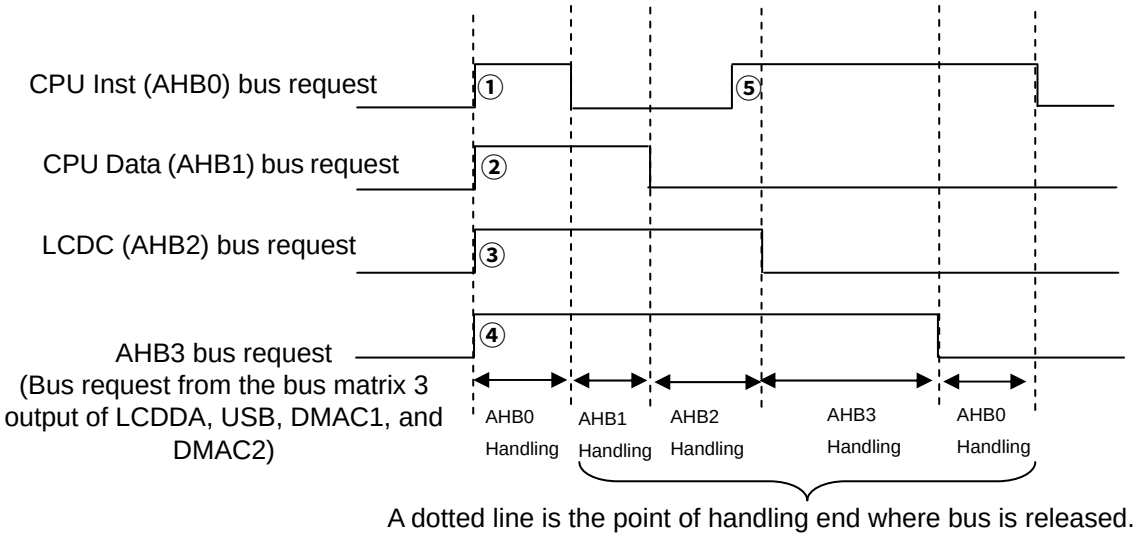


Figure 3.10.1 MPMC0 Block Diagram

(a) Bus matrix

1. Bus matrix of AHB0, AHB1, AHB2 and AHB3 supports Round-Robin arbitration scheme. The following diagram shows the priority of bus requests.



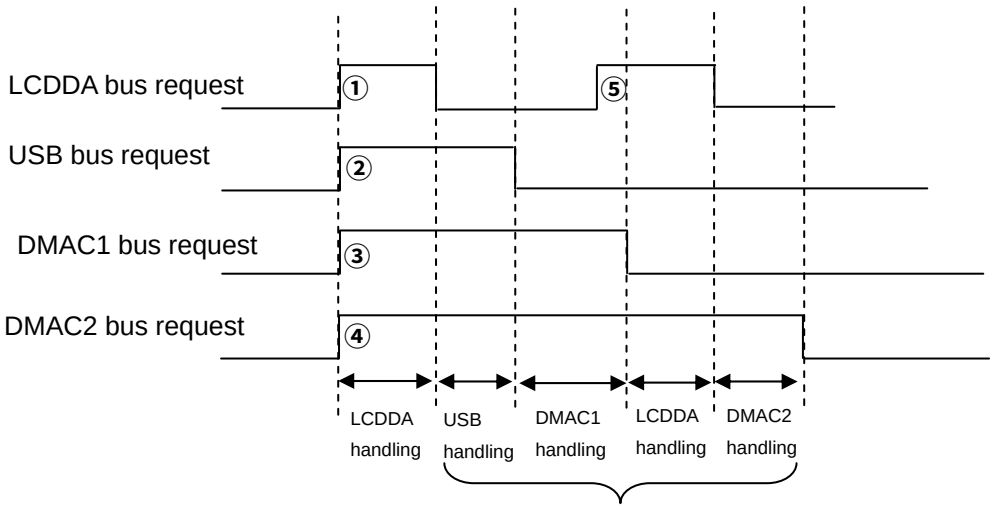
A dotted line is the point of handling end where bus is released.

Handling priority : ①→②→③→④→⑤

2. Bus matrix 2 of LCDDA, DMAC1, DMAC2 and USB handles the earliest bus request first. If multiple bus requests are accepted simultaneously, they are prioritized as shown below.

LCDDA > USB > DMAC1 > DMAC2

Following diagram show the priority of bus request.



Handling priority : ①→②→③→⑤→④

(b) Clock Variety

Control clock is controlled in PLLCG circuit.

1. Dynamic memory clock: Use HCLK clock
2. Static memory clock: Use HCLK or 1/2 HCLK

(Set CLKCR5<SEL_SMC_MCLK>)

3.10.3.1 DMC (Dynamic Memory Controller)

(1) DMC function outline

Table 3.10.3 shows features of DMC.

Table 3.10.3 Features of DMC

	Features
Support memory	SDR SDRAM Support separate bus only
Data bus width	16 bit data bus width
Access areas	Max 512MB access area Chip select: DMCCSn only
Timing adjustment	Adjustable AC timing by register
Command	Mode Register setting, Auto refresh, Self Refresh, Active, Precharge, Read/Write command, Powerdown etc.
Clock	DMCSCLK frequency = f_{HCLK} Fixed to GND (Input clock pin DMCCLKIN can not be used)
External control pin	D15 to D0, A23 to A0, DMCSDQM1, DMCSDQM0, DMCCSn, DMCWEn, DMCRASn, DMCCASn, DMCBA0, DMCBA1, DMCCKE, DMCSCLK, DMCDCLKN, DMCAP

(2) DMC block diagram

Figure 3.10.2 is a DMC block diagram.

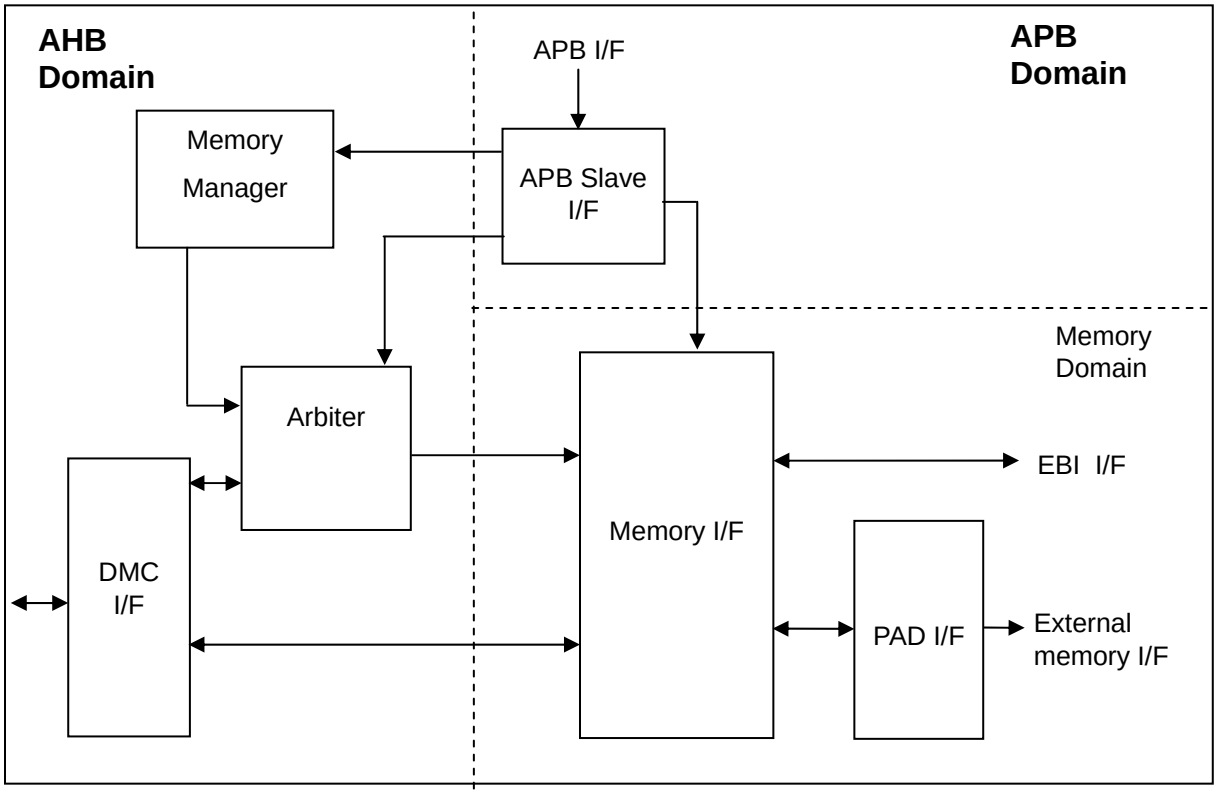


Figure 3.10.2 DMC Block Diagram

(a) Arbiter

The arbiter receives access commands from the DMC I/F and the memory manager, and after access arbitration, it passes the highest priority command to the memory I/F.

Data is read from the memory I/F to the DMC I/F.

(b) Memory manager

The memory manager monitors and controls status of DMC block.

(3) DMC Function operation

(a) Arbiter operation

1. read/write access arbitration
2. For read accesses, QoS (Quality of Service) is provided.
3. Hazard processing

   When selfsame stand-alone bus master access to an external memory, the actual access procedure to memory is executed in the instruction order.

   However, if multiple bus master access to an external memory, the read and write data will be stored temporary into independent buffer and be executing by priority circuit. Therefore, the read and the write instruction may switch execution sequence. So please coordinate a variety of sequences, e.g. making an enough time for next instruction, checking whether or not previous execution is finished, the common-use memory data uses the internal memory and so on.
4. Monitoring the state machine and select an entry of the proper pipeline.

(b) Memory manager operation

1. Monitor and control DMC circuit
2. Issuing direct commands
   - NOP
   - Prechargeall
   - Autorefresh
   - Modereg
   - Extended modereg
3. Auto Refresh function is provided

   Set Auto Refresh timing by 15bit counter

(c) Memory interface operation

According to use, there are three kinds of built-in FIFOs

1. command FIFO: 2 words
2. read data FIFO: 10 words
3. write data FIFO: 10 words

* As the FIFO sizes of either read or write FIFO is 10 words,
the max size for one transfer is 8 words. (1word = 32 bit data)

(d) Low_power function

DMC provide 2 kinds of Low_power modes.

1. By setting dmc_memc_cmd_3 register, Self Refresh Mode is available.
2. By setting dmc_memory_cfg_3 register, either of the two modes is available: Clock Suspend Mode to stop memory clock (DMCCLK) or Power Down Mode to make the CKE pin (CKE = low) invalid automatically when there is no memory access.

Note: Clock Suspend Mode function and Power Down mode cannot be used concurrently.

(e) QoS Function

The QoS function is available in read-accessing only.

The QoS function is the service function for exception handling at Round-Robin which is controlled by Bus matrix for MPMC. This function is available in read-accessing only.

dmc_id_x_cfg_3<qos_min> is set by a register within the DMC on a port by port basis. dmc_id_x_cfg_3<qos_min> indicates a required read maximum latency.

A QoS_max timeout causes the transaction to be raised to a higher priority.

You can also set the dmc_id_x_cfg_3<qos_min> to enable for a specific port so that its transfers are serviced with a higher priority.

This impacts the overall memory band width because it limits the options of the scheduling algorithm.

If dmc_id_x_cfg_3<qos_enable> enable bit for the port is set in the register bank, the qos_max latency value is decremented every cycle until it reaches zero.

If the entry is still in the queue when the Internal counter value reaches zero then the entry becomes the highest priority. This is called a *time-out*.

If qos_min is set to enable, qos_max value is ignored and it always becomes the highest priority.

Table 3.10.4 SDR Memory Setup Example

Base address = 0xF430_0000

Register address	Write data	Description
0x0014	0x00000006	Set cas_Latency to 3
0x0018	0x00000000	Set t_dqss to 0
0x001C	0x00000002	Set t_mrd to 2
0x0020	0x00000007	Set t_ras to 7
0x0024	0x0000000B	Set t_rc to 11
0x0028	0x00000015	Set t_rcd to 5 and schedule_rcd to 2
0x002C	0x000001F2	Set t_rfc to 18 and schedule_rfc to 15
0x0030	0x00000015	Set t_rp to 5 and schedule_rp to 2
0x0034	0x00000002	Set t_rrd to 2
0x0038	0x00000003	Set t_wr to 3
0x003C	0x00000002	Set t_wtr to 2
0x0040	0x00000001	Set t_xp to 1
0x0044	0x0000000A	Set t_xsr to 10
0x0048	0x00000014	Set t_esr to 20
0x000C	0x00010020	Set memory configuration
0x0010	0x00000A60	Set auto refresh period to be every 2656 DMCSCLK periods
0x0200	0x000000FF	Set chip select for chip 0 to be 0x00XXXXXX, rbc configuration
0x0008	0x000C0000	Carry out chip 0 Nop command
0x0008	0x00000000	Carry out chip 0 Prechargeall command
0x0008	0x00040000	Carry out chip 0 Autorefresh command
0x0008	0x00040000	Carry out chip 0 Autorefresh command
0x0008	0x00080032	Carry out chip 0 Mode Reg command 0x32 mapped to low add bits
0x0004	0x00000000	Change DMC state to Ready

(4) DMC register description of MPMC0

Table 3.10.5 DMC SFR list of MPMC0

Base address = 0xF430_0000

Register Name	Address (base +)	Type	Reset Value	Description
dmc_memc_status_3	0x0000	RO	0x00000380	DMC Memory Controller Status Register
dmc_memc_cmd_3	0x0004	WO	–	DMC Memory Controller Command Register
dmc_direct_cmd_3	0x0008	WO	–	DMC Direct Command Register
dmc_memory_cfg_3	0x000C	R/W	0x00010020	DMC Memory Configuration Register
dmc_refresh_prd_3	0x0010	R/W	0x00000A60	DMC Refresh Period Register
dmc_cas_latency_3	0x0014	R/W	0x00000006	DMC CAS Latency Register
dmc_t_dqss_3	0x0018	R/W	0x00000001	DMC t_dqss Register
dmc_t_mrd_3	0x001C	R/W	0x00000002	DMC t_mrd Register
dmc_t_ras_3	0x0020	R/W	0x00000007	DMC t_ras Register
dmc_t_rc_3	0x0024	R/W	0x0000000B	DMC t_rc Register
dmc_t_rcd_3	0x0028	R/W	0x0000001D	DMC t_rcd Register
dmc_t_rfc_3	0x002C	R/W	0x00000212	DMC t_rfc Register
dmc_t_rp_3	0x0030	R/W	0x0000001D	DMC t_rp Register
dmc_t_rrd_3	0x0034	R/W	0x00000002	DMC t_rrd Register
dmc_t_wr_3	0x0038	R/W	0x00000003	DMC t_wr Register
dmc_t_wtr_3	0x003C	R/W	0x00000002	DMC t_wtr Register
dmc_t_xp_3	0x0040	R/W	0x00000001	DMC t_xp Register
dmc_t_xsr_3	0x0044	R/W	0x0000000A	DMC t_xsr Register
dmc_t_esr_3	0x0048	R/W	0x00000014	DMC t_esr Register
dmc_id_0_cfg_3 dmc_id_1_cfg_3 dmc_id_2_cfg_3 dmc_id_3_cfg_3	0x0100 0x0104 0x0108 0x010C	R/W	0x00000000	DMC id_<0-3>_cfg Registers
dmc_chip_0_cfg_3	0x0200	R/W	0x0000FF00	DMC chip_0_cfg Registers
Reserved	0x0204	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0208	–	Undefined	Read as undefined. Write as zero.
Reserved	0x020C	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0300	–	Undefined	Read as undefined. Write as zero.
dmc_user_config_3	0x0304	WO	Undefined	DMC user_config Register
Reserved	0x0E00	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E04	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E08	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FE0-0x0FEC	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FF0-0x0FFC	–-	Undefined	Read as undefined. Write as zero.

Note: The APB supports only single-word 32-bit accesses. Read from or write to registers at single-word 32-bit mode.

MPMC0

The status of register read/write access (dmc_memc_status_3 status)

○:permitted ×:prohibited

Register Name	Type	Read				Write			
		dmc_memc_status_3				dmc_memc_status_3			
		config	Ready	Paused	Low_power	config	Ready	Paused	Low_power
dmc_memc_status_3	RO	○	○	○	○	–	–	–	–
dmc_memc_cmd_3	WO	–	–	–	–	○	○	○	○
dmc_direct_cmd_3	WO	–	–	–	–	○	×	×	×
dmc_memory_cfg_3	R/W	○	○	○	○	○	×	×	×
dmc_refresh_prd_3	R/W	○	○	○	○	○	×	×	×
dmc_cas_latency_3	R/W	○	○	○	○	○	×	×	×
dmc_t_dqss_3	R/W	○	○	○	○	○	×	×	×
dmc_t_mrd_3	R/W	○	○	○	○	○	×	×	×
dmc_t_ras_3	R/W	○	○	○	○	○	×	×	×
dmc_t_rc_3	R/W	○	○	○	○	○	×	×	×
dmc_t_rcd_3	R/W	○	○	○	○	○	×	×	×
dmc_t_rfc_3	R/W	○	○	○	○	○	×	×	×
dmc_t_rp_3	R/W	○	○	○	○	○	×	×	×
dmc_t_rrd_3	R/W	○	○	○	○	○	×	×	×
dmc_t_wr_3	R/W	○	○	○	○	○	×	×	×
dmc_t_wtr_3	R/W	○	○	○	○	○	×	×	×
dmc_t_xp_3	R/W	○	○	○	○	○	×	×	×
dmc_t_xsr_3	R/W	○	○	○	○	○	×	×	×
dmc_t_esr_3	R/W	○	○	○	○	○	×	×	×
dmc_id_0_cfg_3 dmc_id_1_cfg_3 dmc_id_2_cfg_3 dmc_id_3_cfg_3	R/W	○	○	○	○	○	×	×	○
dmc_chip_0_cfg_3	R/W	○	○	○	○	○	×	×	○
dmc_user_config_3	WO	–	–	–	–	○	○	○	○

1. dmc_memc_status_3 (DMC Memory Controller Status Register)

Address = (0xF430_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined.
[9]	memory_banks	RO	0y1	Setting value of the maximum number of banks that the DMC supports: (Fixed to 4 banks)
[8:7]	Reserved	–	Undefined	Read as undefined.
[6:4]	memory_ddr	RO	0y000	Types of SDRAM that the DMC supports: 0y000 = SDR SDRAM 0y001 = Reserved 0y011 = Reserved 0y010 = Reserved 0y1xx = Reserved
[3:2]	memory_width	RO	0y01	External memory bus width: 0y00 = 16-bit 0y01 = Reserved 0y10 = Reserved 0y11 = Reserved
[1:0]	memc_status	RO	0y00	Memory controller status: 0y00 = Config 0y01 = Ready 0y10 = Paused 0y11 = Low-power

[Description]

a. <memory_banks>

Setting value of the maximum number of banks that the DMC supports:
Fixed to 4 banks.

b. <memory_ddr>

Types of SDRAM that the DMC supports.
Fixed to 0y000 (SDR SDRAM)

c. <memory_width>

External memory bus width:
0y00 = 16-bit
0y01 = Reserved
0y10 = Reserved
0y11 = Reserved.

d. <memc_status>

Memory controller status:
0y00 = Config
0y01 = Ready
0y10 = Paused
0y11 = Low-power

2. dmc_memc_cmd_3 (DMC Memory Controller Command Register)

Address = (0xF430_0000) + (0x0004)

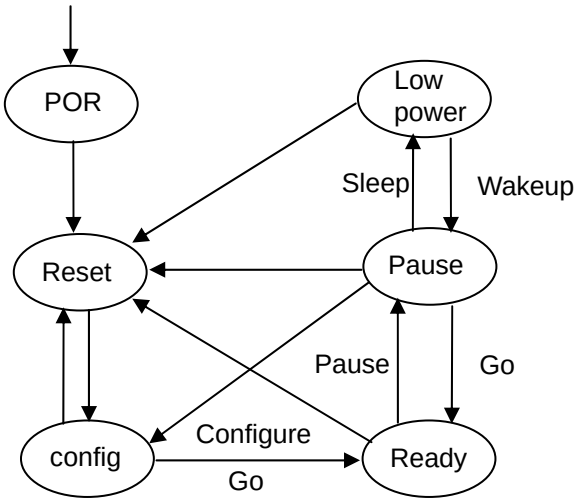
Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	memc_cmd	WO	–	Change the memory controller status: 0y000 = Go 0y001 = Sleep 0y010 = Wakeup 0y011 = Pause 0y100 = Configure

[Description]

a. <memc_cmd>

Settings of this register can change the DMC state machine. If a previously issued command for changing the states is being executed, a new command is issued after the previous command is completed.

The following diagram shows DMC state transitions for Low-power.



DMC State Transitions

When the DMC exits the Reset state, it automatically enters the Config state. The state transition from Pause to Config is effected by a Config command. Register settings must be made during the Config state.

When the DMC state is shifted to Ready, reads from and writes to the SDRAM are allowed. When a read or write is executed, the SDRAM will change from IDLE to ACTIVE.

When the DMC state is Ready, a Pause command shifts the DMC to Pause. The SDRAM state at this time varies depending on the immediately preceding command executed on the SDRAM. If a Read or Write has been executed, the SDRAM will be shifted to ACTIVE. If a AutoRefresh has been executed, the SDRAM will be shifted to IDLE (Note).

When the DMC state is shifted from Pause to Low power by a Sleep command, after All Bank Precharge is executed, CKE will be driven “L” and the SDRAM will automatically enter the Self-refresh state.

When the DMC state is shifted from Low power to Pause by a Wakeup command, a Self-refresh Exit command will be issued. The SDRAM then automatically exists the Self-refresh state and enters the IDLE state.

Note: The SDRAM can be shifted from ACTIVE to IDLE by either of the following two settings: dmc_direct_cmd_3< memory_cmd>0y00 = Prechargeall or 0y01 = Autorefresh

3. dmc_direct_cmd_3 (DMC Direct Command Register)

This register sets each command for external memory and external memory mode register.
This register sets the initial setting of external memory.

Address = (0xF430_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:22]	–	–	Undefined	Read as undefined. Write as zero.
[21:20]	chip_nmbr	WO	–	Always write 0y00.
[19:18]	memory_cmd	WO	–	Determines the command required: 0y00 = Prechargeall 0y01 = Autorefresh 0y10 = Modereg or Extended Modereg 0y11 = NOP
[17:16]	bank_addr	WO	–	Bits mapped to external memory bank address bits when command is Modereg access. 0y00 = bank0 0y01 = bank1 0y10 = bank2 0y11 = bank3
[15:14]	–	–	Undefined	Read as undefined. Write as zero.
[13:0]	addr_13_to_0	WO	–	Bits mapped to external memory address bits [13:0] when command is Modereg access.

a. <memory_cmd>

Determines the command required:
0y00 = Prechargeall
0y01 = Autorefresh
0y10 = Modereg or Extended Modereg
0y11 = NOP

b. <bank_addr>

Bits mapped to external memory bank address bits when command is Modereg access.
0y00 = bank0
0y01 = bank1
0y10 = bank2
0y11 = bank3

c. <addr_13_to_0>

Bits mapped to external memory address bits [13:0] when command is Modereg access.

4. dmc_memory_cfg_3 (DMC Memory Configuration Register)

Address = (0xF430_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:23]	–	–	Undefined	Read as undefined. Write as zero.
[22:21]	active_chips	R/W	0y00	Always write 0y00
[20:18]	–	–	Undefined	Read as undefined. Write as zero.
[17:15]	memory_burst	R/W	0y010	Set the read/write access burst length for the SDRAM 0y000 = Burst 1 0y001 = Burst 2 0y010 = Burst 4 0y011 = Burst 8 0y100 = Burst 16 (Note) Other = Reserved
[14]	stop_mem_clock	R/W	0y0	memory clock stop: 0y0 = Disable 0y1 = Enable
[13]	auto_power_down	R/W	0y0	SDRAM auto Power down Enable: 0y0 = Disable 0y1 = Enable
[12:7]	power_down_prd	R/W	0y000000	Number of SDRAM automatic power-down memory clocks: (Min. value = 1) 0y000001 to 0y111111
[6]	ap_bit	R/W	0y0	The position of the auto-precharge bit in the memory address: 0y0 = address bit 10 0y1 = address bit 8
[5:3]	row_bits	R/W	0y100	The number of row address bits: 0y000 = 11 bits 0y001 = 12 bits 0y010 = 13 bits 0y011 = 14 bits 0y100 = 15 bits 0y101 = 16 bits Other = Reserved
[2:0]	column_bits	R/W	0y000	The number of column address bits: 0y000 = 8 bits 0y001 = 9 bits 0y010 = 10 bits 0y011 = 11 bits 0y100 = 12 bits Other = Reserved

[Description]

a. <memory_burst>

Set the read/write access burst length for the controller.

You must program this value to match the memory burst length set in dmc_direct_cmd_3

b. <stop_mem_clock>

The clock supply to the SDRAM can be stopped while it is not being accessed. When an SDRAM access request occurs again, the clock is automatically restarted.

Note 1: Depending on the SDRAM type, it may not be possible to stop the clock supply to the SDRAM while it is not being accessed. When using this function, be sure to carefully check the specifications of the SDRAM to be used.

Note 2: The memory clock stop function and the SDRAM auto powerdown function cannot be used concurrently. Use only either of the two.

c. <auto_power_down>

When no SDRAM access request is present and the command FIFO of the memory controller becomes empty, the SDRAM can be placed into Powerdown mode by automatically disabling CKE after the number of clock cycles specified in the power_down_prd field. When an SDRAM access request occurs again, CKE is automatically enabled to exit the Powerdown mode.

Note: The memory clock stop function and the SDRAM auto powerdown function cannot be used concurrently. Use only either of the two.

d. <row_bits>, <column_bits>

These bits set the row and column addresses.

Supported selectable memory is limited by the summation of column address and row address.

In case of 16bit bus, less than R+C=26 bits (128Mbytes) then 512Mbytes for 4 banks

5. dmc_refresh_prd_3 (DMC Refresh Period Register)

Address = (0xF430_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:15]	–	–	Undefined	Read as undefined. Write as zero.
[14:0]	refresh_prd	R/W	0x0A60	Auto-refresh cycle (number of memory clocks): 0x0000 to 0x7FFF

[Description]

a. <refresh_prd>

The value of the refresh counter decrements from the value set in the dmc_refresh_prd_3 (the number of Memory clocks), and when the counter reaches zero, auto-refresh requests are occured to external memory.

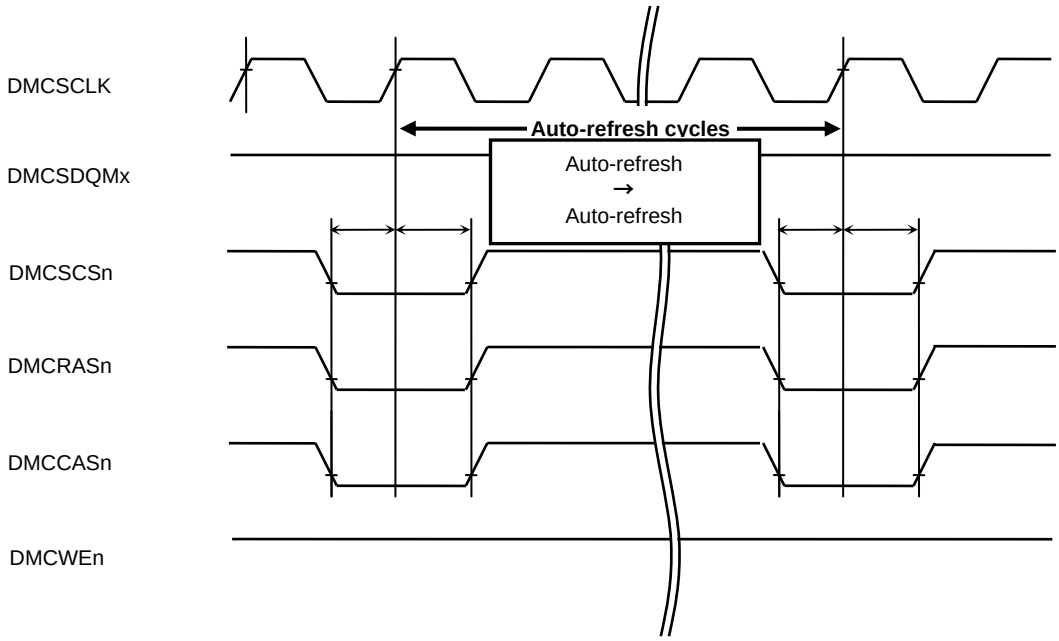


Figure 3.10.3 Auto-refresh Cycles Operation Example

6. dmc_cas_latency_3 (DMC CAS Latency Register)

Address = (0xF430_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:1]	cas_latency	R/W	0y011	CAS latency setting (number of memory clocks): 0y000 to 0y111
[0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <cas_latency>

CAS latency setting (number of memory clocks): 0y000 to 0y111

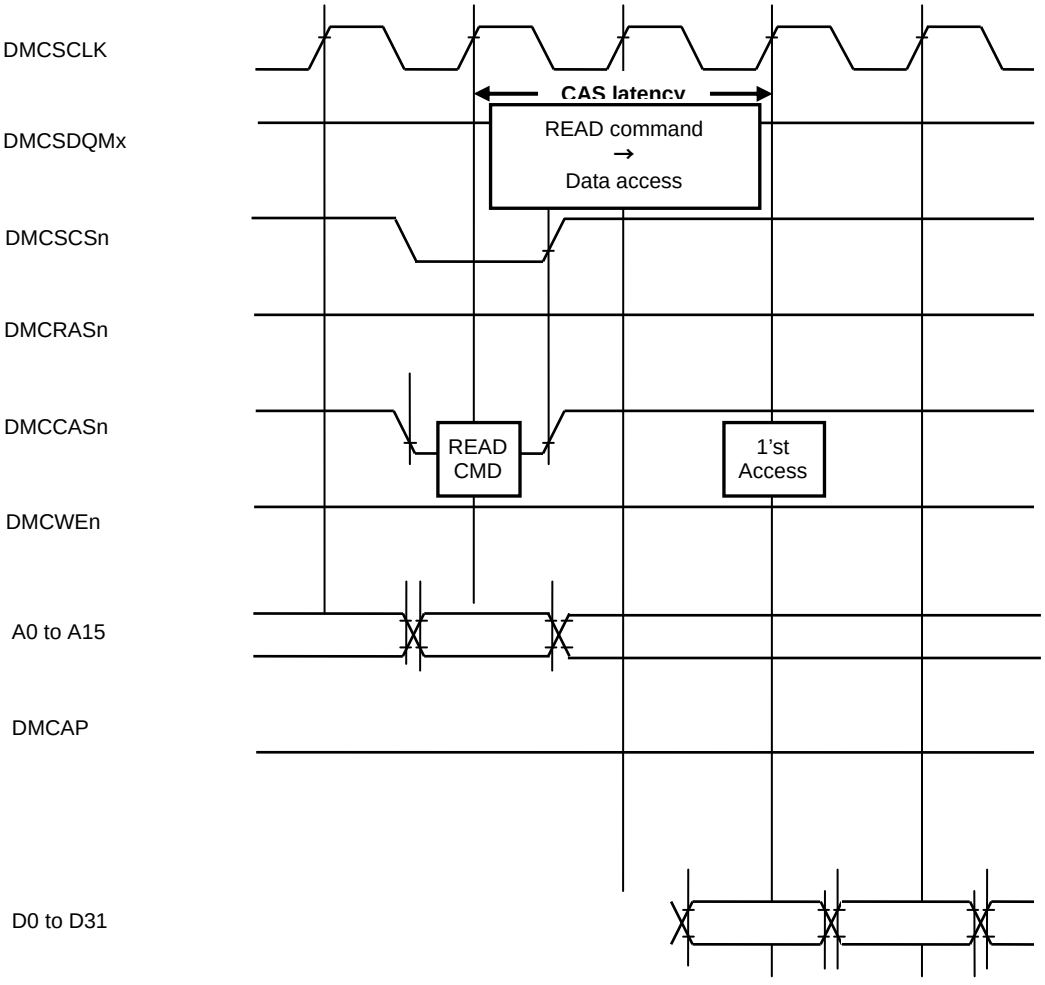


Figure 3.10.4 CAS Latency Example (CL = 2)

7. dmc_t_dqss_3 (DMC t_dqss Register)

Address = (0xF430_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Write as zero.
[1:0]	t_dqss	R/W	0y01	DQS setting (number of memory clocks) In the initial state (before operation), fix to 0y00

[Description]

* The DQS signal is not available in MPMC0. <t_dqss> must be set to 0y00 in initial setting.

8. dmc_t_mrd_3 (DMC t_mrd Register)

Address = (0xF430_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	t_mrd	R/W	0y0000010	Mode register command time (Number of memory clocks): 0x00 to 0x7F

[Description]

a. <t_mrd>
Set time (memory clocks) from mode register command (set by dmc_direct_cmd_3<addr_13_to_0>) to other command:
0x00 to 0x7F

* Depending on other AC settings and operations, the actual delay time may be longer than the specified time. Set the minimum number of clocks in this register.

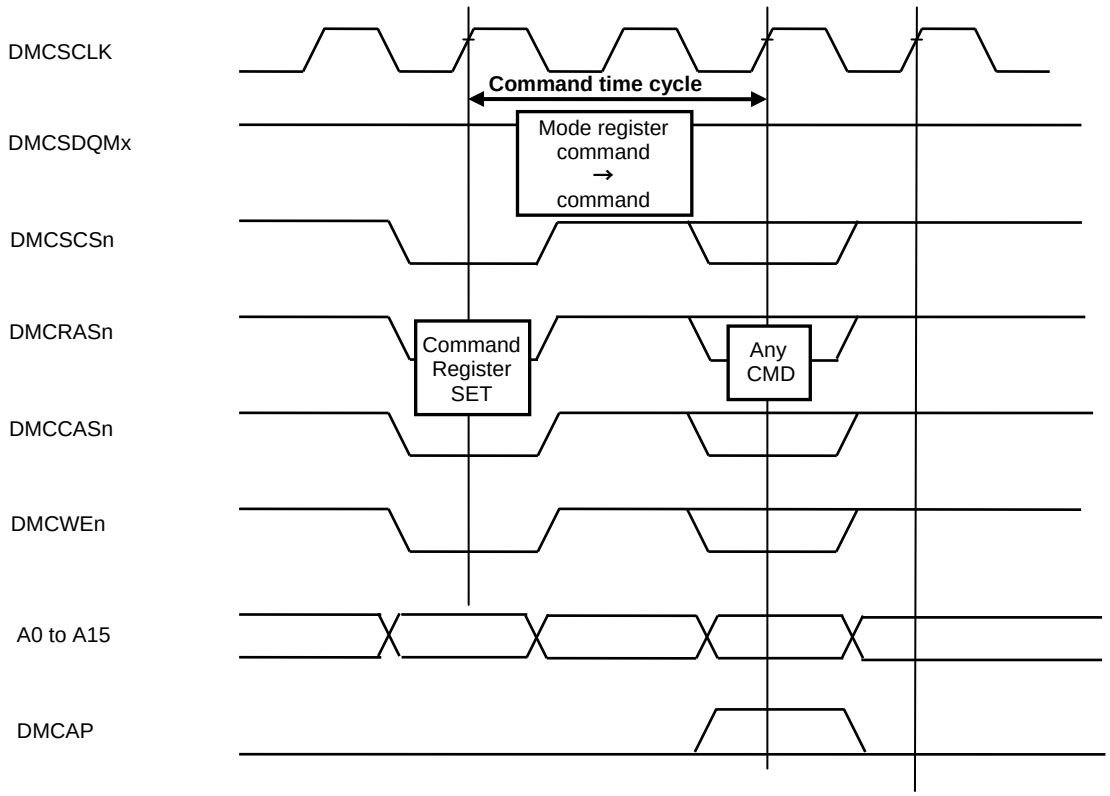


Figure 3.10.5 Example of transition from mode register write to other commands

9. dmc_t_ras_3 (DMC t_ras Register)

Address = (0xF430_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	t_ras	R/W	0x7	Time between RAS and Precharge (number of memory clocks): 0x0 to 0xF

[Description]

a. <t_ras>
Time between RAS and Precharge (number of memory clocks):
0x0 to 0xF

* Depending on other AC settings and operations, the actual delay time may be longer than the specified time. Set the minimum number of clocks in this register.

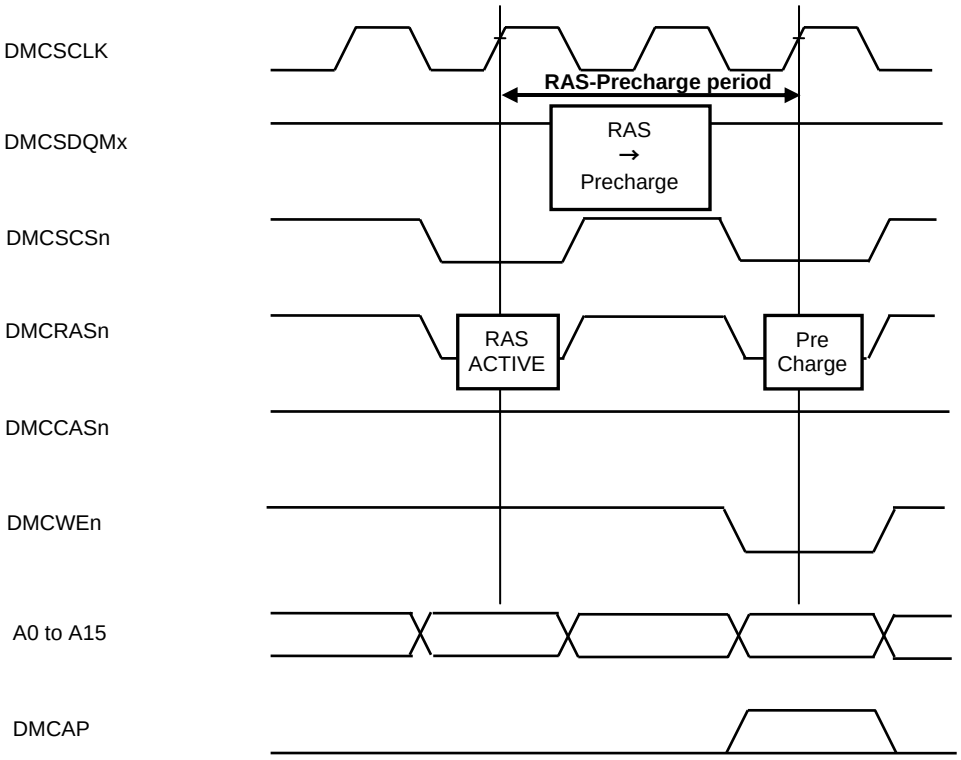


Figure 3.10.6 Time from Active to Precharge

10. dmc_t_rc_3 (DMC t_rc Register)

Address = (0xF430_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	t_rc	R/W	0y1011	Delay between Active bank A and next Active bank A(Number of memory clocks) 0x0 to 0xF

[Description]

a. <t_rc>
The delay time from Active bank command to Active bank command in the same BANK. (memory clocks)
0x0 to 0xF

* Depending on other AC settings and operations, the actual delay time may be longer than the specified time. Set the minimum number of clocks in this register.

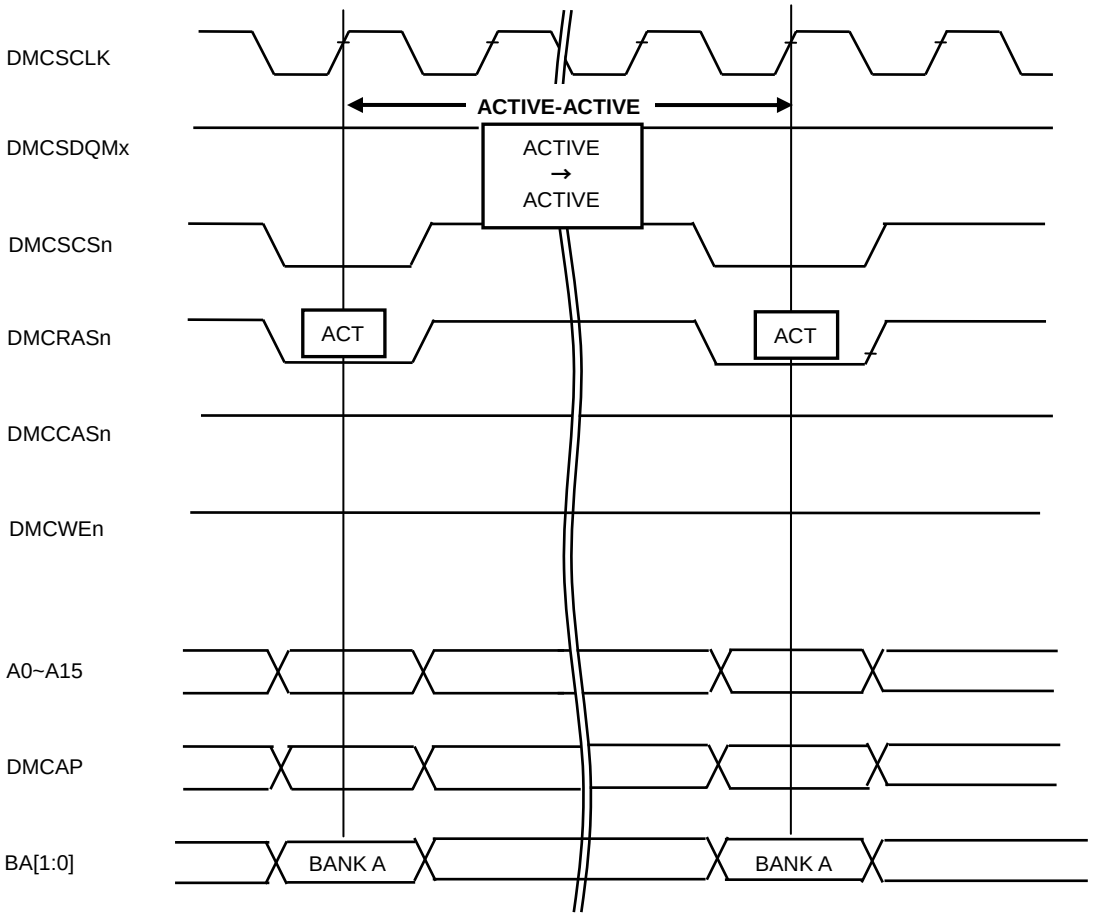


Figure 3.10.7 Delay Time between Two Successive Active Bank As

11. dmc_t_rcd_3 (DMC t_rcd Register)

Address = (0xF430_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:3]	schedule_rcd	R/W	0y011	Set min delay from RAS to CAS. Set to (t_rcd setting value -3)
[2:0]	t_rcd	R/W	0y101	Set min delay from RAS to CAS. (Number of memory clocks): 0y000 to 0y111

[Description]

a. <schedule_rcd>
Set min delay from RAS to CAS. (Number of memory clocks)
Set to (t_rcd setting value -3).

b. <t_rcd>
Set min delay from RAS to CAS (Number of memory clocks):
0y000 to 0y111

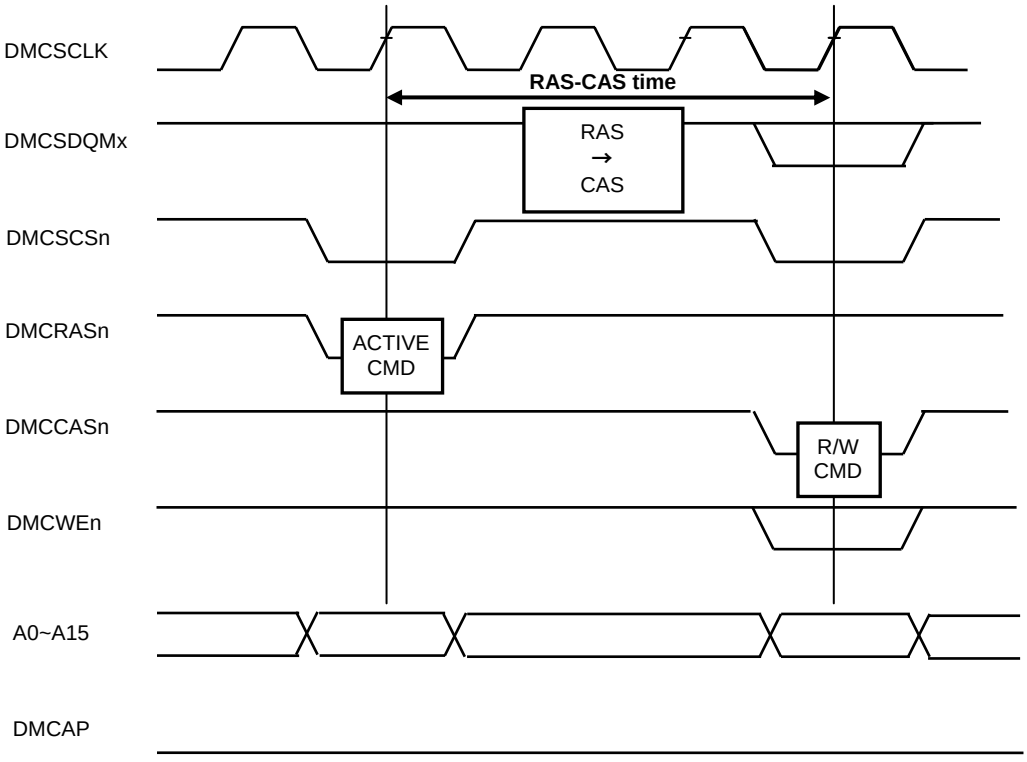


Figure 3.10.8 Time from Active to read command

12. dmc_t_rfc_3 (DMC t_rfc Register)

Address = (0xF430_0000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:5]	schedule_rfc	R/W	0y10000	Autorefresh command time setting Set to (t_rfc setting value -3)
[4:0]	t_rfc	R/W	0y10010	Autorefresh command time setting (Number of memory clocks) 0y00000 to 0y11111

[Description]

a. <schedule_rfc>
Autorefresh command time setting.
Set to (t_rfc setting value -3).

a. <t_rfc>
Autorefresh command time setting (Number of memory clocks):
0y00000 to 0y11111

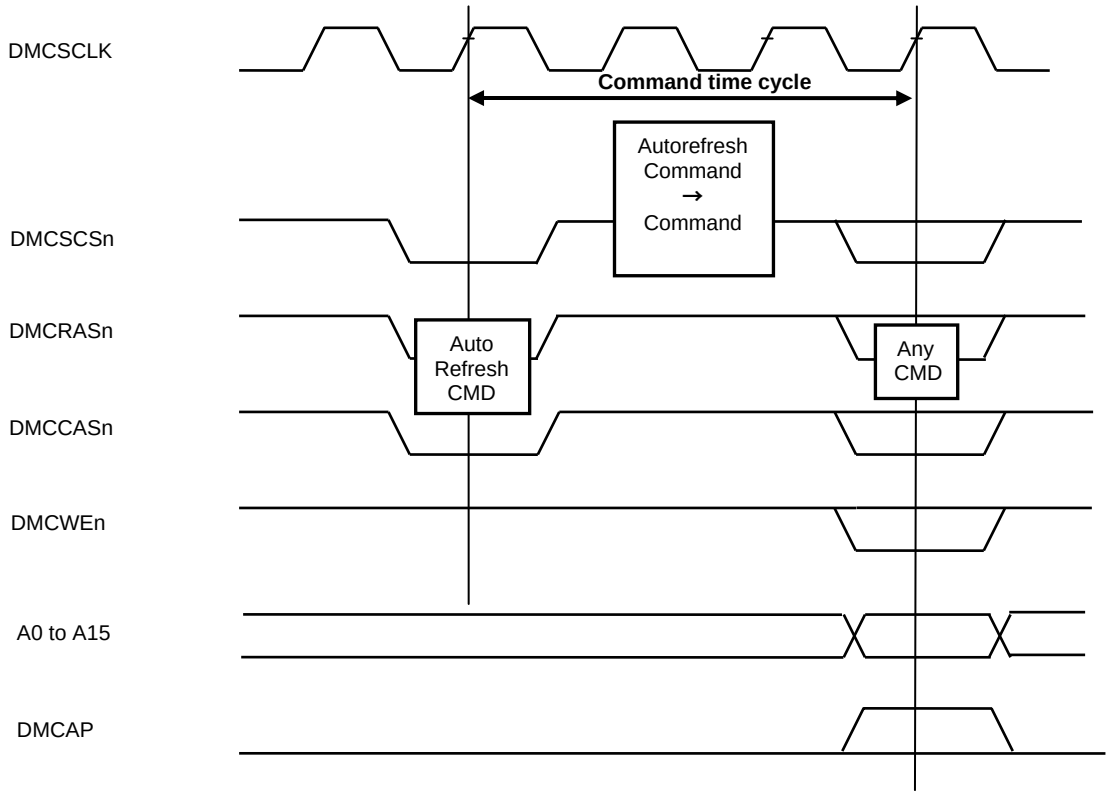


Figure 3.10.9 Time from Autorefresh Command to Other Command

13. dmc _t_rp_3 (DMC t_rp Register)

Address = (0xF430_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:3]	schedule_rp	R/W	0y011	Precharge delay setting to RAS. Set to (t_rp setting value -3).
[2:0]	t_rp	R/W	0y101	Set the time from Precharge to RAS (number of memory clocks): 0y000 to 0y111

[Description]

a. <schedule_rp>
Set the time from Precharge to RAS.
Set to (t_rp setting value -3).

b. <t_rp>
Set the time from Precharge to RAS (number of memory clocks):
0y000 to 0y111

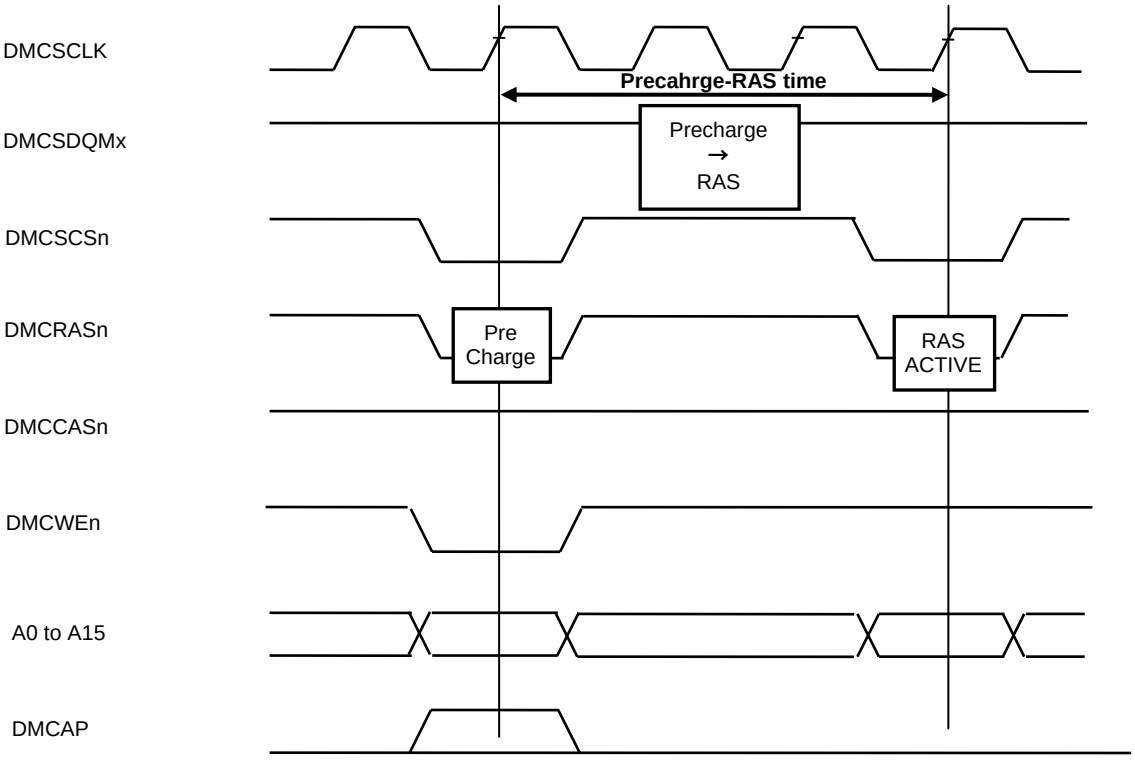


Figure 3.10.10 Time from Precharge to Other Command (including AutoRefresh)

14. dmc_t_rrd_3 (DMC t_rrd Register)

Address = (0xF430_0000) + (0x0034)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	t_rrd	R/W	0y0010	Delay time from Active bank A to Active bank B (Number of memory clocks): 0x0 to 0xF

[Description]

a. <t_rrd>

Delay time from Active bank A to Active bank B (Number of memory clocks):
0x0 to 0xF

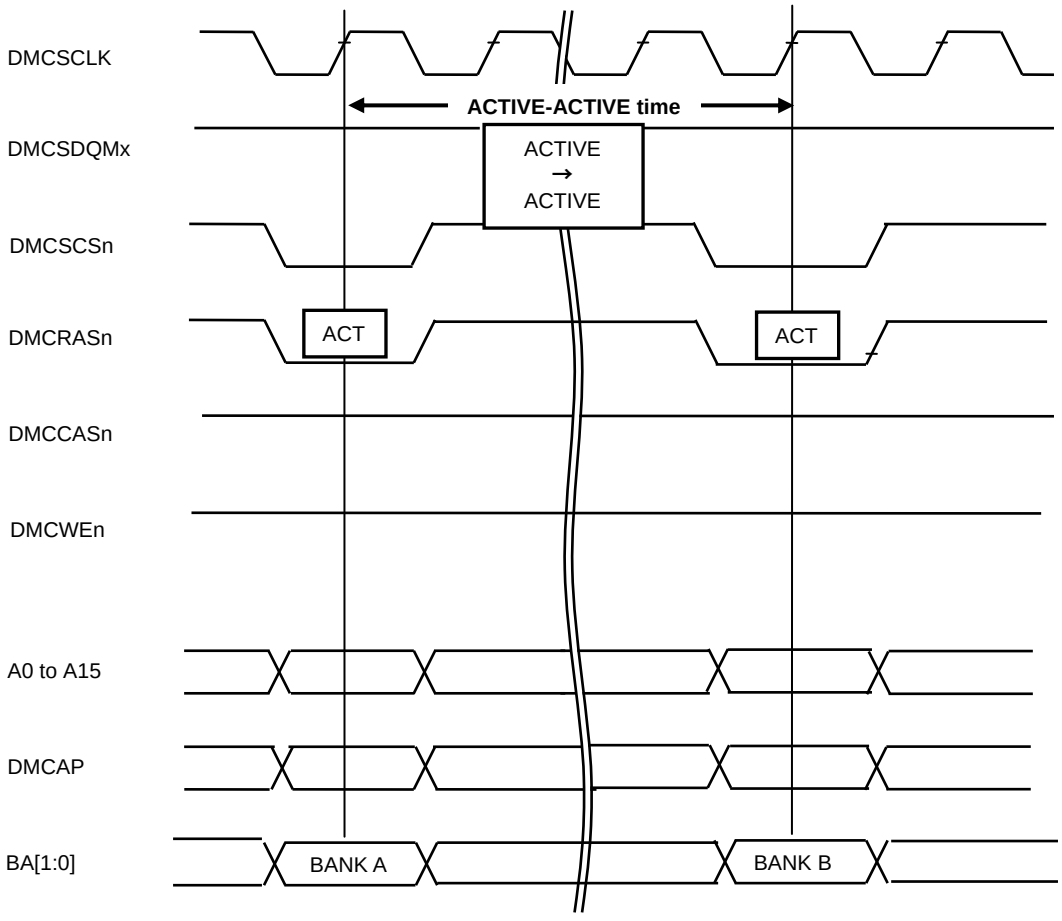


Figure 3.10.11 Time between Active bank A and other Active bank B

15. dmc_t_wr_3 (DMC t_wr Register)

Address = (0xF430_0000) + (0x0038)

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	t_wr	R/W	0y011	Delay from the last write data to Precharge (Number of memory clocks): 0y000 to 0y111

[Description]

a. <t_wr>

Delay from the last write data to Precharge (number of memory clocks).
Actual time (memory clocks): <t_wr> + 1.
When <t_wr> = 0y000, actual time (memory clocks) = 9 memory clocks.

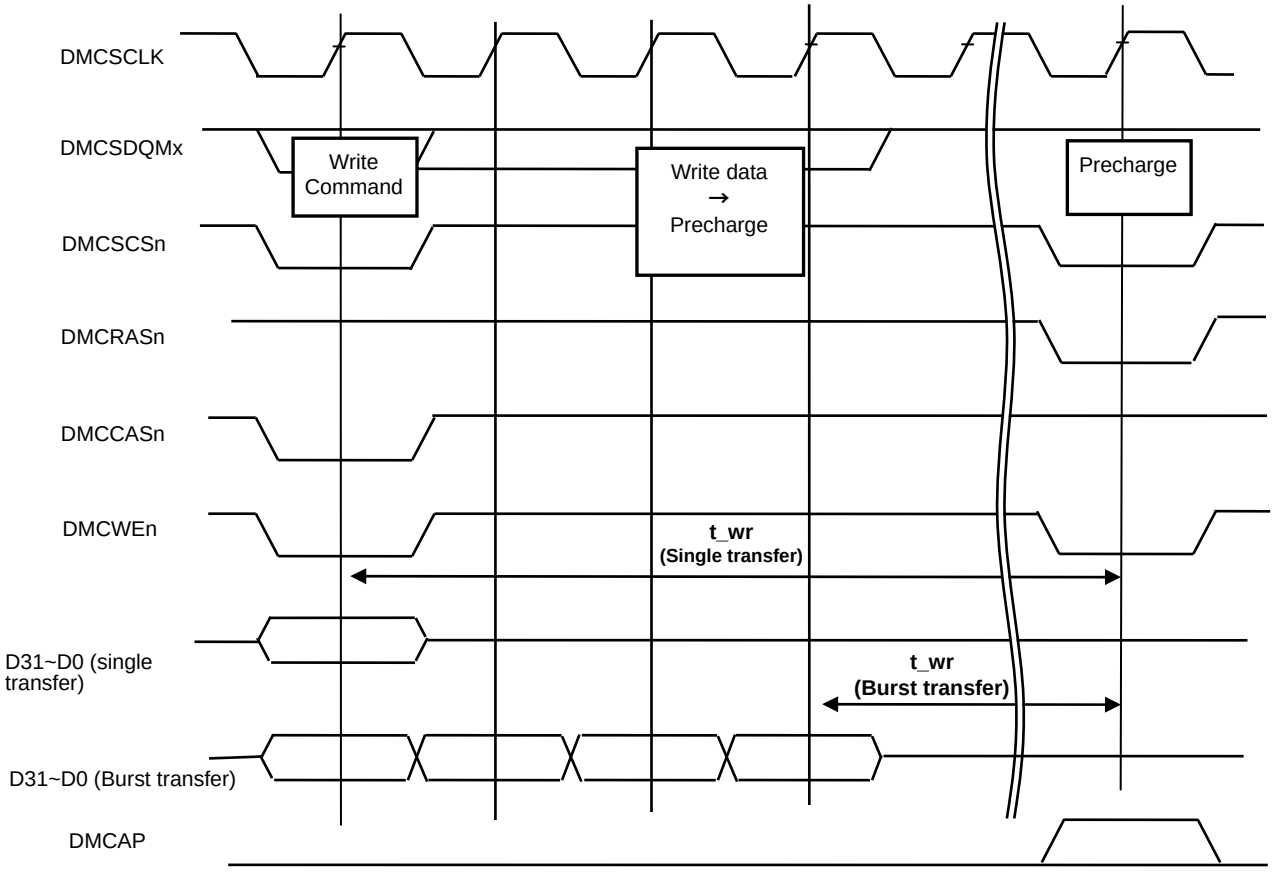


Figure 3.10.12 Time between Last Data of Write and Precharge

16. dmc_t_wtr_3 (DMC t_wtr Register)

Address = (0xF430_0000) + (0x003C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	t_wtr	R/W	0y010	Setting value from the last write data to read command (memory clocks) 0y000 to 0y111

[Description]

a. <t_wtr>

Delay from the last write data to read command (memory clocks).
When <t_wtr> = 0y000, actual time (memory clocks) = 8 memory clocks.

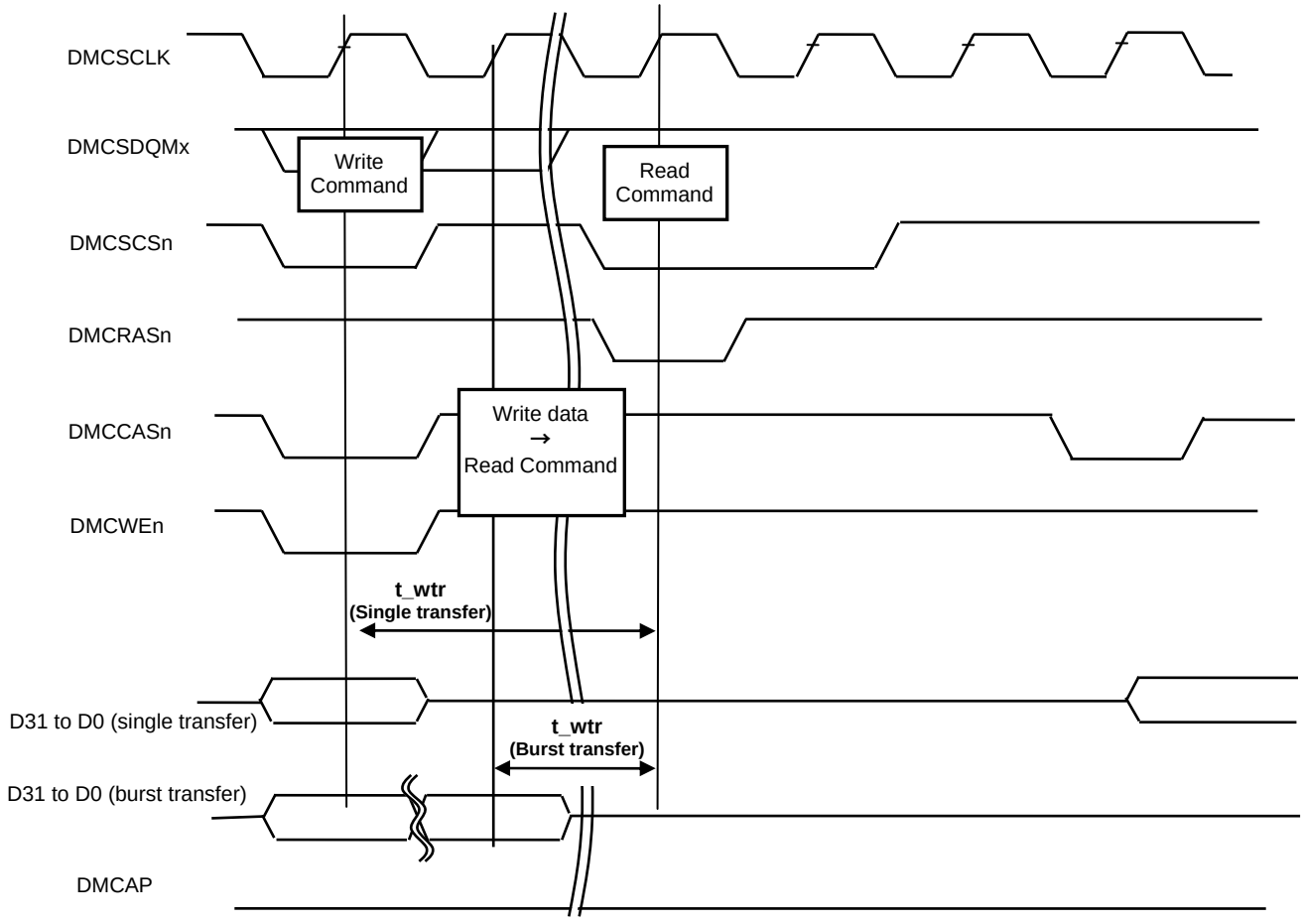


Figure 3.10.13 Time between Last data of write and Read command

17. dmc _t_xp_3 (DMC t_xp Register)

Address = (0xF430_0000) + (0x0040)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_xp	R/W	0x01	Set the exit power-down command time (Number of memory clocks) 0x00 to 0xFF

[Description]

a. <t_xp>

Time between Powerdown Exit command and other command (memory clocks)
Actual time (memory clocks): <t_xp> + 1

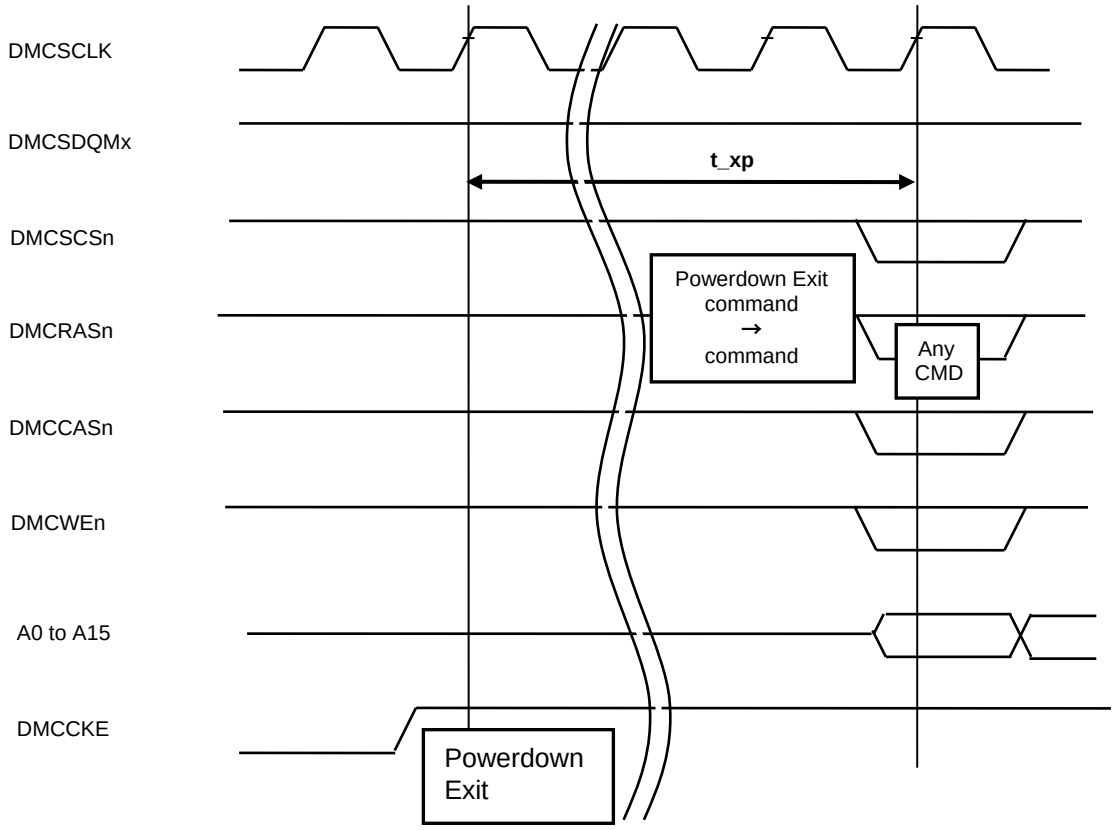


Figure 3.10.14 Time between Powerdown Exit Command and Other Command

18. dmc_ t_xsr_3 (DMC t_xsr Register)

Address = (0xF430_0000) + (0x0044)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_xsr	R/W	0x0A	Time from Self-refresh Exit command to other command (memory clocks) 0x00 to 0xFF

[Description]

a. <t_xp>
Time from Self-refresh Exit command to other command (memory clocks)

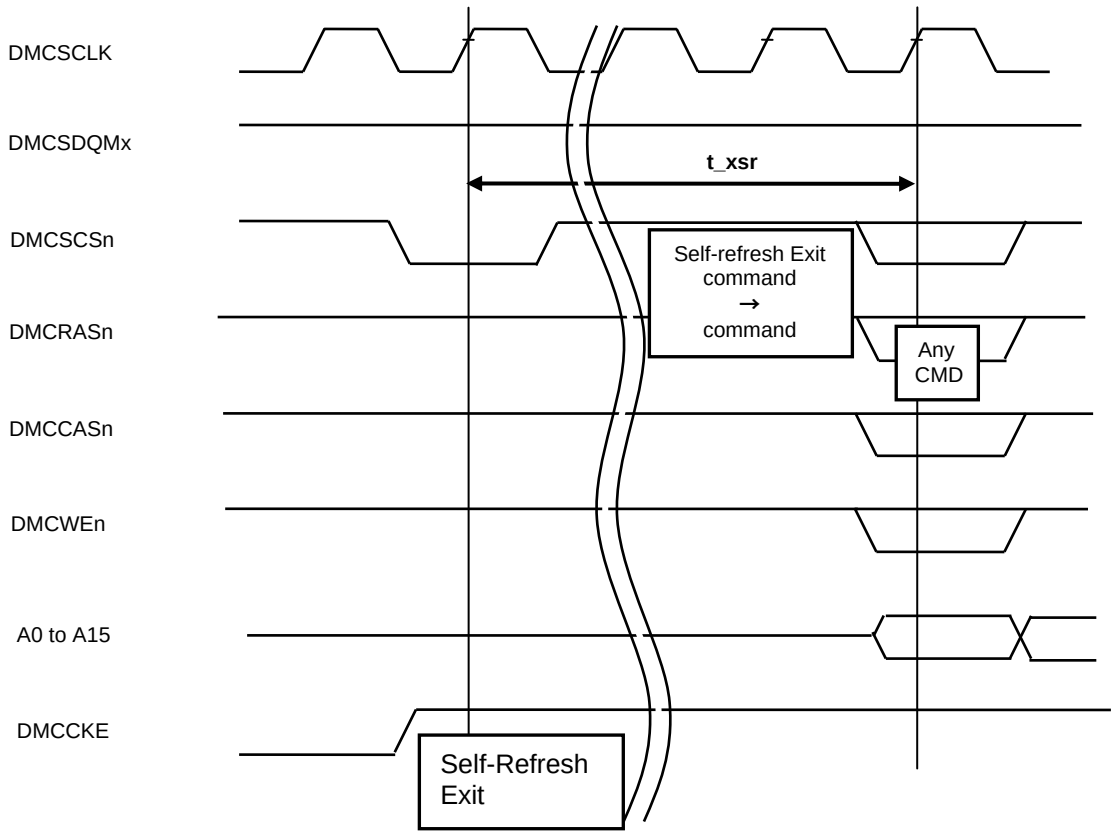


Figure 3.10.15 Time between Self-refresh Exit command and other command

19. dmc_t_esr_3 (DMC t_esr Register)

Address = (0xF430_0000) + (0x0048)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_esr	R/W	0x14	The minimum time from Self-refresh Entry to Exit: (memory clocks) 0x00 to 0xFF

Note: Self-refersh Exit have to use Wakeup direct command ,this register is only to set the the minimum time from Self-refresh Entry to Exit

[Description]

a. <t_esr>

The minimum time from Self-refresh Entry to Exit (memory clocks)
0x00 to 0xFF

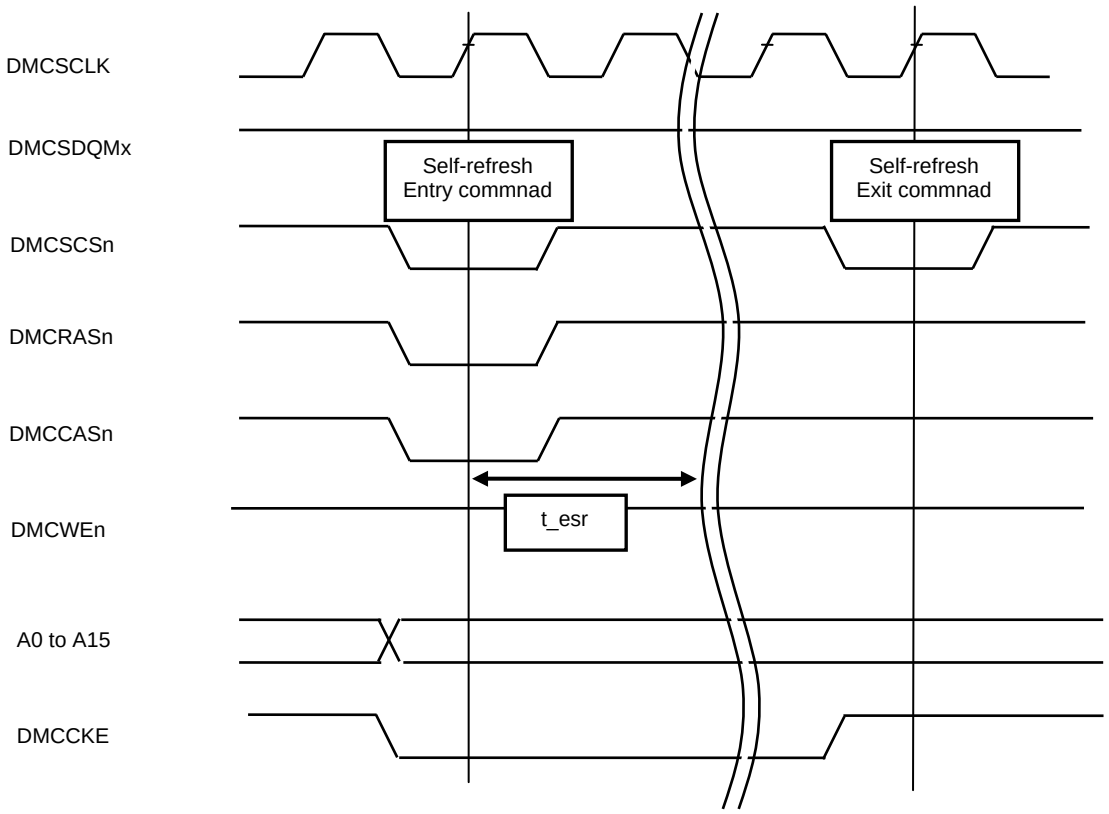


Figure3.10.16 Minimum execution time between Self-refresh Entry and Exit

20. dmc_id_<0-3>_cfg_3 Registers

Address = (0xF430_0000) + (0x0100)
Address = (0xF430_0000) + (0x0104)
Address = (0xF430_0000) + (0x0108)
Address = (0xF430_0000) + (0x010C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:2]	qos_max	R/W	0x00	maximum QoS: 0x00 ~ 0xFF
[1]	qos_min	R/W	0y0	minimum QoS selection: 0y0 = QoS max mode 0y1 = QoS min mode
[0]	qos_enable	R/W	0y0	Enable QoS 0y0 = Disable 0y1 = Enable

[Description]

QoS setting register list

Register	Address	Correspond to AHB
dmc_id_0_cfg_3	(0xF430_0000) + (0x0100)	AHB0 : CPU Data
dmc_id_1_cfg_3	(0xF430_0000) + (0x0104)	AHB1 : CPU Inst
dmc_id_2_cfg_3	(0xF430_0000) + (0x0108)	AHB2 : LCDC
dmc_id_3_cfg_3	(0xF430_0000) + (0x010C)	AHB3 : multilayer bus matrix2 (LCDDA,USB,DMAC1,DMAC2)

a. <qos_max>
QoS maximum value setting:
0x00 to 0xFF

b. <qos_min>
Minimum QoS selection:
0y0 = QoS max mode
0y1 = QoS min mode,
QoS minimum has priority over QoS maximum.

c. <qos_enable>
Enable QoS:
0y0 = Disable
0y1 = Enable

21. dmc_chip_0_cfg_3 (DMC chip_0_cfg Registers)

Address = (0xF430_0000) + (0x0200)

Bit	Bit Symbol	Type	Reset Value	Description
[31:17]	–	–	Undefined	Read as undefined. Write as zero.
[16]	brc_n_rbc	R/W	0y0	SDRAM address structure: 0y0 = row, bank, column 0y1 = bank, row, column
[15:8]	address_match	R/W	0xFF	Set the start address [31:24]: 0x00 to 0xFF
[7:0]	address_mask	R/W	0x00	Set the mask value of the start address [31:24]: The bit for the value 1 is a bit for address comparison 0x00 to 0xFF

[Description]

a. <brc_n_rbc>

SDRAM address structure:
0y0 = row, bank, column
0y1 = bank, row, column

b. <address_match>

Set the start address [31:24].
Do not access DMC area (Not used) except for configured CS area, if you accessed to memory less than 512 MB.

Note: When you set the start address, refer to the section 3.3 Memory Map, and confirm valid areas.

c. <address_mask>

Set the CS areas.
Determine which bit in the start address should be or should not be compared.
0y0 = Not compare
0y1 = Compare

Setting example

16MB CS area : <address_mask [7:0]> = 0y11111111
32MB CS area : <address_mask [7:0]> = 0y11111110
|
512MB CS area : <address_mask [7:0]> = 0y11100000
4GB CS area : <address_mask [7:0]> = 0y00000000

22. dmc_user_config_3 (DMC user_config Register)

Address = (0xF430_0000) + (0x0304)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	Reserved	–	Undefined	Read as undefined. Write as zero.
[6:4]	dmclk_out1	WO	0y000	SDR SDRAM constant value setting: must fix to 0y000
[3:1]	Reserved	–	Undefined	Read as undefined. Write as zero.
[0]	sdr_width	WO	0y0	Set the memory data bus width of corresponding external SDR memory: 0y0: 16-bit 0y1: Reserved

[Description]

a. <sdr_width>

Set the memory data bus width of corresponding external SDR memory:
0y0 = 16-bit
0y1 = Reserved

3.10.3.2 SMC (Static Memroy Controller)

This device contains SMC (Static Memory Controller) that controls the external memory (NOR Flash memory, Mask ROM SRAM and etc.).

(1) SMC function outline

Table 3.10.6 shows features of SMC.

Table 3.10.6 Features of SMC

	Features
Support memory	External asynchronous static memory (NOR Flash memory and SRAM, etc.) Support separate bus only
Data bus width	16bit data bus width
Access areas	2 areas supported by Chip select. Max acceess area: SMCCS0n: 512 MB SMCCS1n: 512 MB
Timing adjustment	Adjustable AC timing by register
	Support external wait request (only in Synchronous mode)
Clock	Selectable clock for external pin (f_{HCLK} or f_{HCLK} /2) by the clock controller register CLKCR5<SEL_SMC_MCLK>
External control pin	D15 to D0, A23 to A0, SMCBE0n, SMCBE1n, SMCCS0n, SMCCS1n,SMCOEn,SMCWEn

(2) SMC block diagram

Figure 3.10.17 is a SMC block diagram.

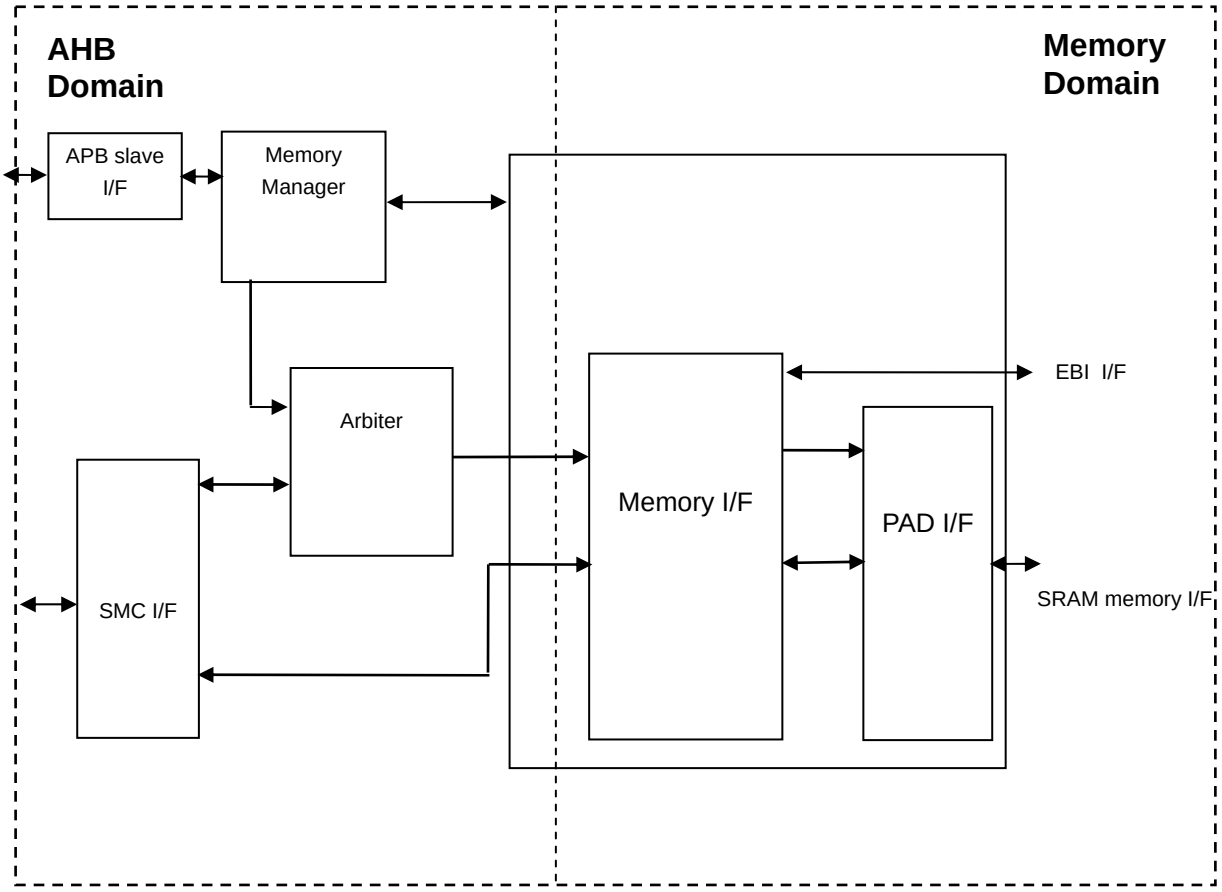


Figure 3.10.17 SMC block diagram

(a) Arbiter

The Arbiter receives accesses from the SMC I/F and memory manager. Read/Write requests are arbitrated on a Round-Robin basis. Requests from the manager have the highest priority.

(b) Memory manager

Updates timing registers and controls commands issued to memory

(3) SMC Function

(a) APB slave I/F

The APB slave I/F adds a wait state for all reads and writes

More than one wait state is generated in the following case:

Outstanding direct commands

A memory command is received, but the previous memory command has not been completed.

(b) Format

1. Hazard processing

When selfsame stand-alone bus master access to an external memory, the actual access procedure to memory is executed in the instruction order.

However, if multiple bus master access to an external memory, the read and write data will be stored temporary into independent buffer and be executing by priority circuit. Therefore, the read and the write instruction may switch execution sequence. So please coordinate a variety of sequences, e.g. making an enough time for next instruction, checking whether or not previous execution is finished, the common-use memory data uses the internal memory and so on.

2. Access to the SRAM memory

- Standard SRAM access
- Memory address shifting
- Memory burst alignment

The burst align settings are necessary in order to support asynchronous page mode memory. Refer to SMC register of MPMC, smc_set_opmode_3 (SMC Set Opmode Register).

Note: In case of not having any page mode methods, e.g. NOR Flash, it is unnecessary to set burst align.

Memory burst length: Supported memory burst transfer length is 4 beats.

(c) Memory manager operation

The memory manager controls the SMC state and manages update of chip configuration registers.

(d) Memory I/F operation

The memory I/F issues commands and controls their timings.

Table 3.10.7 Static Memory Setup Example

Base address = 0xF430_1000

Register address	Write data	Description
0x0014	0x00029266	smc_set_cycles_3
0x0018	0x00000809	smc_set_opmode_3
0x0010	0x00400000	smc_direct_cmd_3

(4) SMC Registers for MPMC0

Table 3.10.8 MPMC0 SMC SFR list

Base address = 0xF430_1000

Register Name	Address (base+)	Type	Reset value	Description
Reserved	0x0000	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0004	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0008	–	–	Write prohibited
Reserved	0x000C	–	–	Write prohibited
smc_direct_cmd_3	0x0010	WO	–	SMC Direct Command Register
smc_set_cycles_3	0x0014	WO	–	SMC Set Cycles Register
smc_set_opmode_3	0x0018	WO	–	SMC Set Opmode Register
Reserved	0x0020	–	Undefined	Read as undefined. Write as zero.
smc_sram_cycles0_0_3	0x0100	RO	0x0002B3CC	SMC SRAM Cycles Registers <0-1>
smc_sram_cycles0_1_3	0x0120			
Reserved	0x0140	RO	Undefined	Read as undefined.
Reserved	0x0160	RO	Undefined	Read as undefined.
smc_opmode0_0_3	0x0104	RO	0x20E00802	SMC Opmode Registers <0-1>
smc_opmode0_1_3	0x0124		0x60E00802	
Reserved	0x0144	RO	Undefined	Read as undefined.
Reserved	0x0164	RO	Undefined	Read as undefined.
Reserved	0x0200	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0204	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E00	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E04	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E08	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FE0-0x0FEC	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FF0-0x0FFC	–	Undefined	Read as undefined. Write as zero.

Note: The APB supports only single-word 32-bit accesses. Read from or write to registers at single-word 32-bit mode.

1. smc_direct_cmd_3 (SMC Direct Command Register)

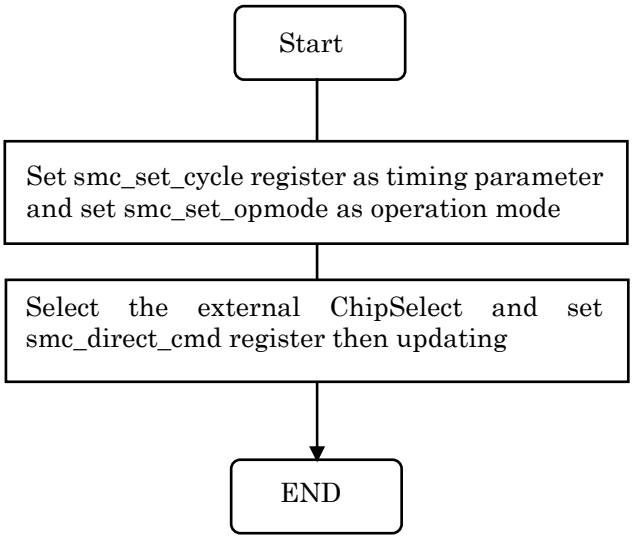
Address = (0xF430_1000) + (0x0010)

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

[Description]

a. <chip_select>
CS selection
0y000 = CS0
0y001 = CS1
0y010 to 0y111 = Reserved

b. <cmd_type>
Current command:
0y00 = Reserved
0y01 = Reserved
0y10 = UpdateRegs
0y11 = Reserved


Start
Set smc_set_cycle register as timing parameter and set smc_set_opmode as operation mode
Select the external ChipSelect and set smc_direct_cmd register then updating
END

2. smc_set_cycles_3 (SMC Set Cycles Register)

Address = (0xF430_1000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:23]	–	–	Undefined	Read as undefined. Write as zero.
[22:20]	Reserved	–	Undefined	Read as undefined. Write as zero.
[19:17]	Set_t5	WO	–	Set value of t_{TR} (holding register) 0y000 to 0y111
[16:14]	Set_t4	WO	–	Set value of t_{PC} (holding register) 0y000 to 0y111
[13:11]	Set_t3	WO	–	Set value of t_{WP} (holding register) 0y000 to 0y111
[10:8]	Set_t2	WO	–	Set value of t_{CEOE} (holding register) 0y000 to 0y111
[7:4]	Set_t1	WO	–	Set value of t_{WC} (holding register) 0y0000 to 0y1111
[3:0]	Set_t0	WO	–	Set value of t_{RC} (holding register) 0y0000 to 0y1111

This register is provided to adjust the access cycle of static memory and should be set to satisfy the A.C. specifications of the memory to be used, the access cycle is determined to satisfy the settings of both this register and the external wait signal.

Note that the external wait signal is only effective in synchronous mode. It cannot be used in asynchronous mode.

This is a holding register for enabling setting values. By executing of the following operations, the settings values of this register will be updated to the configuration register of the memory manager and enabled.

・The smc_direct_cmd Register indicates only a register update is taking place.

[Description]

a. <Set_t5>

Set value of t_{TR} (holding register).

0y000 to 0y111

b. <Set_t4>

Set value of t_{PC} (holding register).

0y000 to 0y111

c. <Set_t3>

Set value of t_{WP} (holding register).

0y000 to 0y111

d. <Set_t2>

Set value of t_{CEOE} (holding register).

0y000 to 0y111

e. <Set_t1>

Set value of t_{WC} (holding register).

0y0000 to 0y1111

f. <Set_t0>

Set value of t_{RC} (holding register).

0y0000 to 0y1111

Example of setting timing

Setting Example: SMC Set Cycles Register = 0x0002B1C3

Register setting value	t_{TR}	t_{PC}	t_{WP}	t_{CEOE}	T_{WC}	t_{RC}
0x0002B1C3	−	−	−	1	−	3

− : don't care

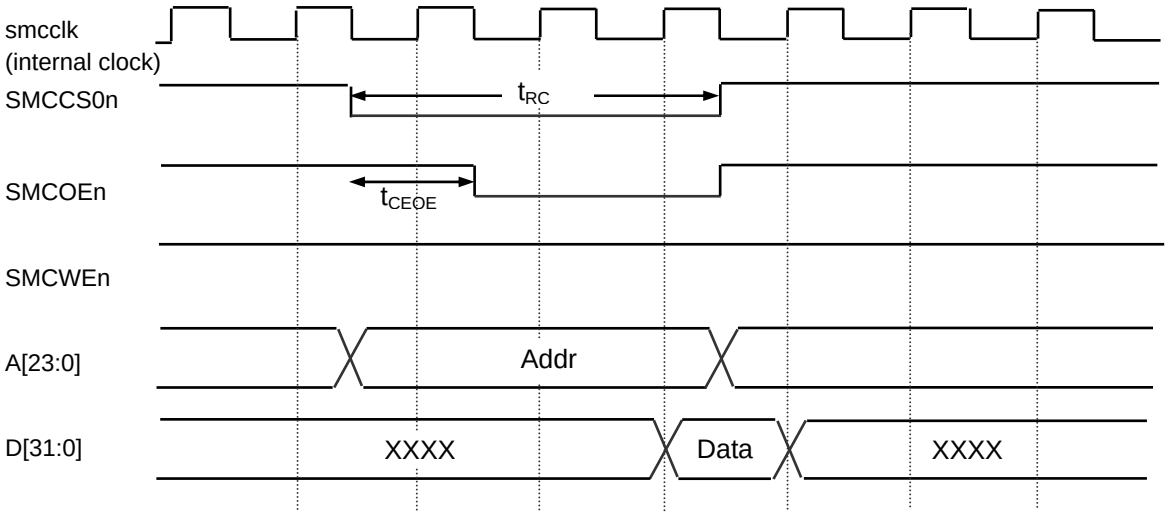


Figure 3.10.18 Asynchronous Read

Setting Example: SMC Set Cycles Register = 0x0002934C

Register setting value	tTR	tPC	tWP	tCEOE	TWC	tRC
0x0002934C	–	–	2	–	4	–

– : don't care

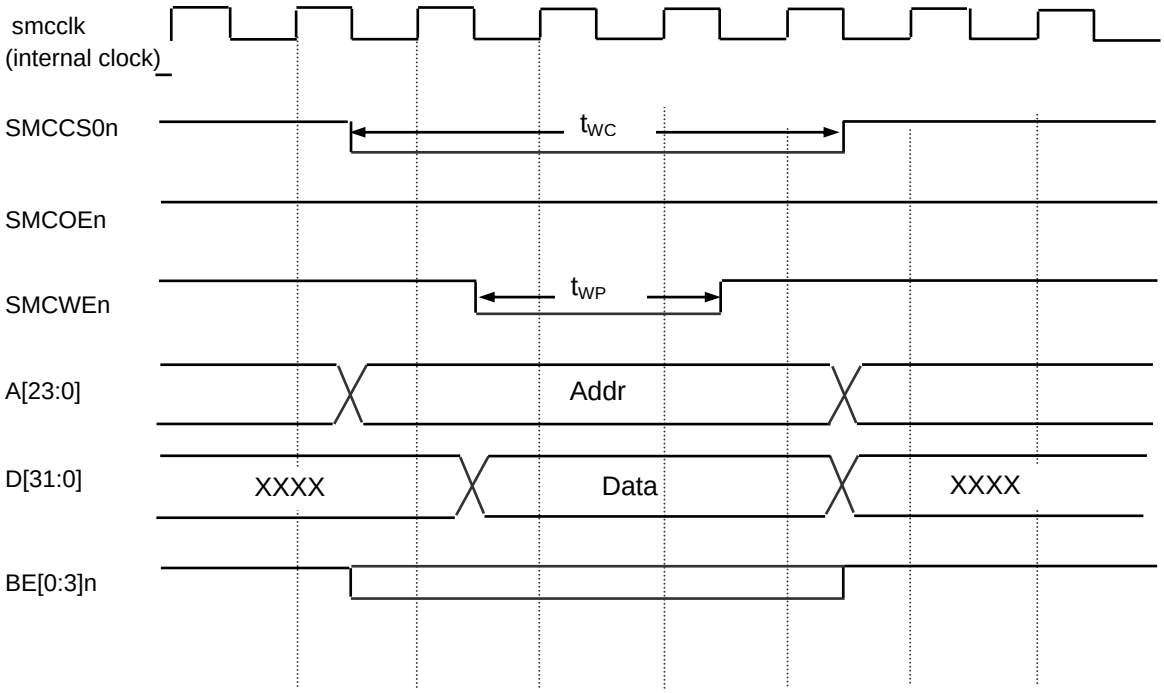


Figure 3.10.19 Asynchronous Write

Setting Example: SMC Set Cycles Register = 0x000272C3

Register setting value	t_{TR}	t_{PC}	t_{WP}	t_{CEOE}	T_{WC}	t_{RC}
0x000272C3	–	1	–	2	–	3

– : don't care

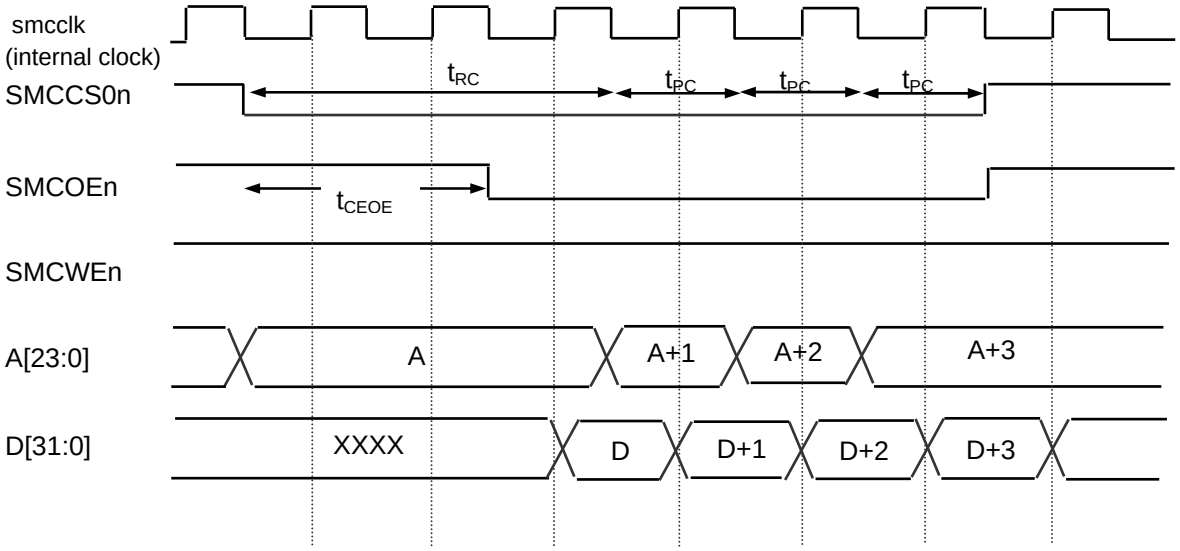


Figure 3.10.20 Asynchronous Page Read

Setting Example: SMC Set Cycles Register = 0x00029143

Register setting value	t_{TR}	t_{PC}	t_{WP}	t_{CEOE}	T_{WC}	t_{RC}
0x00029143	1	−	2	1	4	3

− : don't care

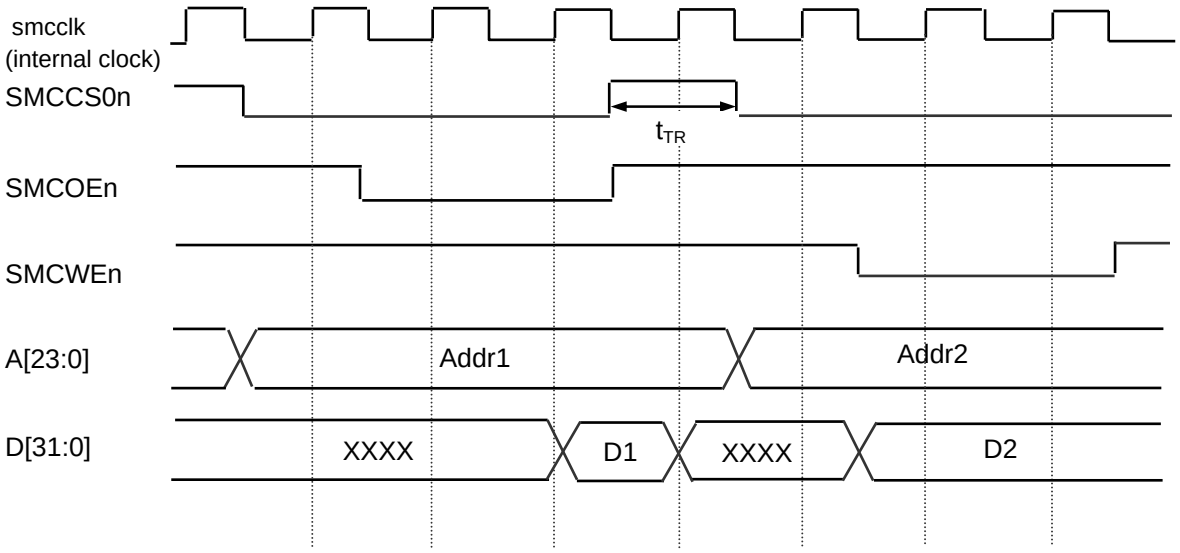


Figure 3.10.21 Asynchronous Write after Asynchronous Read

3. smc_set_opmode_3 (SMC Set Opmode Register)

Address = (0xF430_1000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:13]	set_burst_align	WO	–	Memory burst boundary split setting: (holding register) 0y000 = bursts can cross any address boundary 0y001 = split at the 32-beat burst boundary 0y010 = split at the 64-beat burst boundary 0y011 = split at the 128-beat burst boundary 0y100 = split at the 256-beat burst boundary other = Reserved
[12]	set_bls	WO	–	Byte Enable (SMCBE0-3)bls timing setting: 0y0 = SMCCSn timing 0y1 = SMCWEn timing
[11]	Reserved	WO	–	Write as zero.
[10]	-	–	Undefined	Read as undefined. Write as zero.
[9:7]	set_wr_bl	WO	–	Write burst length 0y000 = 1 beat 0y001 = 4 beats other = Reserved
[6]	set_wr_sync	WO	–	Write synchronization mode setting 0y0 = asynchronous write mode 0y1 = Reserved
[5:3]	set_rd_bl	WO	–	Read burst length 0y000 = 1 beat 0y001 = 4 beats other = Reserved
[2]	set_rd_sync	WO	–	Read synchronization mode setting: 0y0 = asynchronous read mode 0y1 = Reserved
[1:0]	set_mw	WO	–	Holding register of the memory data bus width set value: 0y00 = reserved 0y01 = 16 bits 0y10 = reserved 0y11 = Reserved

This is a holding register for enabling setting values. By executing of the following operations, the settings values of this register will be updated to the configuration register of the memory manager and enabled.

・The smc_direct_cmd Register takes place the UpdateRegs.

[Description]

a. < set_burst_align >

Memory burst boundary split setting:
0y000 = bursts can cross any address boundary
0y001 = split at the 32-beat burst boundary
0y010 = split at the 64-beat burst boundary
0y011 = split at the 128-beat burst boundary
0y100 = split at the 256-beat burst boundary
other = Reserved

b. < set_bls >

Byte Enable (SMCBE0-3) timing setting:
0y0 = SMCCSn timing
0y1 = SMCWEn timing

c. < set_wr_bl >

Write burst length
0y000 = 1 beat
0y001 = 4 beats
other = Reserved

d. < set_wr_sync >

Write synchronization mode setting:
0y0 = asynchronous write mode
0y1 = synchronous write mode

e. < set_rd_bl >

Read burst length
0y000 = 1 beat
0y001 = 4 beats
other = Reserved

f. < set_rd_sync >

Read synchronization mode setting:
0y0 = asynchronous read mode
0y1 = synchronous read mode

g. < set_mw >

Holding register of the memory data bus width set value:
0y00 = Reserved
0y01 = 16 bits
0y10 = Reserved
0y11 = Reserved

4. smc_sram_cycles0_0_3 (SMC SRAM Cycles Registers 0 <0>)

Address = (0xF430_1000) + (0x0100)

Bit	Bit Symbol	Type	Reset Value	Description
[31:20]	–	–	Undefined	Read as undefined.
[19:17]	t_tr	RO	0y001	Turnaround time for SRAM chip configuration 0y000 to 0y111
[16:14]	t_pc	RO	0y010	page cycle time: 0y000 to 0y111
[13:11]	t_wp	RO	0y110	delay time for smc_we_n_0: 0y000 to 0y111
[10:8]	t_ceoe	RO	0y011	delay time for smc_oe_n_0: 0y000 to 0y111
[7:4]	t_wc	RO	0y1100	write cycle time: 0y0000 to 0y1111
[3:0]	t_rc	RO	0y1100	read cycle time: 0y0000 to 0y1111

[Description]

a. <t_tr>
Turnaround time for SRAM chip configuration:
0y000 to 0y111

b. <t_ pc>
Page cycle time:
0y000 to 0y111

c. <t_wp>
Delay time for s smc_we_n_0:
0y000 to 0y111

d. <t_ceoe>
Delay time for smc_oe_n_0:
0y000 to 0y111

e. <t_wc>
Write cycle time:
0y0000 to 0y1111

f. <t_rc>
Read cycle time:
0y0000 to 0y1111

- smc_sram_cycles0_x_3 (SMC SRAM Cycles Registers 0 <x>) (x = 0 to 1)

  The structure and description of these registers are same as smc_sram_cycles0_0_3. Please refer to the description of smc_sram_cycles0_0_3.
  The name and address of these registers, please refer to Table 3.10.8 MPMC0 SMC SFR list.

5. smc_opmode0_0_3 (SMC Opmode Registers 0<0>)

Address = (0xF430_1000) + (0x0104)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	Reserved	RO	0x20	Read as 0x20.
[23:16]	Reserved	RO	0xE0	Read as 0xE0.
[15:13]	burst_align	RO	0y000	Memory burst boundary split set value: 0y000 = bursts can cross any address boundary 0y001 = split at the 32-beat burst boundary 0y010 = split at the 64-beat burst boundary 0y011 = split at the 128-beat burst boundary 0y100 = split at the 256-beat burst boundary other = Reserved
[12]	bls	RO	0y0	bls timing : 0y0 = chip select 0y1 = Reserved
[11]	Reserved	RO	0y1	–
[10]	–	–	Undefined	Read as undefined.
[9:7]	wr_bl	RO	0y000	Write memory burst length: 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[6]	wr_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous write operation 0y1 = Reserved.
[5:3]	rd_bl	RO	0y000	Read memory burst length: 0y000 = 1 beat 0y001 = 4 beats other = Reserved
[2]	rd_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous read operation 0y1 = Reserved.
[1:0]	mw	RO	0y10	Memory data bus width : 0y00 = Reserved 0y01 = 16-bits 0y10 = Reserved 0y11 = Reserved

6. smc_opmode0_1_3 (SMC Opmode Registers 0<1>)

Address = (0xF430_1000) + (0x0124)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	Reserved	RO	0x60	Read as 0x60.
[23:16]	Reserved	RO	0xE0	Read as 0xE0.
[15:13]	burst_align	RO	0y000	Memory burst boundary split set value: 0y000 = bursts can cross any address boundary 0y001 = split at the 32-beat burst boundary 0y010 = split at the 64-beat burst boundary 0y011 = split at the 128-beat burst boundary 0y100 = split at the 256-beat burst boundary other = Reserved
[12]	bls	RO	0y0	bls timing : 0y0 = chip select 0y1 = Reserved
[11]	Reserved	RO	0y1	–
[10]	–	–	Undefined	Read as undefined.
[9:7]	wr_bl	RO	0y000	Write memory burst length: 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[6]	wr_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous write operation 0y1 = Reserved.
[5:3]	rd_bl	RO	0y000	Read memory burst length: 0y000 = 1 beat 0y001 = 4 beats other = Reserved
[2]	rd_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous read operation 0y1 = Reserved.
[1:0]	mw	RO	0y10	Memory data bus width : 0y00 = Reserved 0y01 = 16-bits 0y10 = Reserved 0y11 = Reserved

[Description]

a. <burst_align>

Memory burst boundary split set value:
0y000 = bursts can cross any address boundary
0y001 = split at the 32-beat burst boundary
0y010 = split at the 64-beat burst boundary
0y011 = split at the 128-beat burst boundary
0y100 = split at the 256-beat burst boundary
Other = Reserved

b. <bls>

It shows the timing of bls (byte-lane strobe) output.
0y0 = chip select
0y1 = Reserved

c. <wr_bl>

Write memory burst length:
0y000 = 1-beat
0y001 = 4-beats
Other = Reserved

d. <wr_sync>

Memory operation mode:
0y0 = asynchronous write operation
0y1 = Reserved.

e. <rd_bl>

Read memory burst length:
0y000 = 1-beat
0y001 = 4-beats
Other = Reserve

f. <rd_sync>

Memory operation mode:
0y0 = asynchronous read operation
0y1 = Reserved.

g. <mw>

The Reset value depends on setting state. The CS0 memory data bus width can be set for Boot.:

3.10.4 Overview of MPMC1

MPMC1 contains both a DMC (Dynamic Memory Controller) that controls SDRAM and SMC (Static Memory Controller) that controls NOR Flash and SRAM.

Features of a DMC (Dynamic Memory Controller):

a. Supports 16-bit DDR SDRAM(only supports LVCMOS type memory I/O power)
b. Supports 1 channel Chip Select signal
c. Supports adjusting function in each clock for SDRAM each timing.

Features of an SMC (Static Memory Controller):

a. Supports asynchronous, 16-bit SRAM and NOR Flash (only separate buses are supported, and multiplex buses are not supported)
b. Supports 2 channels Chip Select signals
c. Cycle timings and memory data bus widths can be programmed for each Chip Select signal

3.10.5 Function of MPMC 1

Figure 3.10.22 is a simplified block diagram of MPMC1 circuits.

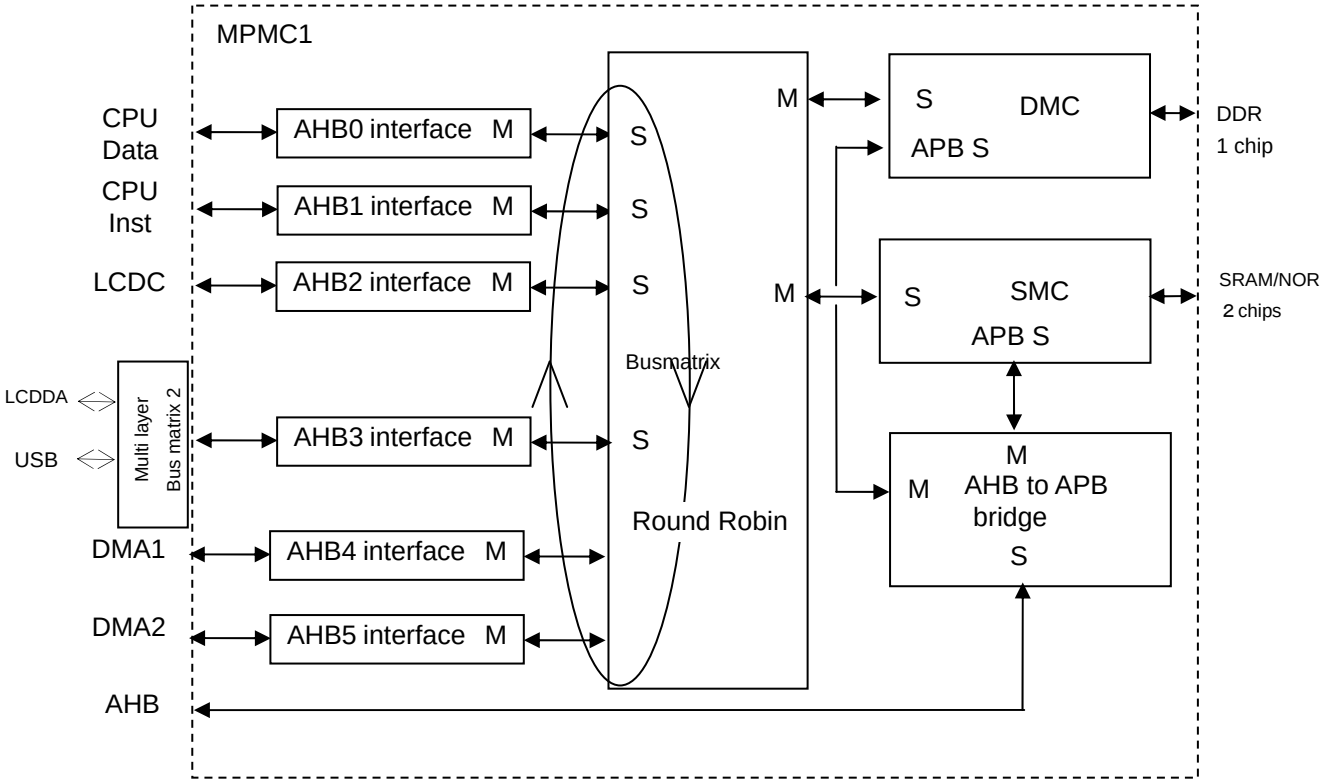
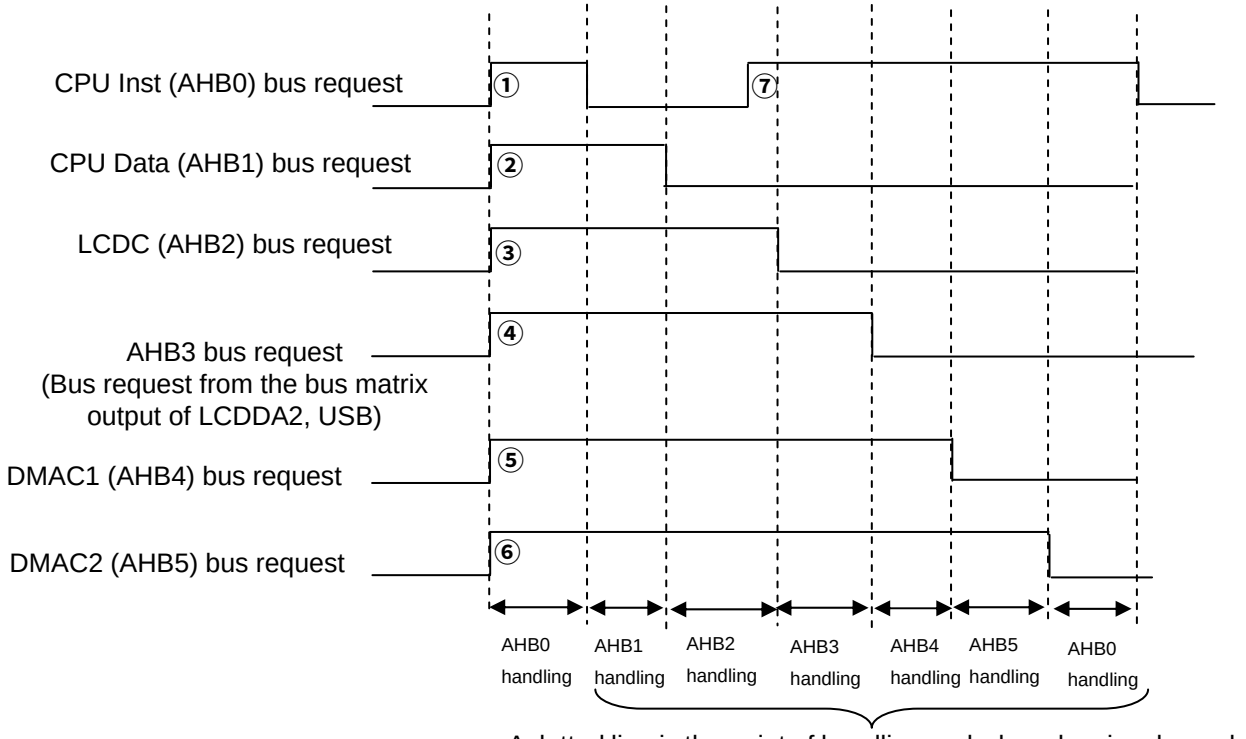


Figure 3.10.22 MPMC1block diagram

(a) Bus matrix

1. Bus matrix of AHB0, AHB1, AHB2, AHB3, AHB4 and AHB5 supports Round-Robin arbitration scheme.

The following diagram shows the priority of bus requests.



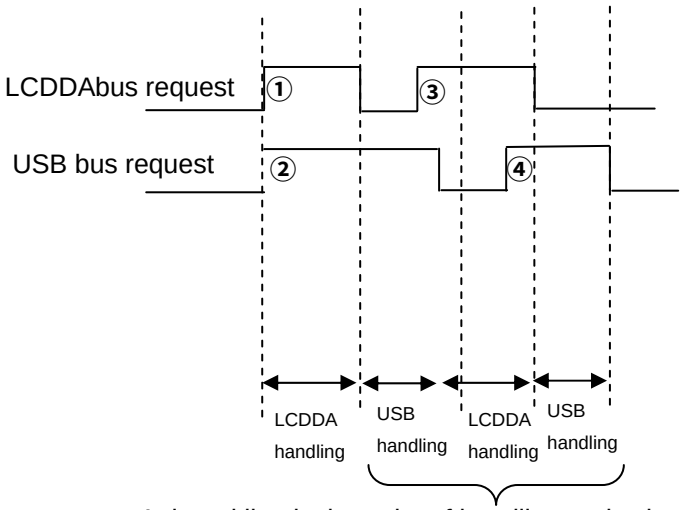
Priority of handling :① → ② → ③ → ④→⑤→⑥→⑦

2. Bus matrix 2 of LCDDA and USB handles the earliest bus request first. If multiple bus requests are accepted simultaneously, they are handled according to hardware priority.

   Hardware priority is shown following

   Hardware priority is shown following

   LCDDA (high) C1 → USB (low)



A dotted line is the point of handling end, where bus is released.

Handling priority : ① → ② → ③ →④

(b) Clock Variety

Control clock is controlled in PLLCG circuit:

1. Dynamic memory clock: Use HCLK clock
2. Static memory clock: Use HCLK or 1/2 HCLK

3.10.5.1 DMC (Dynamic Memory Controller)

(1) DMC block diagram

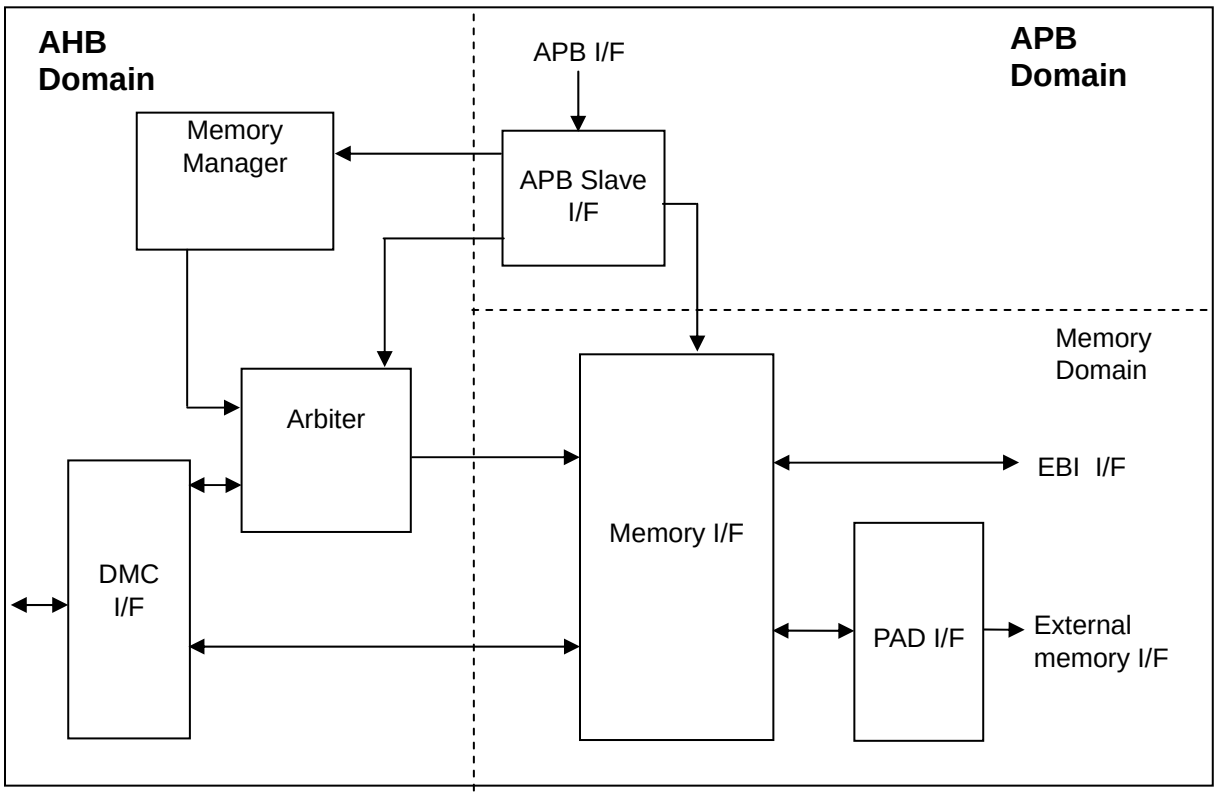


Figure 3.10.23 DMC block diagram

(a) Arbiter

The arbiter receives access commands from the DMC I/F and the memory manager, and after access arbitration, it passes the highest priority command to the memory I/F.

Data is read from the memory I/F to the DMC I/F.

(b) Memory manager

The memory manager monitors and controls the DMC current.

(2) DMC Function operation

(a) Arbiter operation

1. read/write access arbitration
2. For read accesses, QoS (Quality of Service) is provided.
3. Hazard processing

   When selfsame stand-alone bus master access to an external memory, the actual access procedure to memory is executed in the instruction order.

   However, if multiple bus master access to an external memory, the read and write data will be stored temporary into independent buffer and be executing by priority circuit. Therefore, the read and the write instruction may switch execution sequence. So please coordinate a variety of sequences, e.g. making an enough time for next instruction, checking whether or not previous execution is finished, the common-use memory data uses the internal memory and so on.

4. Monitoring the state machine and select an entry of the proper pipeline.

(b) Memory manager operation

1. Monitor and control DMC circuit
2. Issuing direct comand
   - NOP
   - PRECHARGEALL
   - AUTOREFRESH
   - MODEREG
   - EXTENDED MODEREG
3. Auto Refresh function is provided

   Set Auto Refresh timing by 15bit counter.

(c) Memory interface operation

According to use, there are three kinds of built-in FIFOs.

1. command FIFO: 2 words
2. read data FIFO:10 words
3. write data FIFO:10 words

- As the FIFO sizes of either read or write FIFO is 10 words.

  For one transfer, the max size is 8 words.

(d) Low Power function

DMC provide 2 kinds of Low Power modes.

1. Set dmc_memc_cmd_3 register to realize Low_power (Self Refresh Mode).
2. Set dmc_memory_cfg_3 register, stop memory clock (DMCCLK) or as no memory access, CKE is set to invalid (CKE = low).

Note: Clock Suspend Mode function and Power Down mode cannot be used concurrently.

(e) QoS Function

The QoS function is available in read-accessing only.

The QoS function is the service function for exception handling at Round-Robin which is controlled by Bus matrix for MPMC. dmc_id_x_cfg_5<qos_max> is set by a register within the DMC on a port by port basis. dmc_id_x_cfg_5<qos_max> indicates a required read maximum latency. A QoS_max timeout causes the transaction to be raised to a higher priority.

You can also set dmc_id_x_cfg_5<qos_min> to enable for a specific port so that its transfers are serviced with a higher priority. This impacts the overall memory band width because it limits the options of the scheduling algorithm.

Table 3.10.9 Example DDR memory setup

Base address = 0xF431_0000

Register address	Write data	Description
0x0014	0x00000004	Set cas_latency to 2
0x0018	0x00000001	Set t_dqss to 1
0x001C	0x00000002	Sett_mrd to 2
0x0020	0x00000007	Sett_ras to 7
0x0024	0x0000000B	Set t_rc to 11
0x0028	0x00000015	Set t_rcd to 5 and schedule_rcdto 2
0x002C	0x000001F2	Set t_rfc to 18 and schedule_rfcto 15
0x0030	0x00000015	Set t_rp to 5 and schedule_rpto 2
0x0034	0x00000002	Set t_rrd to 2
0x0038	0x00000003	Set t_wrto 3
0x003C	0x00000002	Set t_wtr to 2
0x0040	0x00000001	Set t_xp to 1
0x0044	0x0000000A	Set t_xsr to 10
0x0048	0x00000014	Set t_esr to 20
0x000C	0x00010009	Set memory configuration
0x0010	0x00000640	Set auto refresh time to be every 1600 DMCSCLK periods
0x0200	0x000000FF	Set chip select for chip 0 to be 0x00XXXXXX, rbc configuration
0x0008	0x000C0000	Carry out chip 0 Nopcommand
0x0008	0x00000000	Carry out chip 0 Prechargeall command
0x0008	0x00090000	Extended mode register setup
0x0008	0x00080122	Mode register setup
0x0008	0x00000000	Precharge all
0x0008	0x00040000	Carry out chip 0 Autorefresh command
0x0008	0x00040000	Carry out chip 0 Autorefresh command
0x0008	0x00080032	Carry out chip 0 Mode Reg command 0x32 mapped to low add bits
0x0004	0x00000000	Change DMC state to ready

(3) MPMC1 DMC register

Table 3.10.10 SFR list

Base address = 0xF431_0000

Register Name	Address (base+)	Type	Reset value	Description
dmc_memc_status_5	0x0000	RO	0x00000390	DMC Memory Controller Status Register
dmc_memc_cmd_5	0x0004	WO	–	DMC Memory Controller Command Register
dmc_direct_cmd_5	0x0008	WO	–	DMC Direct Command Register
dmc_memory_cfg_5	0x000C	R/W	0x00010020	DMC Memory Configuration Register
dmc_refresh_prd_5	0x0010	R/W	0x00000A60	DMC Refresh Period Register
dmc_cas_latency_5	0x0014	R/W	0x00000006	DMC CAS Latency Register
dmc_t_dqss_5	0x0018	R/W	0x00000001	DMC t_dqss Register
dmc_t_mrd_5	0x001C	R/W	0x00000002	DMC t_mrd Register
dmc_t_ras_5	0x0020	R/W	0x00000007	DMC t_ras Register
dmc_t_rc_5	0x0024	R/W	0x0000000B	DMC t_rc Register
dmc_t_rcd_5	0x0028	R/W	0x0000001D	DMC t_rcd Register
dmc_t_rfc_5	0x002C	R/W	0x00000212	DMC t_rfc Register
dmc_t_rp_5	0x0030	R/W	0x0000001D	DMC t_rp Register
dmc_t_rrd_5	0x0034	R/W	0x00000002	DMC t_rrd Register
dmc_t_wr_5	0x0038	R/W	0x00000003	DMC t_wr Register
dmc_t_wtr_5	0x003C	R/W	0x00000002	DMC t_wtr Register
dmc_t_xp_5	0x0040	R/W	0x00000001	DMC t_xp Register
dmc_t_xsr_5	0x0044	R/W	0x0000000A	DMC t_xsr Register
dmc_t_esr_5	0x0048	R/W	0x00000014	DMC t_esr Register
dmc_id_0_cfg_5 dmc_id_1_cfg_5 dmc_id_2_cfg_5 dmc_id_3_cfg_5 dmc_id_4_cfg_5 dmc_id_5_cfg_5	0x0100 0x0104 0x0108 0x010C 0x0110 0x0114	R/W	0x00000000	DMC id_<0-5>_cfg Registers
dmc_chip_0_cfg_5	0x0200	R/W	0x0000FF00	DMC chip_0_cfg Registers
Reserved	0x0204	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0208	–	Undefined	Read as undefined. Write as zero.
Reserved	0x020C	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0300	–	Undefined	Read as undefined. Write as zero.
dmc_user_config_5	0x0304	WO	Undefined	DMC user_config Register
Reserved	0x0E00	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E04	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E08	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FE0-0x0FEC	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FF0-0x0FFC	–	Undefined	Read as undefined. Write as zero.

Note: The APB supports only single-word 32-bit accesses. Read from or write to registers at single-word 32-bit mode.

MPMC1

The permission status of Register Read/Write access (dmc_memc_status_5 status)

○: permitted ×: prohibited

Register Name	Type	Read				Write			
		dmc_memc_status_5				dmc_memc_status_5			
		Config	Ready	Paused	Low_power	Config	Ready	Paused	Low_power
dmc_memc_status_5	RO	○	○	○	○	–	–	–	–
dmc_memc_cmd_5	WO	–	–	–	–	○	○	○	○
dmc_direct_cmd_5	WO	–	–	–	–	○	×	×	×
dmc_memory_cfg_5	R/W	○	○	○	○	○	×	×	×
dmc_refresh_prd_5	R/W	○	○	○	○	○	×	×	×
dmc_cas_latency_5	R/W	○	○	○	○	○	×	×	×
dmc_t_dqss_5	R/W	○	○	○	○	○	×	×	×
dmc_t_mrd_5	R/W	○	○	○	○	○	×	×	×
dmc_t_ras_5	R/W	○	○	○	○	○	×	×	×
dmc_t_rc_5	R/W	○	○	○	○	○	×	×	×
dmc_t_rcd_5	R/W	○	○	○	○	○	×	×	×
dmc_t_rfc_5	R/W	○	○	○	○	○	×	×	×
dmc_t_rp_5	R/W	○	○	○	○	○	×	×	×
dmc_t_rrd_5	R/W	○	○	○	○	○	×	×	×
dmc_t_wr_5	R/W	○	○	○	○	○	×	×	×
dmc_t_wtr_5	R/W	○	○	○	○	○	×	×	×
dmc_t_xp_5	R/W	○	○	○	○	○	×	×	×
dmc_t_xsr_5	R/W	○	○	○	○	○	×	×	×
dmc_t_esr_5	R/W	○	○	○	○	○	×	×	×
dmc_id_0_cfg_5 dmc_id_1_cfg_5 dmc_id_2_cfg_5 dmc_id_3_cfg_5 dmc_id_4_cfg_5 dmc_id_5_cfg_5	R/W	○	○	○	○	○	×	×	○
dmc_chip_0_cfg_5	R/W	○	○	○	○	○	×	×	○
dmc_user_config_5	WO	–	–	–	–	○	○	○	○

MPMC1 registers can't be read/write in reset status.

1. dmc_memc_status_5 (DMC Memory Controller Status Register)

Address = (0xF431_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined. Write as zero.
[9]	memory_banks	RO	0y1	Setting value of the maximum number of banks that the DMC supports: Fixed to 4 banks
[8:7]	–	–	Undefined	Read as undefined.
[6:4]	memory_ddr	RO	0y001	Types of SDRAM that the DMC supports: 0y000 = Reserved 0y001 = DDR SDRAM 0y011 = Reserved 0y010 = Reserved 0y1xx = Reserved
[3:2]	memory_width	RO	0y00	External memory bus width: 0y00 = 16-bit 0y01 = Reserved 0y10 = Reserved 0y11 = Reserved
[1:0]	memc_status	RO	0y00	Memory controller status: 0y00 = Config 0y01 = Ready 0y10 = Paused 0y11 = Low-power

[Description]

a. <memory_banks>
Setting value of the maximum number of banks that the DMC supports:
Fixed to 4 banks

b. <memory_ddr>
Types of SDRAM that the DMC supports. Fixed to 0y001.

c. <memory_width>
External memory bus width:
0y00 = 16-bit
0y01 = Reserved
0y10 = Reserved
0y11 = Reserved

d. <memc_status>
Memory controller status:
0y00 = Config
0y01 = Ready
0y10 = Paused
0y11 = Low-power

2. dmc_memc_cmd_5 (DMC Memory Controller Command Register)

Address = (0xF431_0000) + (0x0004)

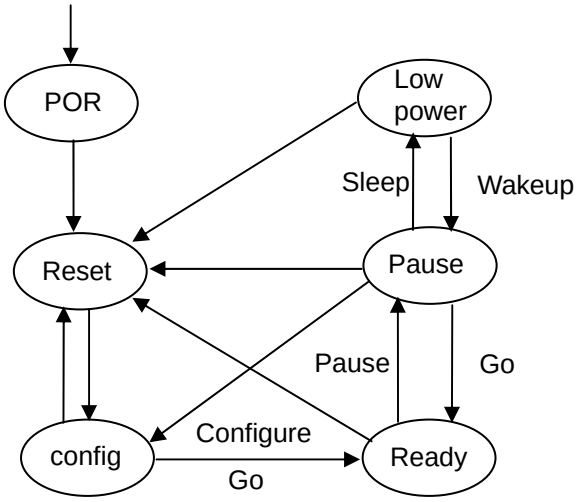
Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	memc_cmd	WO	–	Change the memory controller status: 0y000 = Go 0y001 = Sleep 0y010 = Wakeup 0y011 = Pause 0y100 = Configure

[Description]

a. <memc_cmd>

Settings of this register can change the DMC state machine. If a previously issued command for changing the states is being executed, a new command is issued after the previous command is completed.

The following diagram shows DMC state transitions.



External memory state transitions

When the DMC exits the Reset state, it automatically enters the Config state. The state transition from Pause to Config is effected by a Config command. Register settings must be made during the Config state.

When the DMC state is shifted to Ready, reads from and writes to the SDRAM are allowed. When a read or write is executed, the SDRAM will change from IDLE to ACTIVE.

When the DMC state is Ready, a Pause command shifts the DMC to Pause. The SDRAM state at this time varies depending on the immediately preceding command executed on the SDRAM. If a Read or Write has been executed, the SDRAM will be shifted to ACTIVE. If AutoRefresh has been executed, the SDRAM will be shifted to IDLE (Note).

When the DMC state is shifted from Pause to Low power by a Sleep command, after All Bank Precharge is executed, CKE will be driven "L" and the SDRAM will automatically enter the Self-refresh state.

When the DMC state is shifted from Low power to Pause by a Wakeup command, a Self-refresh Exit command will be issued. The SDRAM then automatically exists the Self-refresh state and enters the IDLE state.

Note: The SDRAM can be shifted from ACTIVE to IDLE by either of the following two settings: dmc_direct_cmd_5< memory_cmd>0y00 = Prechargeall or 0y01 = Autorefresh

3. dmc_direct_cmd_5 (DMC Direct Command Register)

This register sets each command for external memory and external memory mode register. This register sets the initial setting of external memory.

Address = (0xF431_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:22]	–	–	Undefined	Read as undefined. Write as zero.
[21:20]	chip_nmbr	WO	–	Always write 0y00
[19:18]	memory_cmd	WO	–	Determines the command required: 0y00 = Prechargeall 0y01 = Autorefresh 0y10 = Modereg or Extended modereg 0y11 = NOP
[17:16]	bank_addr	WO	–	Bits mapped to external memory bank address bits when command is Modereg access. 0y00 = bank0 0y01 = bank1 0y10 = bank2 0y11 = bank3
[15:14]	–	–	Undefined	Read as undefined. Write as zero.
[13:0]	addr_13_to_0	WO	–	Bits mapped to external memory address bits [13:0] when command is Modereg access.

Note:

Use dmc_direct_cmd_5 to configure cas latency of DDR_SDRAM memory,
The setting of cas latency(CL) is different from SDR_SDRAM.
The CL setting value of memory controler must be 1 smaller than the CL setting value of DDR_SDRAM memory.

Examples:
dmc_cas_latency_5 ← 0x00000004 (set memory controller CL = 2)
dmc_direct_cmd_5 ← 0x00080033 (set DDR SDRAM memory CL = 3)

[Description]

a. <memory_cmd>

Determines the command required:
0y00 = Prechargeall
0y01 = Autorefresh
0y10 = Modereg or Extended modereg
0y11 = NOP

b. <bank_addr>

Bits mapped to external memory bank address bits when command is Modereg access.
0y00 = bank0
0y01 = bank1
0y10 = bank2
0y11 = bank3

c. <addr_13_to_0>

Bits mapped to external memory address bits [13:0] when command is Modereg access.
Corresponding to external memory address bit

4. dmc_memory_cfg_5 (DMC Memory Configuration Register)

Address = (0xF431_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:23]	–	–	Undefined	Read as undefined. Write as zero.
[22:21]	active_chips	R/W	0y00	Always write 0y00
[20:18]	–	–	Undefined	Read as undefined. Write as zero.
[17:15]	memory_burst	R/W	0y010	Set the read and write burst length for the SDRAM 0y000 = Reserved 0y001 = Burst 2 0y010 = Burst 4 0y011 = Burst 8 0y100 = Burst 16 other = Reserved
[14]	stop_mem_clock	R/W	0y0	memory clock stop: 0y1 = Enable 0y0 = Disable
[13]	auto_power_down	R/W	0y0	SDRAM auto Powerdown Enable: 0y1 = Enable 0y0 = Disable
[12:7]	power_down_prd	R/W	0y000000	Number of SDRAM automatic Powerdown memory clocks: (Min. value = 1) 0y000001 to 0y111111
[6]	ap_bit	R/W	0y0	The position of the auto-precharge bit in the memory address: 0y0 = address bit 10 0y1 = address bit 8
[5:3]	row_bits	R/W	0y100	The number of row address bits: 0y000 = 11 bits 0y001 = 12 bits 0y010 = 13 bits 0y011 = 14 bits 0y100 = 15 bits 0y101 = 16 bits other = Reserved
[2:0]	column_bits	R/W	0y000	The number of column address bits: 0y000 = 8 bits 0y001 = 9 bits 0y010 = 10 bits 0y011 = 11 bits 0y100 = 12 bits other = Reserved

[Description]

a. <memory_burst>

Set the burst length of the memory access controller.

This needs to correspond with the burst length of the memory configured in the dmc_direct_cmd_5 register.

b. <stop_mem_clock>

The clock supply to the SDRAM can be stopped while it is not being accessed. When an SDRAM access request occurs again, the clock is automatically restarted.

Note1: Depending on the SDRAM type, it may not be possible to stop the clock supply to the SDRAM while it is not being accessed. When using this function, be sure to carefully check the specifications of the SDRAM to be used.

Note2: The memory clock stop function and the SDRAM auto Powerdown function cannot be used concurrently. Use only either of the two.

c. <auto_power_down>

When no SDRAM access request is present and the command FIFO of the memory controller becomes empty, the SDRAM can be placed into Powerdown mode by automatically disabling CKE after the number of clock cycles specified in the power_down_prd field. When an SDRAM access request occurs again, CKE is automatically enabled to exit the Powerdown mode.

Note: The memory clock stop function and the SDRAM auto Powerdown function cannot be used concurrently. Use only either of the two.

d. <row_bits><column_bits>

These bits set the row and column addresses.

Supported selectable memory is limited by the summation of column address and row address.

In case of 32bit bus, less than R+C=25 bits (128Mbytes) then 512Mbytes for 4 banks

In case of 16bit bus, less than R+C=26 bits (128Mbytes) then 512Mbytes for 4 banks

5. dmc_refresh_prd_5 (DMC Refresh Period Register)

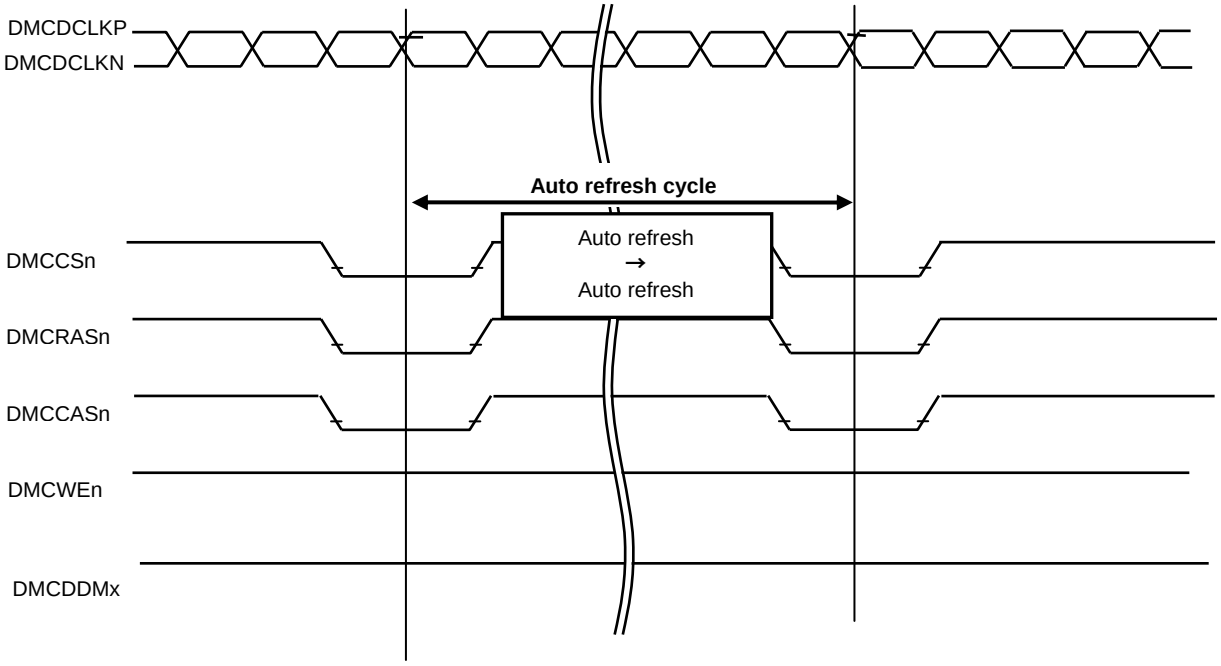
Address = (0xF431_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:15]	–	–	Undefined	Read as undefined. Write as zero.
[14:0]	refresh_prd	R/W	0x0A60	Auto-refresh cycle (number of memory clocks): 0x0000 to 0x7FFF

[Description]

a. <refresh_prd>

The value of the refresh counter decrements from the value set in the dmc_refresh_prd_5 (the number of memory clocks), and when the counter reaches zero, the Autorefresh command is issued to external memory.



6. dmc_cas_latency_5 (DMC CAS Latency Register)

Address = (0xF431_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:1]	cas_latency	R/W	0y11	CAS latency setting (number of memory clocks) 0y000 to 0y111
[0]	cas_half_cycle	R/W	0y0	set CAS latency offset 0y0 = 0 offset 0y1 = Half cycle offset

Note:

Use dmc_cas_latency_5 to configure cas latency of memory controler,
The setting of cas latency(CL) is different from SDR_SDRAM.
The CL setting value of memory controler is 1 smaller than the CL setting value of DDR_SDRAM memory.

Example:
dmc_cas_latency_5 ← 0x00000004 (set memory controller CL = 2)
dmc_direct_cmd_5 ← 0x00080033 (set DDR SDRAM memory CL = 3)

[Description]

a. <cas_latency>
CAS latency setting (number of memory clocks): 0y000 to 0y111

b. <cas_half_cycle>
CAS latency offset setting:
0y0 = 0 offset
0y1 = Half-cycle offset

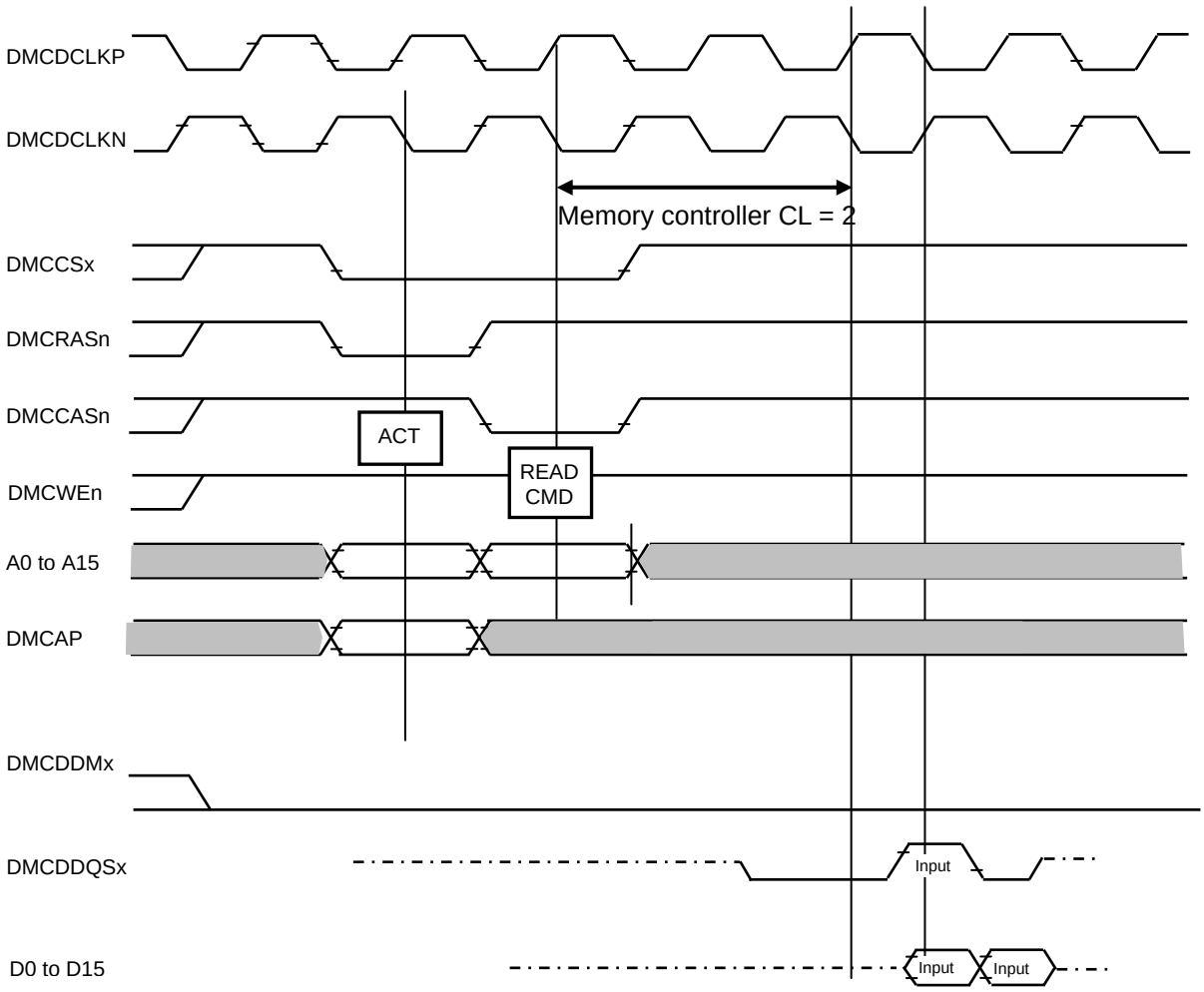


Figure 3.10.24 CAS latency example (CL=2)

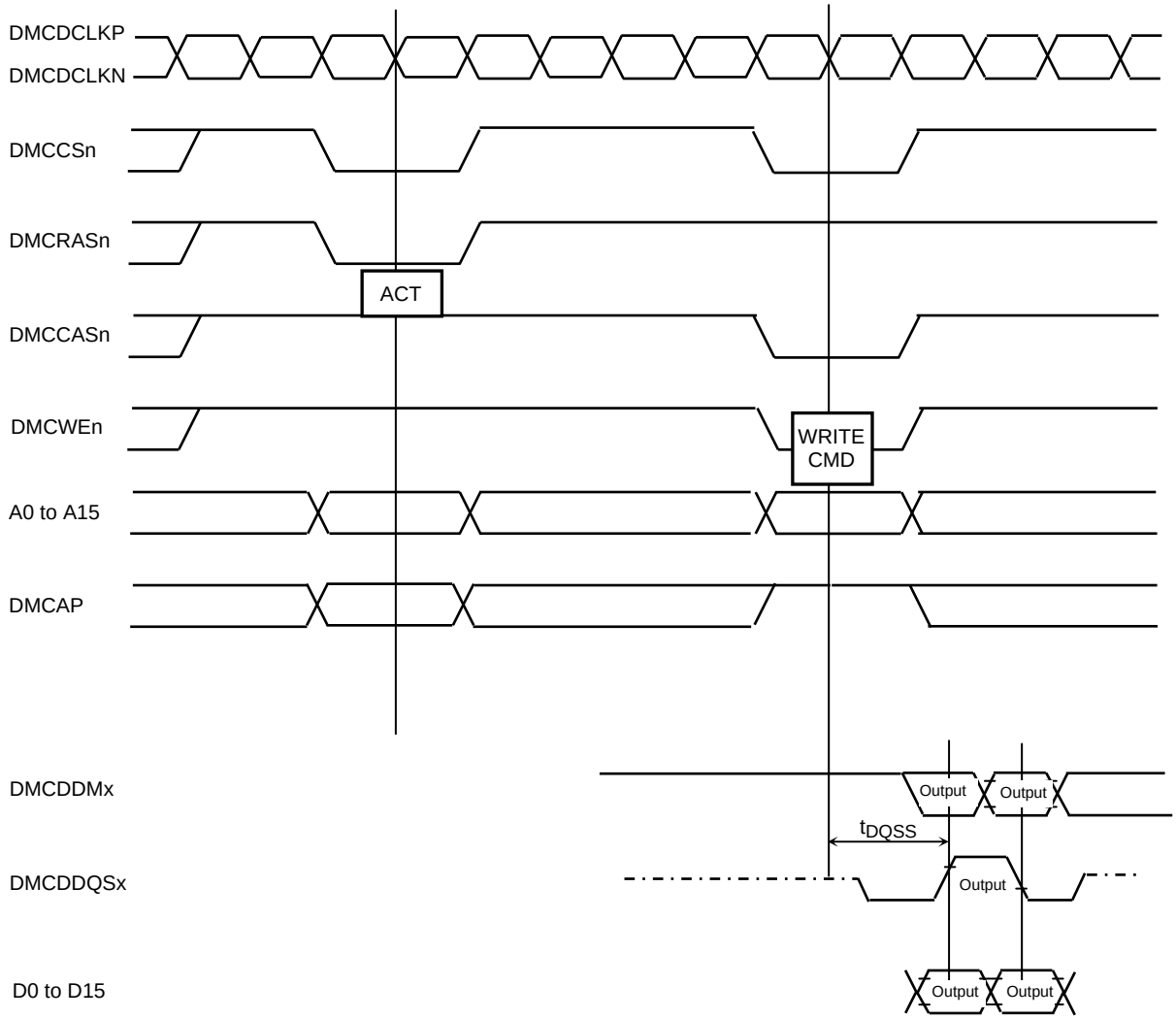
7. dmc_t_dqss_5 (DMC t_dqss Register)

Address = (0xF431_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Write as zero.
[1:0]	t_dqss	R/W	0y01	DQS setting (number of memory clocks): 0y00 to 0y11

[Description]

a. <t_dqss>
Set DQS (memory clocks):
0y00 to 0y11



8. dmc_t_mrd_5 (DMC t_mrd Register)

Address = (0xF431_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	t_mrd	R/W	0y0000010	Mode register command time (Number of memory clocks) 0x00 to 0x7F

[Description]

a. <t_mrd>

Set time from mode register command time set by the direct command register (dmc_direct_cmd_5<addr_13_to_0>) to all other commands (memory clocks):
0x00 to 0x7F

※ Depending on other AC settings and operations, the actual delay time may be longer than the specified time. Set the minimum number of clocks in this register.

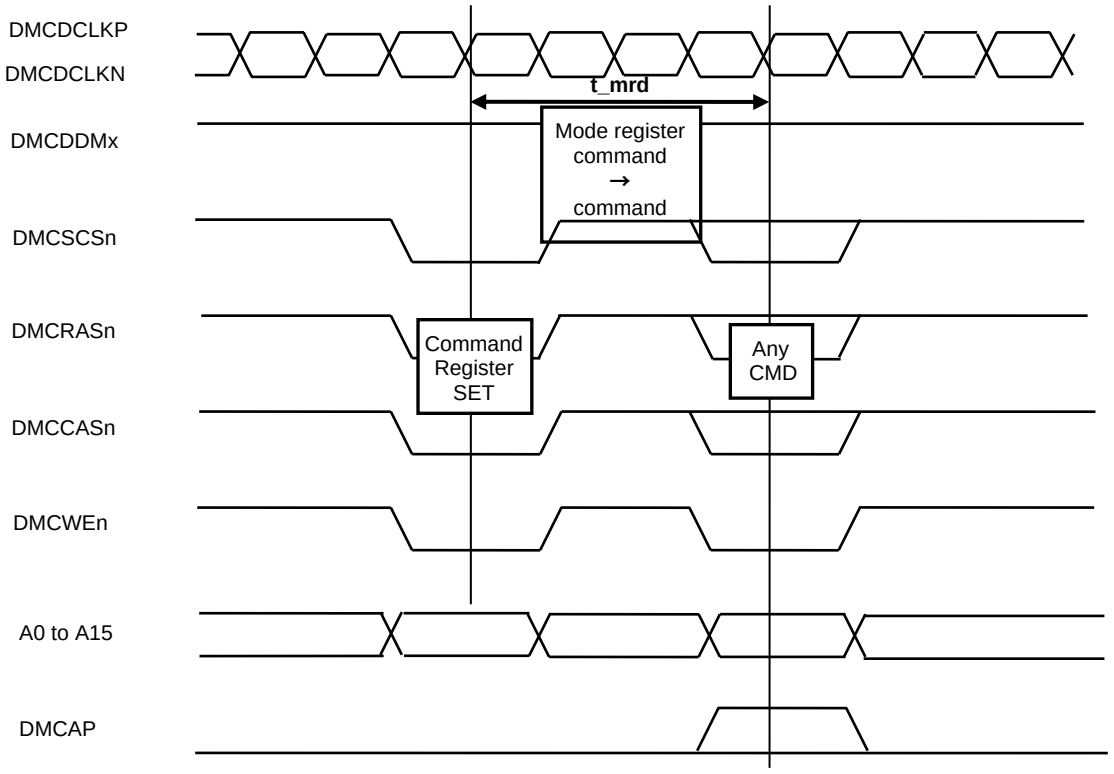


Figure 3.10.25 <t_mrd> set the time from mode register write to command

9. dmc_t_ras_5 (DMC t_ras Register)

Address = (0xF431_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	t_ras	R/W	0x7	Time between RAS and Precharge (number of memory clocks) 0x0 to 0xF

[Description]

a. <t_ras>

Time between RAS and Precharge (number of memory clocks)

0x0 to 0xF

※ Depending on other AC settings and operations, the actual delay time may be longer than the specified time. Set the minimum number of clocks in this register.

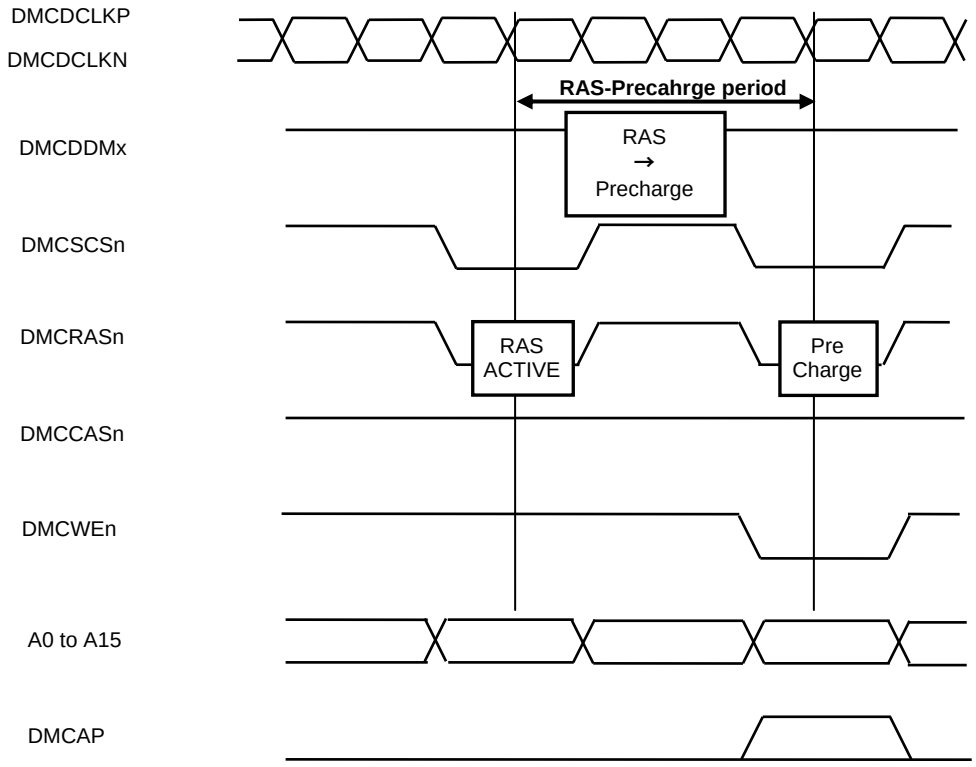


Figure 3.10.26 Time from Active to Precharge

10. dmc_t_rc_5 (DMC t_rc Register)

Address = (0xF431_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	t_rc	R/W	0y1011	Delay between Active bank A and Active bank A (Number of memory clocks) 0x0 to 0xF

[Description]

a. <t_rc>

Set delay time from Active bank command to Active command time in the same bank (memory clocks):

0x0 ~ 0xF

※ Depending on other AC settings and operations, the actual delay time may be longer than the specified time. Set the minimum number of clocks in this register.

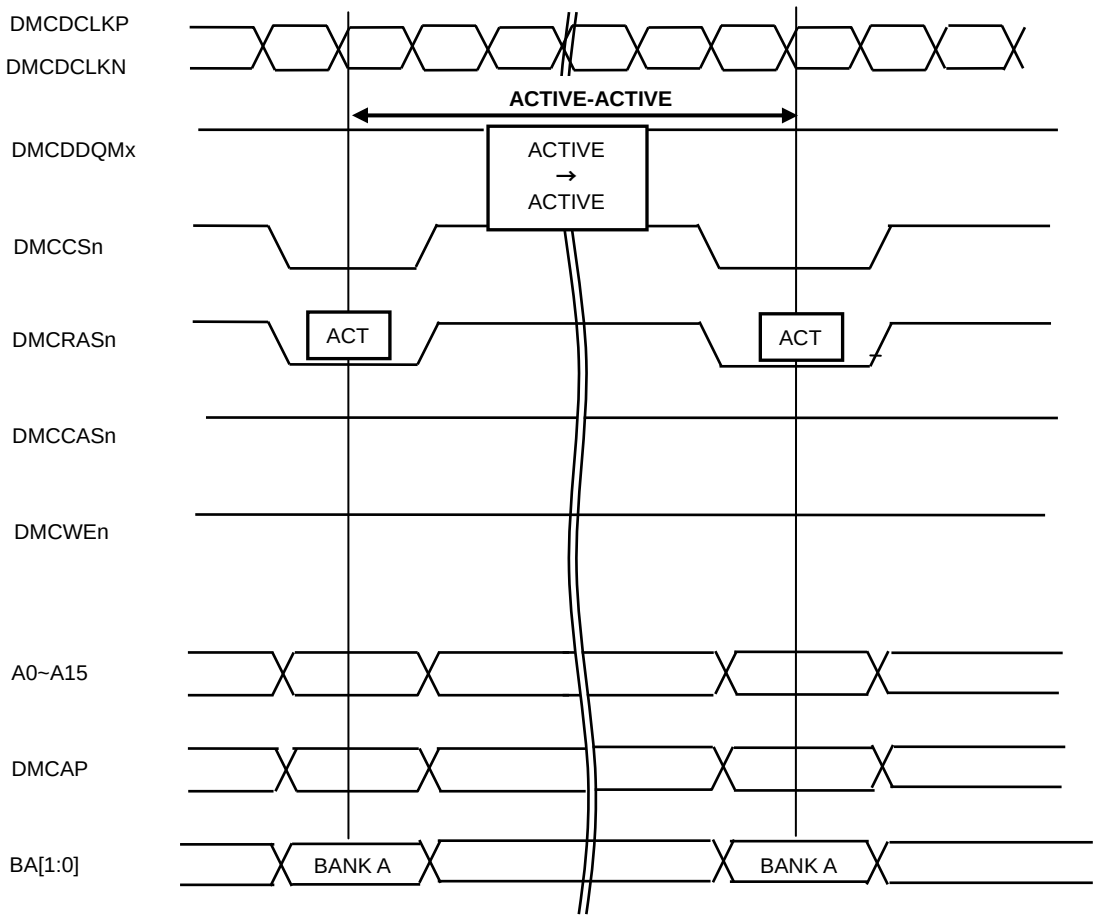


Figure 3.10.27 from Active bank A to Active bank A

11. dmc_t_rcd_5 (DMC t_rcd Register)

Address = (0xF431_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:3]	schedule_rcd	R/W	0y011	Set min delay from RAS to CAS: Set to (t_rcd setting value -3)
[2:0]	t_rcd	R/W	0y101	Set min delay from RAS to CAS (number of memory clocks): 0y000 to 0y111

[Description]

a. <schedule_rcd>
Set min delay from RAS to CAS (number of memory clocks):
Set to (t_rcd setting value -3)

b. <t_rcd>
Set min delay from RAS to CAS (number of memory clocks):
0y000 to 0y111

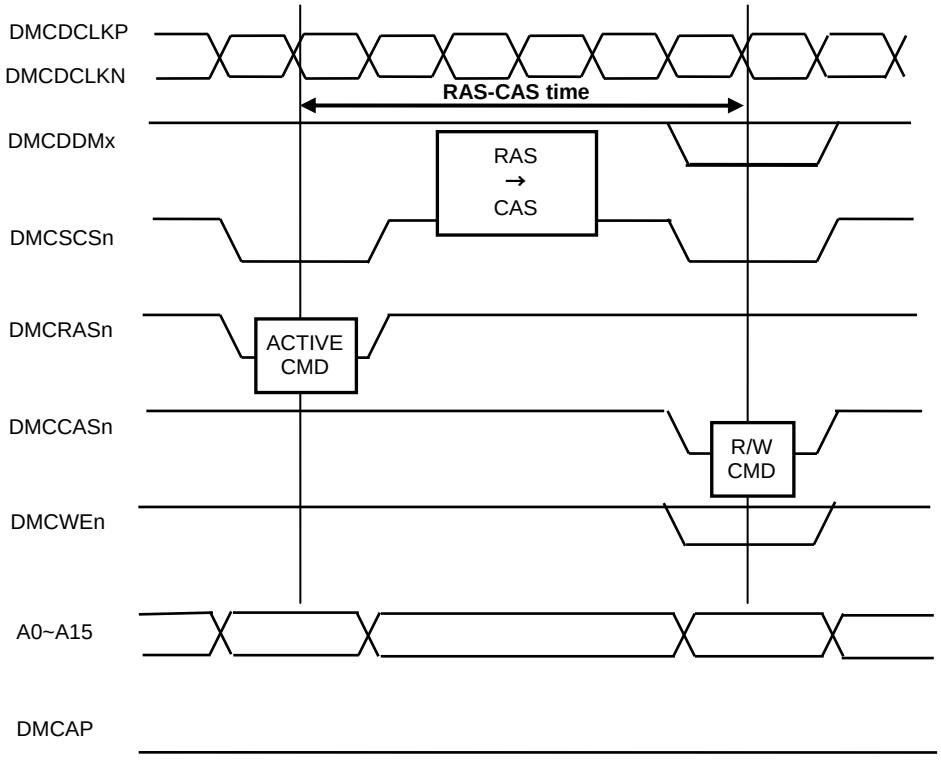


Figure 3.10.28 Time from Active to Read/ Write Command

12. dmc_t_rfc_5 (DMC t_rfc Register)

Address = (0xF431_0000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:5]	schedule_rfc	R/W	0y10000	Autorefresh command time setting Set to (t_rfc setting value -3)
[4:0]	t_rfc	R/W	0y10010	Autorefresh command time setting (Number of memory clocks): 0y00000 to 0y11111

[Description]

a. <schedule_rfc>
Autorefresh command time setting
Set to (t_rfc setting value -3)

b. <t_rfc>
Autorefresh command time setting (number of memory clocks):
0y00000 to 0y11111

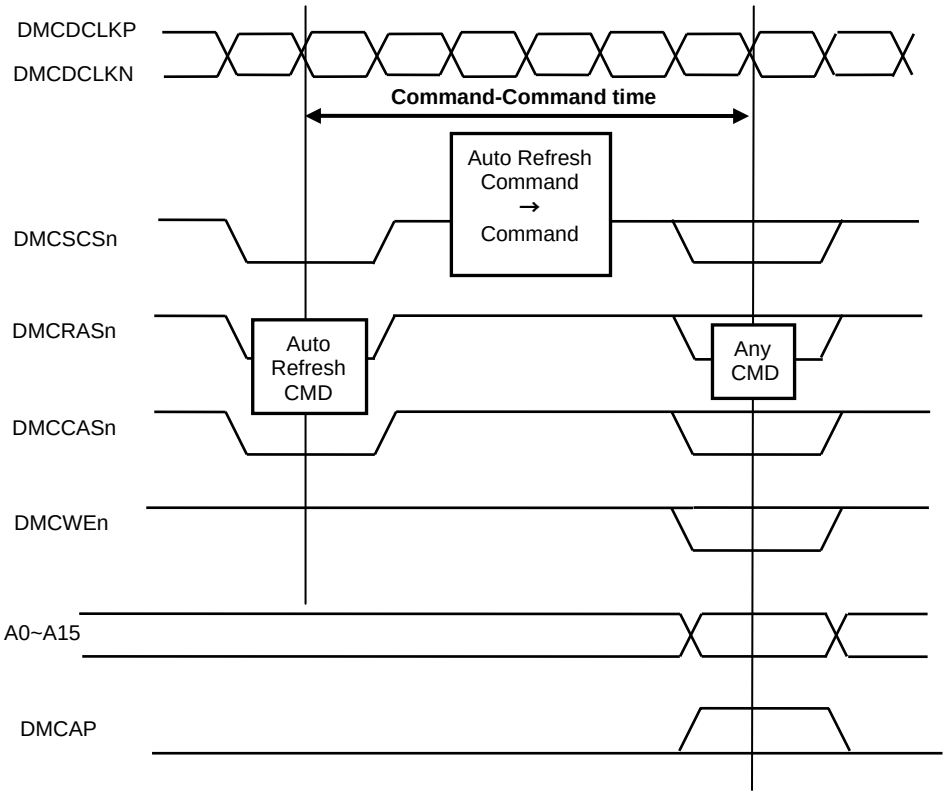


Figure 3.10.29 Time from Autorefresh command to other command

13. dmc _t_rp_5 (DMC t_rp Register)

Address = (0xF431_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:3]	schedule_rp	R/W	0y011	Precharge delay setting to RAS Set to (t_rp setting value -3)
[2:0]	t_rp	R/W	0y101	Set the time from Precharge to RAS (number of memory clocks): 0y000 to 0y111

[Description]

a. <schedule_rp>
Set the time from Precharge to RAS
Set to (t_rp setting value -3)

b. <t_rp>
Set the time from Precharge to RAS (number of memory clocks)
0y000 to 0y111

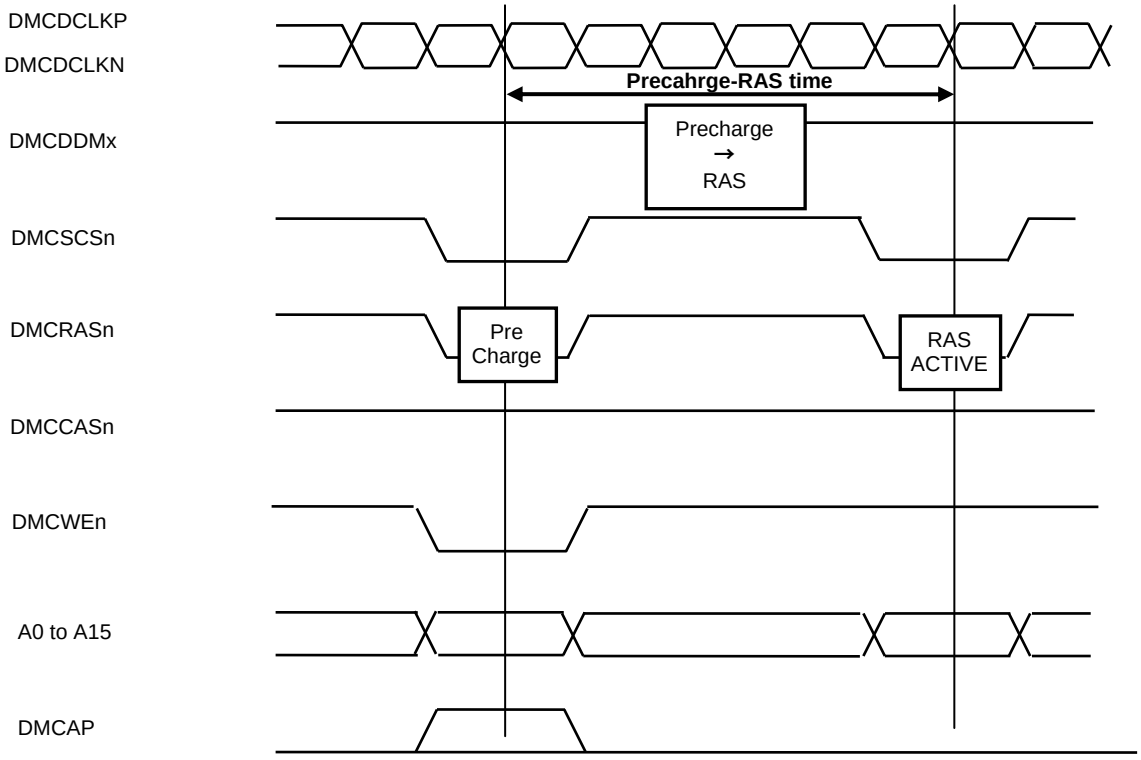


Figure 3.10.30 Precharge to command, Autorefresh time

14. dmc_t_rrd_5 (DMC t_rrd Register)

Address = (0xF431_0000) + (0x0034)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:0]	t_rrd	R/W	0y0010	Delay time from Active bank A to Active bank B (Number of memory clocks): 0x0 to 0xF

[Description]

a. <t_rrd>
Delay time from Active bank A to Active bank B (number of memory clocks):
0x0 to 0xF

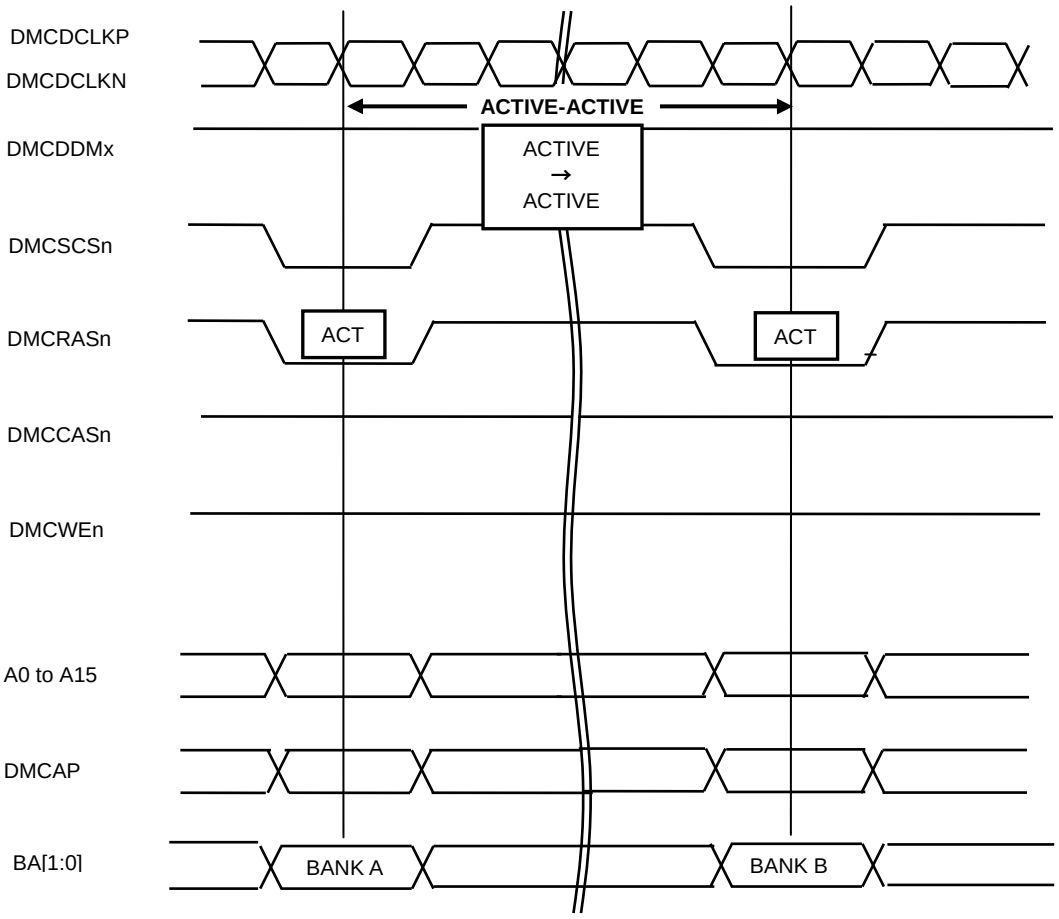


Figure 3.10.31 Time between Active bank A and other Active bank B

15. dmc_t_wr_5 (DMC t_wr Register)

Address = (0xF431_0000) + (0x0038)

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	t_wr	R/W	0y011	Delay from write last data to Precharge (Number of memory clocks): 0y000 to 0y111

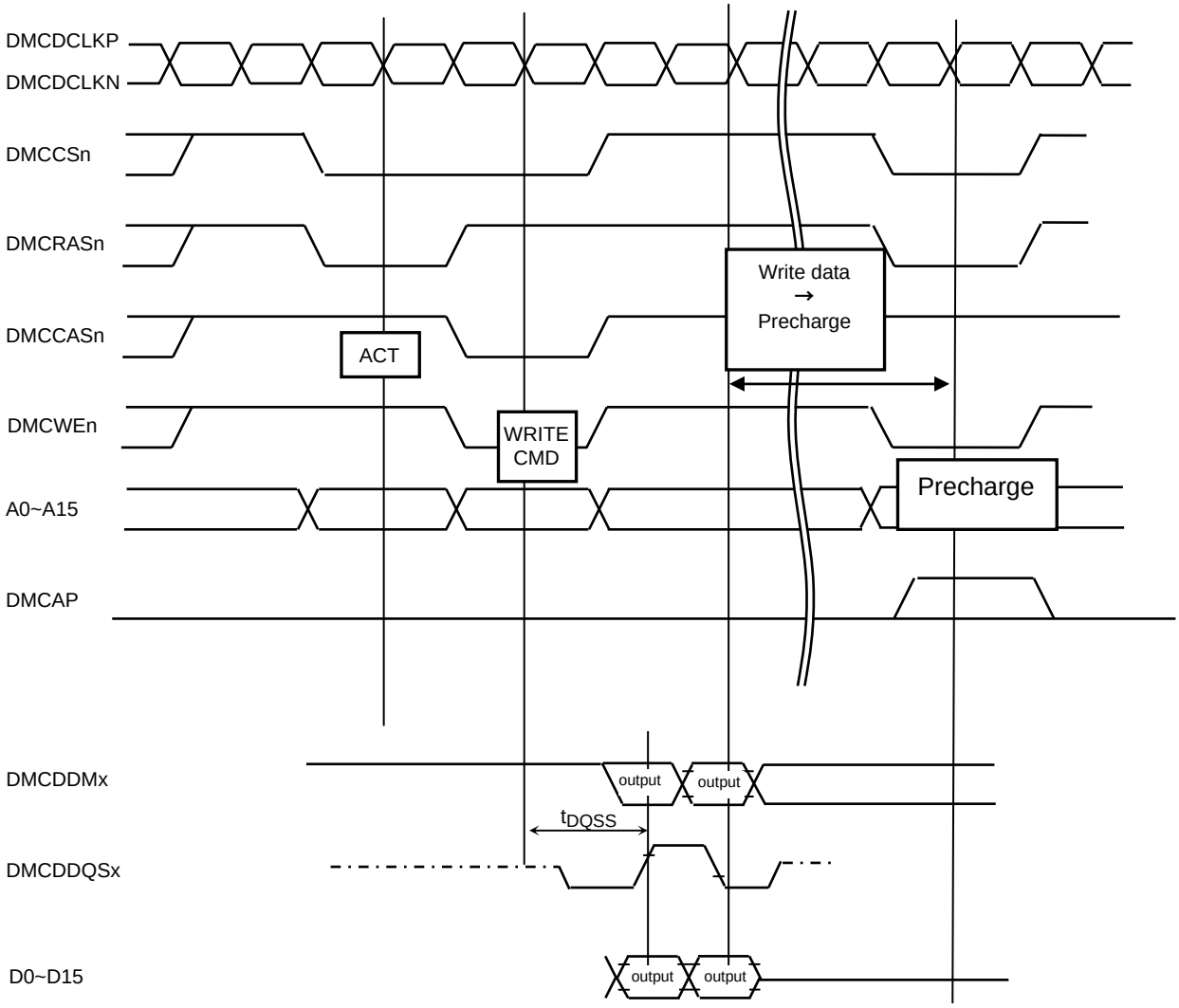
[Description]

a. <t_wr>

Delay from write last data to Precharge (Number of memory clocks).

Actual time (memory clocks): <t_wr> + 1.

But when <t_wr> = 0y000, actual time (memory clocks) = 9 memory clocks.



16. dmc_t_wtr_5 (DMC t_wtr Register)

Address = (0xF431_0000) + (0x003C)

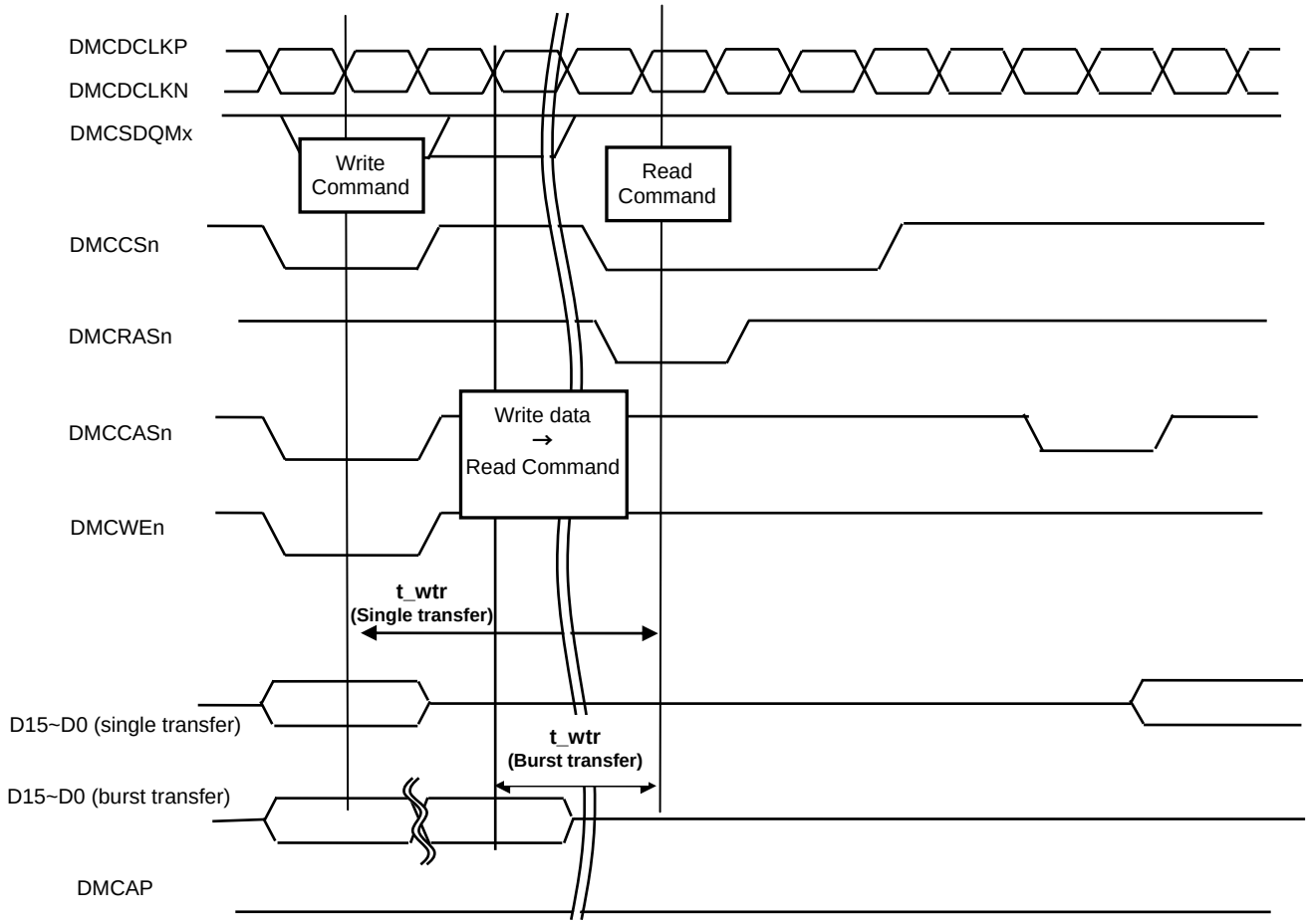
Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	t_wtr	R/W	0y010	Setting value from the last write data to read command (memory clocks): 0y000 to 0y111

[Description]

a. <t_wtr>

Setting value from write last data to read command (memory clocks)

Note: When <t_wtr> = 0y000, Actual time (memory clocks) = 8 memory clocks.



17. dmc _t_xp_5 (DMC t_xp Register)

Address = (0xF431_0000) + (0x0040)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_xp	R/W	0x01	Setting value of the exit power-down command time (Number of memory clocks): 0x00 to 0xFF

[Description]

a. <t_xp>

Set time from Powerdown Exit command to other command (memory clocks):

Actual time (memory clocks): t_xp set value + 1

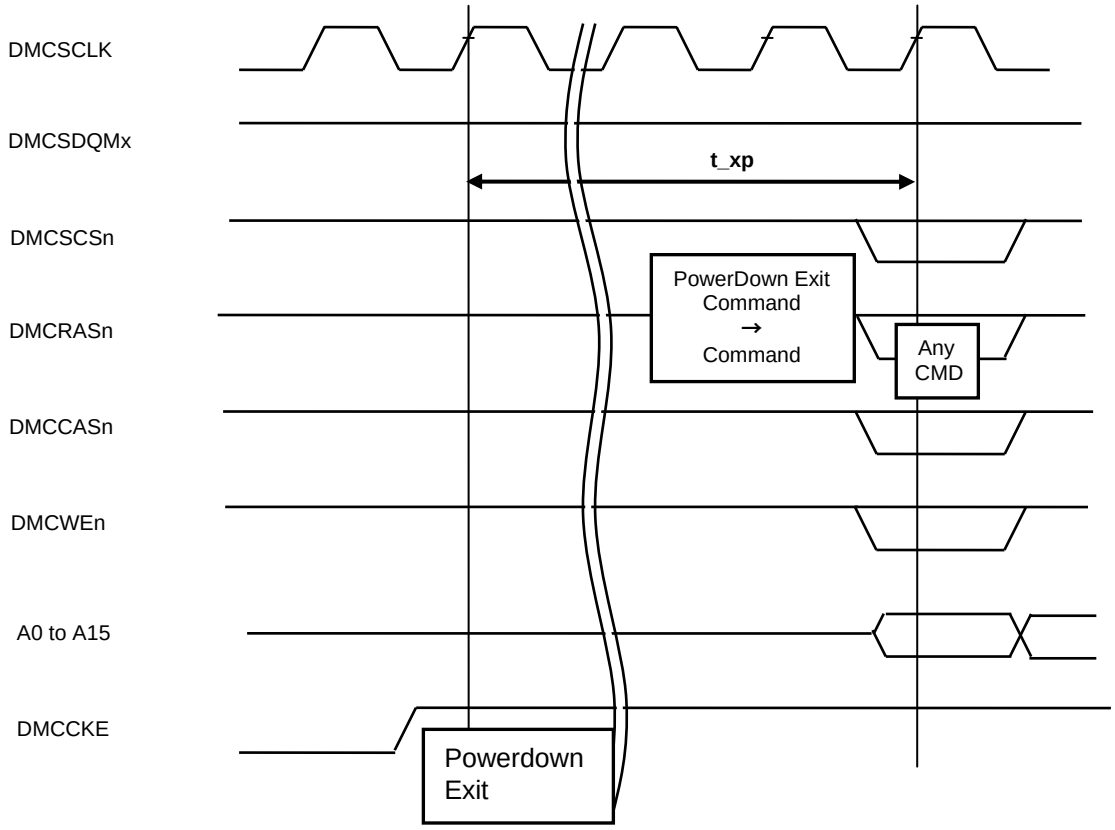


Figure 3.10.32 Time from Powerdown entry to Exit

18. dmc_ t_xsr_5 (DMC t_xsr Register)

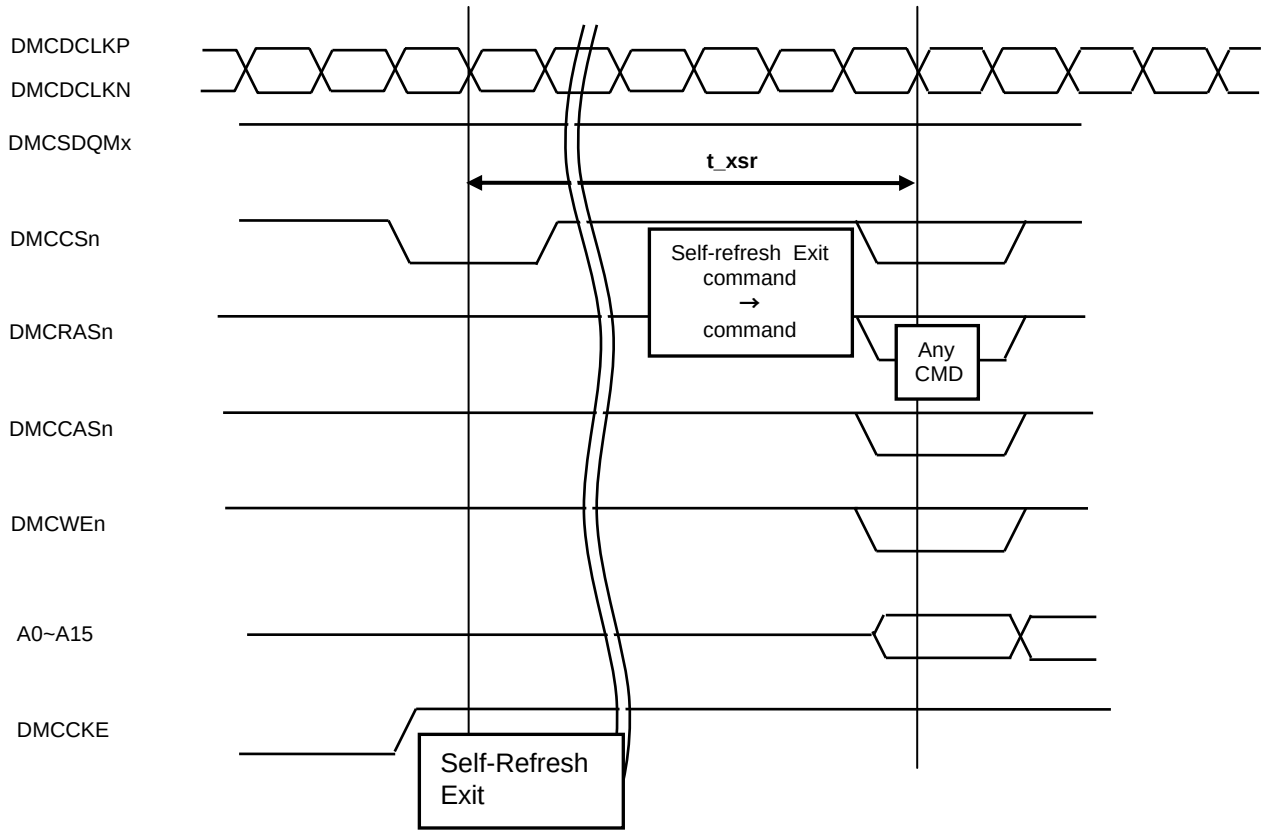
Address = (0xF431_0000) + (0x0044)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_xsr	R/W	0x0A	Set the Self-refresh Exit command time: (memory clocks): 0x00 to 0xFF

[Description]

a. <t_xsr>

Set time from Self-refresh Exit command to other command (memory clocks)
0x00 to 0xFF



19. dmc_t_esr_5 (DMC t_esr Register)

Address = (0xF431_0000) + (0x0048)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_esr	R/W	0x14	The minimum time from Self-refresh Entry to Exit: (memory clocks) 0x00 to 0xFF

Note: Self-refersh Exit is triggered by Wakeup direct command. This register is to set the minimum time from Self-refresh Entry to Exit.

[Description]

a. <t_esr>

The minimum time from Self-refresh Entry to Exit (memory clocks)
0x00 to 0xFF

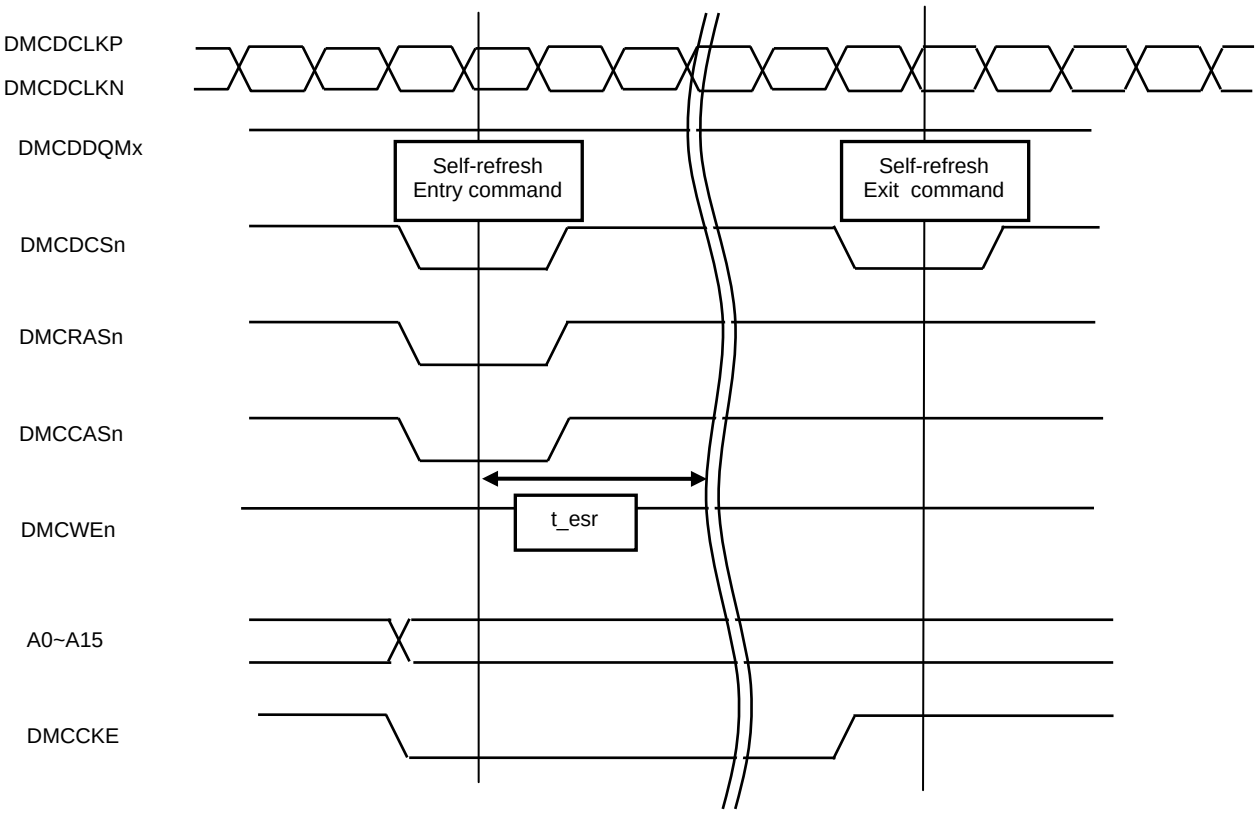


Figure 3.10.33 SelfRefresh Entry and Exit

20. dmc_id_<0-5>_cfg_5 (DMC id_<0-5>_cfg Registers)

Address = (0xF431_0000) + (0x0100)
Address = (0xF431_0000) + (0x0104)
Address = (0xF431_0000) + (0x0108)
Address = (0xF431_0000) + (0x010C)
Address = (0xF431_0000) + (0x0110)
Address = (0xF431_0000) + (0x0114)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:2]	qos_max	R/W	0x00	maximum QoS: 0x00 ~ 0xFF
[1]	qos_min	R/W	0y0	minimum QoS selection: 0y0 = QoS max mode 0y1 = QoS min mode
[0]	qos_enable	R/W	0y0	QoS setting: 0y0 = Disable 0y1 = Enable

QoS setting register list

Register	Address	Correspond to AHB
dmc_id_0_cfg_5	(0xF431_0000) + (0x0100)	AHB0 : CPU Data
dmc_id_1_cfg_5	(0xF431_0000) + (0x0104)	AHB1 : CPU Inst
dmc_id_2_cfg_5	(0xF431_0000) + (0x0108)	AHB2 : LCDC
dmc_id_3_cfg_5	(0xF431_0000) + (0x010C)	AHB3 : multilayer bus matrix2 (LCDDA, USB)
dmc_id_4_cfg_5	(0xF431_0000) + (0x0110)	AHB4 : DMA1
dmc_id_5_cfg_5	(0xF431_0000) + (0x0114)	AHB5 : DMA2

[Description]

a. <qos_max>
QoS maximum value setting:
0x00 to 0xFF

b. <qos_min>
Minimum QoS selction:
0y0 = QoS max mode
0y1 = QoS min mode,
QoS minimum have priority over QoS maximum

c. <qos_enable>
Enable QoS:
0y0 = Disable
0y1 = Enable

21. dmc_chip_0_cfg_5 (DMC chip_0_cfg Registers)

Address = (0xF431_0000) + (0x0200)

Bit	Bit Symbol	Type	Reset Value	Description
[31:17]	–	–	Undefined	Read as undefined. Write as zero.
[16]	brc_n_rbc	R/W	0y0	SDRAM address structure: 0y0 = row, bank, column 0y1 = bank, row, column
[15:8]	address_match	R/W	0xff	Set the start address [31:24] : 0x00 to 0xFF
[7:0]	address_mask	R/W	0x00	Set the mask value of the start address [31:24] : 0y0 = Do not compare 0y1 = Compare 0x00 to 0xFF

[Description]

a. <brc_n_rbc>

SDRAM address structure

0y0 = row, bank, column

0y1 = bank, row, column

b. <address_match>

Set the start address [31:24].

If the size of connected memory is less than 512 bytes, do not access unused DMC area outside the specified CS area.

Note: Before setting the start address, check the valid address area by referring to Chapter 3.3 Memory Map.

c. <address_mask>

This register specifies the CS area. Set whether or not each bit in the start address [31:24] should be compared.

0y0 = Do not compare

0y1 = Compare

22. dmc_user_config_5 (DMC user_config Register)

Address = (0xF431_0000) + (0x0304)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	Reserved	–	Undefined	Read as undefined. Write as zero.
[6:4]	dqs_in	WO	0y000	DDR SDRAM constant value setting: Fix to 0y101
[3:1]	dmc_clk_in	WO	0y000	DDR SDRAM constant value setting: Fix to 0y100
[0]	sdr_width	WO	0y0	data bus width of external DDR SDRAM : 0y0: 16-bit 0y1: Reserved

[Description]

a. <sdr_width>

Set the memory data bus width of corresponding external SDR memory:
0y0 = 16bit
0y1 = Reserved

3.10.5.2 SMC (Static Memroy Controller)

This device contains and SMC (Static Memory Controller) that controls the external memory (NOR Flash memory, Mask ROM SRAM and etc.).

(1) SMC (Static Memory Controller)

Table 3.10.11 shows Feature of SMC.

Table 3.10.11 Feature of SMC

Features	Chip select 0/1
Support memory	External asynchronous memory (NOR Flash memory and SRAM, etc.) Support separate bus only
Data bus width	16bit data bus width
Access areas	4 areas support by Chip select. Max acceess area: SMCCS0n: 512 MB SMCCS1n: 512 MB
Timing adjustment	Adjustable AC timing by register
	Support external wait request (only in Synchronous mode)
Clock	Selectable the clock for external pin (f_{HCLK} or f_{HCLK} /2) by the clock controller register CLKCR5<SEL_SMC_MCLK>
External control pin	D15 to D0, A23 to A0, SMCBE0n, SMCBE1n, , SMCCS0n, SMCCS1n,SMCOEn,SMCWEn,

(2) SMC (Static Memory Controller)

Figure 3.10.34 is a SMC block diagram.

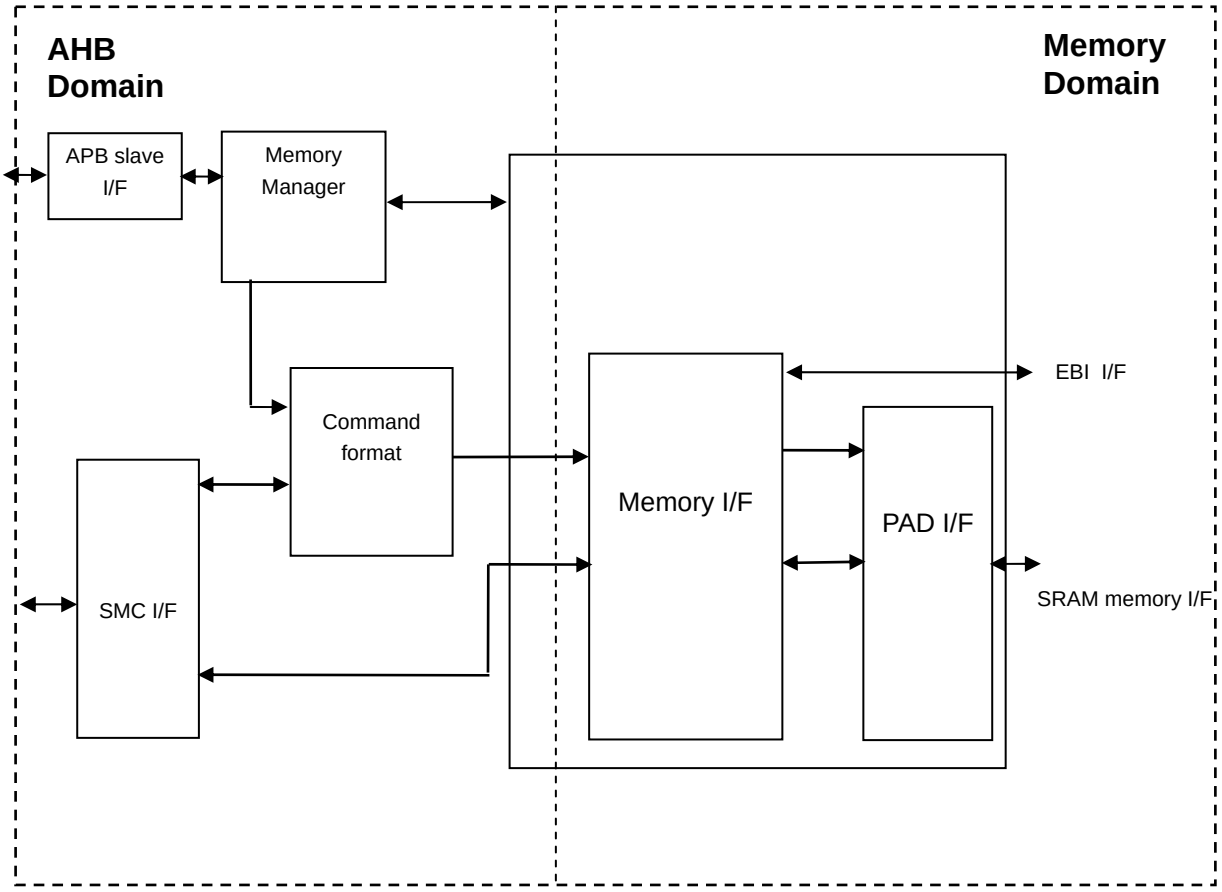


Figure 3.10.34 SMC Block Diagram

(a) Arbiter

The arbiter receives access commands from the SMC I/F and the memory manager, and after access arbitration, it passes the highest priority command to the memory I/F.

Data is read from the memory I/F to the SMC I/F.

(b) Memory manager

Updates timing registers and controls commands issued to memory.

(3) SMC Function

(a) APB slave I/F

The APB slave I/F adds a wait state for all reads and writes

More than one wait stat is generated in the following cases:

- Outstanding direct commands.
- A memory command is received, but the previous memory command has not been completed.

(b) Format

1. hazard processing

When selfsame stand-alone bus master access to an external memory, the actual access procedure to memory is executed in the instruction order.

However, if multiple bus master access to an external memory, the read and write data will be stored temporary into independent buffer and be executing by priority circuit. Therefore, the read and the write instruction may switch execution sequence. So please coordinate a variety of sequences, e.g. making an enough time for next instruction, checking whether or not previous execution is finished, the common-use memory data uses the internal memory and so on.

2. Access to the SRAM memory

- Standard SRAM access
- Memory address shifting
- Memory burst alignment

The burst align settings are necessary in order to support asynchronous page mode memory. Refer to SMC register of MPMC, smc_set_opmode_5 (SMC Set Opmode Register).

Note: In case of not having any page mode methods, e.g. NOR Flash, it is unnecessary to set burst align.

Memory burst length: Supported memory burst transfer length is 4 beats.

(c) Memory manager operation

The memory manager controls the SMC state and manages update of chip configuration registers.

(d) Memory I/F operation

The memory I/F issues commands and control their timings.

Table 3.10.12 Static Memory Setup Example

Base address = 0xF431_1000

Register address	Write data	Description
0x0014	0x00029266	smc_set_cycles_5
0x0018	0x00000809	smc_set_opmode_5
0x0010	0x00400000	smc_direct_cmd_5

(4) SMC Registers for MPMC1

Table 3.10.13 MPMC1 SMC SFR list

Base address = 0xF431_1000

Register Name	Address (base+)	Type	Reset value	Description
Reserved	0x0000	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0004	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0008	–	–	Writing prohibited
Reserved	0x000C	–	–	Writing prohibited
smc_direct_cmd_5	0x0010	WO	–	SMC Direct Command Register
smc_set_cycles_5	0x0014	WO	–	SMC Set Cycles Register
smc_set_opmode_5	0x0018	WO	–	SMC Set Opmode Register
Reserved	0x0020	–	Undefined	Read as undefined. Write as zero.
smc_sram_cycles0_0_5	0x0100	RO	0x0002B3CC	SMC SRAM Cycles Registers <0-1>
smc_sram_cycles0_1_5	0x0120			
Reserved	0x0140	RO	Undefined	Read as undefined
Reserved	0x0160	RO	Undefined	Read as undefined
smc_opmode0_0_5	0x0104	RO	0x20E00802	SMC Opmode Registers <0-1>
smc_opmode0_1_5	0x0124		0x60E00802	
Reserved	0x0144	RO	Undefined	Read as undefined
Reserved	0x0164	RO	Undefined	Read as undefined
Reserved	0x0200	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0204	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E00	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E04	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0E08	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FE0-0x0FEC	–	Undefined	Read as undefined. Write as zero.
Reserved	0x0FF0-0x0FFC	–	Undefined	Read as undefined. Write as zero.

Note: The APB supports only single-word 32-bit accesses. Read from or write to registers at single-word 32-bit mode.

1. smc_direct_cmd_5 (SMC Direct Command Register)

Address = (0xF431_1000) + (0x0010)

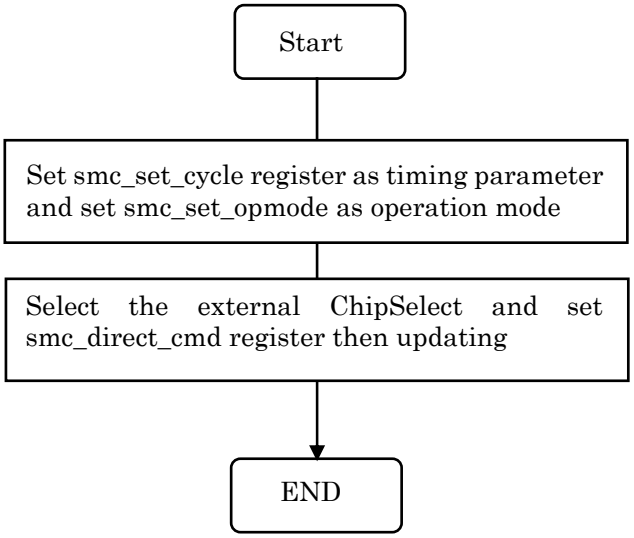
Bit	Bit Symbol	Type	Reset Value	Description
[31:26]	–	–	Undefined	Read as undefined. Write as zero.
[25:23]	chip_select	WO	–	CS selection: 0y000 = CS0 0y001 = CS1 0y010-0y111 = Reserved
[22:21]	cmd_type	WO	–	current command: 0y00 = Reserved 0y01 = Reserved 0y10 = UpdateRegs 0y11 = Reserved
[20:0]	–	–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

.

[Description]

a. <chip_select>
CS selection:
0y000 = CS0
0y001 = CS1
0y010 to 0y111 = Reserved

b. <cmd_type>
Current command:
0y00 = Reserved
0y01 = Reserved
0y10 = UpdateRegs
0y11 = Reserved


Start
Set smc_set_cycle register as timing parameter and set smc_set_opmode as operation mode
Select the external ChipSelect and set smc_direct_cmd register then updating
END

2. smc_set_cycles_5 (SMC Set Cycles Register)

Address = (0xF431_1000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:23]	–	–	Undefined	Read as undefined. Write as zero.
[22:20]	Reserved	–	Undefined	Read as undefined. Write as zero.
[19:17]	Set_t5	WO	–	Set value of t_{TR} (holding register) 0y000 to 0y111
[16:14]	Set_t4	WO	–	Set value of t_{PC} (holding register) 0y000 to 0y111
[13:11]	Set_t3	WO	–	Set value of t_{WP} (holding register) 0y000 to 0y111
[10:8]	Set_t2	WO	–	Set value of t_{CEOE} (holding register) 0y000 to 0y111
[7:4]	Set_t1	WO	–	Set value of t_{WC} (holding register) 0y0000 to 0y1111
[3:0]	Set_t0	WO	–	Set value of t_{RC} (holding register) 0y0000 to 0y1111

This register is provided to adjust the access cycle of static memory and should be set to satisfy the AC specifications of the memory to be used, If the wait signal by an external pin is also used, the access cycle is determined to satisfy the settings of both this register and the external wait signal.

Note that the external wait signal is only effective in synchronous mode. It cannot be used in asynchronous mode.

This is a holding register for enabling setting values. By executing of the following operations, the settings values of this register will be updated to the configuration register of the memory manager and enabled.

・The smc_direct_cmd Register indicates only a register update is taking place.

[Description]

a. <Set_t5>

Set value of t_{TR} (holding register).

0y000 to 0y111

b. <Set_t4>

Set value of t_{PC} (holding register).

0y000 to 0y111

c. <Set_t3>

Set value of t_{WP} (holding register).

0y000 to 0y111

d. <Set_t2>

Set value of t_{CEOE} (holding register).

0y000 to 0y111

e. <Set_t1>

Set value of t_{WC} (holding register).

0y0000 to 0y1111

f. <Set_t0>

Set value of t_{RC} (holding register).

0y0000 to 0y1111

Setting Example: SMC Set Cycles Register = 0x0002B1C3

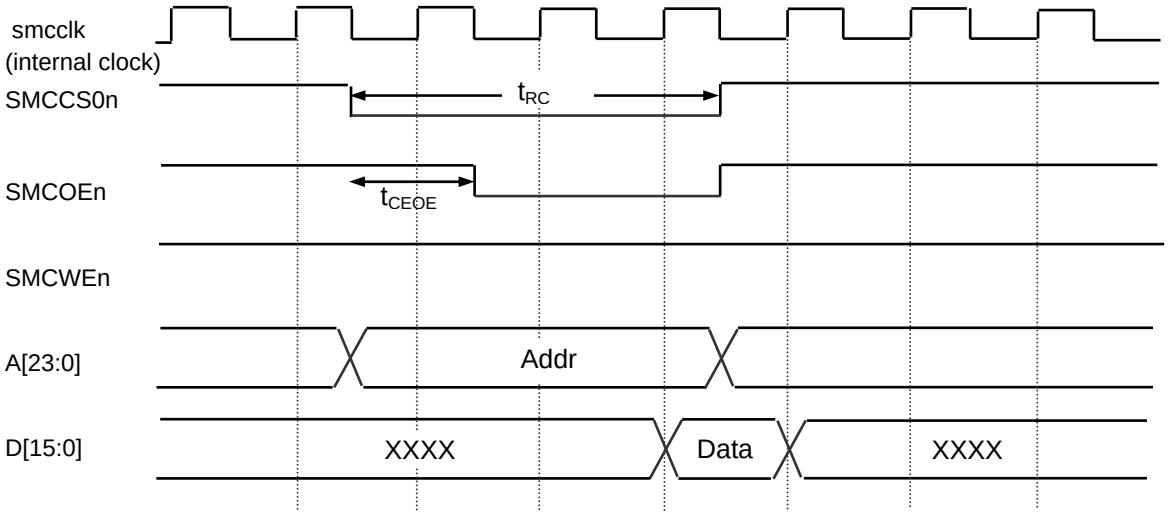


Figure 3.10.35 Asynchronous Read

Setting Example: SMC Set Cycles Register = 0x0002934C

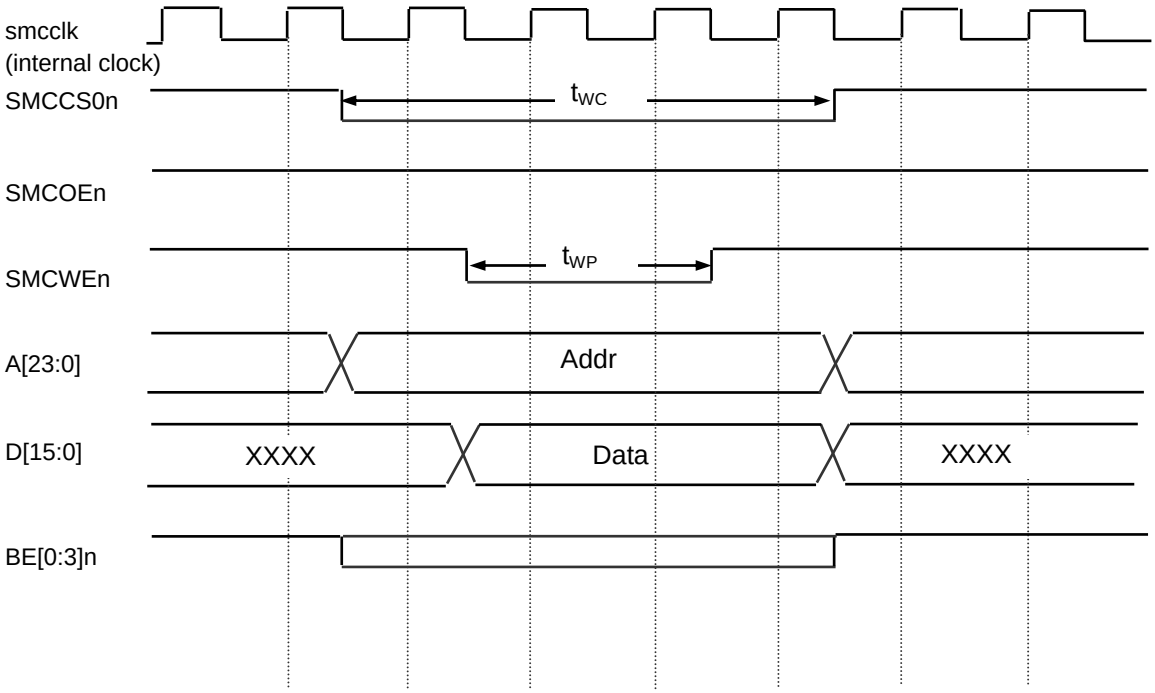


Figure 3.10.36 Asynchronous Write

Setting Example: SMC Set Cycles Register = 0x000272C3

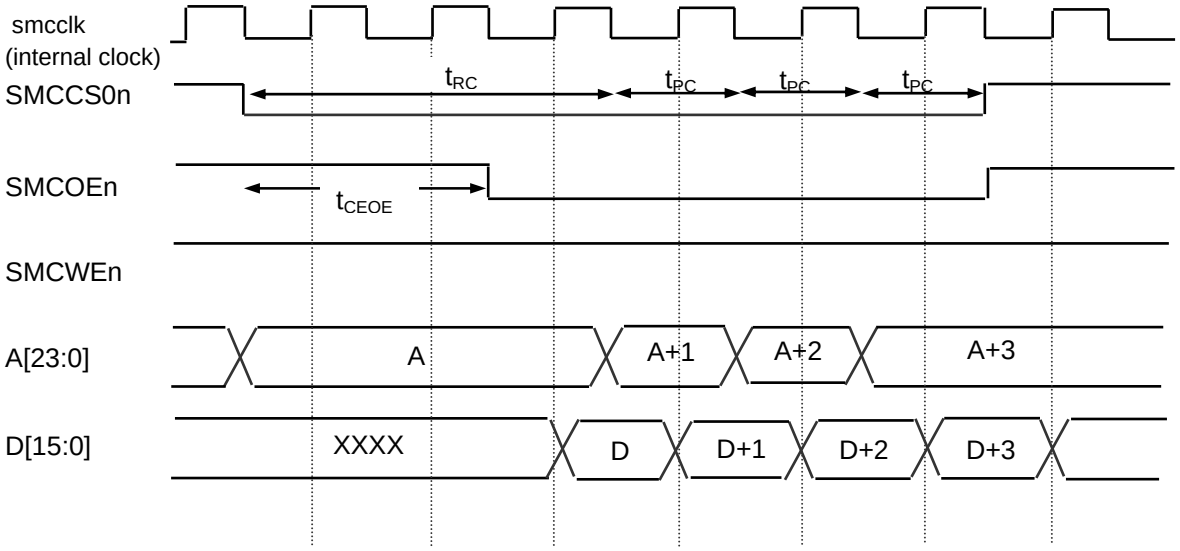


Figure 3.10.37 Asynchronous Page Read

Setting Example: SMC Set Cycles Register = 0x00029143

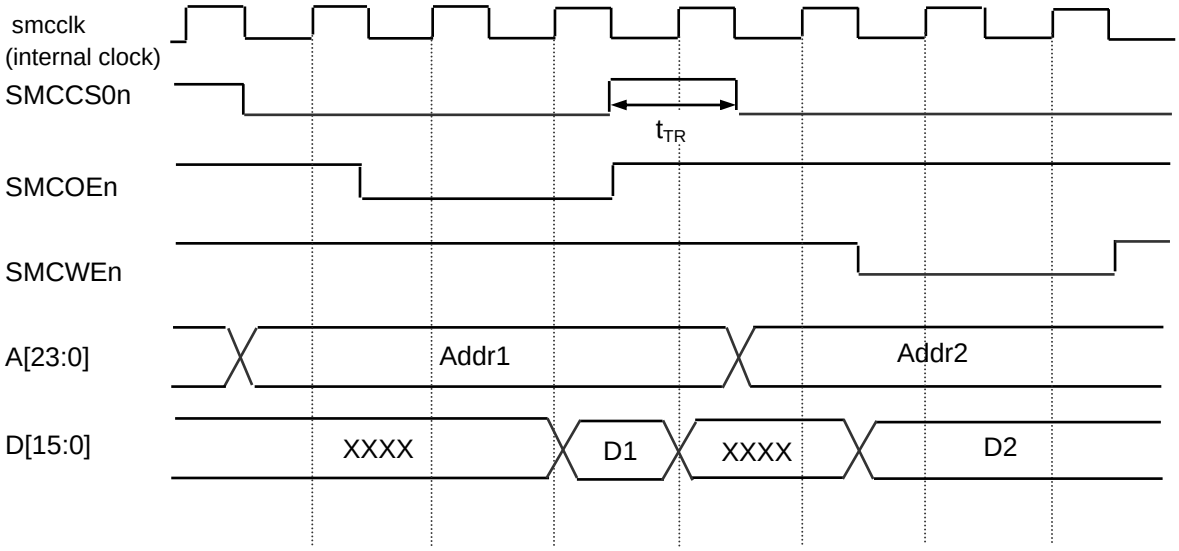


Figure 3.10.38 Asynchronous Write after Asynchronous Read

3. smc_set_opmode_5 (SMC Set Opmode Register)

Address = (0xF431_1000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:13]	set_burst_align	WO	–	Memory burst boundary split setting: 0y000 = bursts can cross any address boundary 0y001 = split at the 32-beat burst boundary 0y010 = split at the 64-beat burst boundary 0y011 = split at the 128-beat burst boundary 0y100 = split at the 256-beat burst boundary other = Reserved
[12]	set_bls	WO	–	Byte Enable (SMCBE0-1) timing setting: 0y0 = SMCCSn timing 0y1 = SMCWEn timing
[11]	Reserved	WO	–	Read as undefined. Write as zero.
[10]	-	–	Undefined	Read as undefined. Write as zero.
[9:7]	set_wr_bl	WO	–	Write burst length (holding register) 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[6]	set_wr_sync	WO	–	Holding register of the wr_sync field set value: 0y0 = asynchronous write mode 0y1 = Reserved
[5:3]	set_rd_bl	WO	–	Read burst length 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[2]	set_rd_sync	WO	–	Holding register of the rd_sync field set value: 0y0 = asynchronous read mode 0y1 = Reserved
[1:0]	set_mw	WO	–	Holding register of the memory data bus width set value: 0y00 = reserved 0y01 = 16-bits 0y10 = Reserved 0y11 = Reserved

Note: This register cannot be written while it is in the Reset state.

This is a holding register for enabling setting values. By executing of the following operations, the settings values of this register will be updated to the configuration register of the memory manager and enabled.

・ The smc_direct_cmd Register indicates only a register update is taking place.

[Description]

a. <set_burst_align>

For asynchronous transfers:
When set_rd_sync = 0, MPMC1 always aligns read bursts to the memory burst boundary.
When set_wr_sync = 0, MPMC1 always aligns write bursts to the memory burst boundary.

b. < set_bls >

Byte Enable (SMCBE0-1) timing setting:
0y0 = SMCCSn timing
0y1 = SMCWEn timing

c. < set_wr_bl >

Write burst length
0y000 = 1 beat
0y001 = 4 beats
other = Reserved

d. < set_wr_sync >

Write synchronization mode setting:
0y0 = asynchronous write mode
0y1 = Reserved

e. < set_rd_bl >

Read burst length
0y000 = 1 beat
0y001 = 4 beats
other = Reserved

f. < set_rd_sync >

Read synchronization mode setting:
0y0 = asynchronous read mode
0y1 = Reserved

g. < set_mw >

Holding register of the memory data bus width set value:
0y00 = Reserved
0y01 = 16 bits
0y10 = Reserved
0y11 = Reserved

4. smc_sram_cycles0_0_5 (SMC SRAM Cycles Registers 0 <0>)

Address = (0xF431_1000) + (0x0100)

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

[Description]

a. <t_tr>
Turnaround time for SRAM chip configuration
0y000 to 0y111

b. <t_ pc>
Page cycle time:
0y000 to 0y111

c. <t_wp>
Delay time for smc_we_n_0:
0y000 to 0y111

d. <t_ceoe>
Delay time for smc_oe_n_0:
0y000 to 0y111

e. <t_wc>
Write cycle time
0y0000 to 0y1111

f. <t_rc>
Read cycle time
0y0000 to 0y1111

- smc_sram_cycles0_x_5 (SMC SRAM Cycles Registers 0 <x>) (x = 0 to 3)
  The structure and description of these registers are same as smc_sram_cycles0_0_5. Please refer to the description of smc_sram_cycles0_0_5.
  For the name and address of these registers, please refer to Table 3.10.13 MPMC1 SMC SFR list.

5. smc_opmode0_0_5 (SMC Opmode Registers 0 <0>)

Address = (0xF431_1000) + (0x0104)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	Reserved	RO	0x20	Read as 0x20.
[23:16]	Reserved	RO	0xE0	Read as 0xE0.
[15:13]	burst_align	RO	0y000	Memory burst boundary split set value: 0y000 = bursts can cross any address boundary 0y001 = split at the 32-beat burst boundary 0y010 = split at the 64-beat burst boundary 0y011 = split at the 128-beat burst boundary 0y100 = split at the 256-beat burst boundary other = Reserved
[12]	bls	RO	0y0	bls timing : 0y0 = chip select 0y1 = Reserved
[11]	Reserved	RO	0y1	–
[10]	–	–	Undefined	Read as undefined.
[9:7]	wr_bl	RO	0y000	Write memory burst length: 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[6]	wr_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous write operation 0y1 = Reserved
[5:3]	rd_bl	RO	0y000	Read memory burst length: 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[2]	rd_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous read operation 0y1 = Reserved
[1:0]	mw	RO	0y10	Memory data bus width : 0y00 = Reserved 0y01 = 16-bits 0y10 = Reserved 0y11 = Reserved

6. smc_opmode0_1_5 (SMC Opmode Registers 0 <1>)

Address = (0xF431_1000) + (0x0124)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	Reserved	RO	0x60	Read as 0x60.
[23:16]	Reserved	RO	0xE0	Read as 0xE0.
[15:13]	burst_align	RO	0y000	Memory burst boundary split set value: 0y000 = bursts can cross any address boundary 0y001 = split at the 32-beat burst boundary 0y010 = split at the 64-beat burst boundary 0y011 = split at the 128-beat burst boundary 0y100 = split at the 256-beat burst boundary other = Reserved
[12]	bls	RO	0y0	bls timing : 0y0 = chip select 0y1 = Reserved
[11]	Reserved	RO	0y1	–
[10]	–	–	Undefined	Read as undefined.
[9:7]	wr_bl	RO	0y000	Write memory burst length: 0y000 = 1-beat 0y001 = 4-beats other = Reserved
[6]	wr_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous write operation 0y1 = Reserved
[5:3]	rd_bl	RO	0y000	Read memory burst length: 0y000 = 1 beat 0y001 = 4 beats other = Reserved
[2]	rd_sync	RO	0y0	Memory operation mode: 0y0 = asynchronous read operation 0y1 = Reserved
[1:0]	mw	RO	0y10	Memory data bus width : 0y00 = Reserved 0y01 = 16-bits 0y10 = Reserved 0y11 = Reserved

[Description]

a. <burst_align>
Memory burst boundary split set value:
0y000 = bursts can cross any address boundary
0y001 = split at the 32-beat burst boundary
0y010 = split at the 64-beat burst boundary
0y011 = split at the 128-beat burst boundary
0y100 = split at the 256-beat burst boundary
Other = Reserved

b. <bls>
Bls timing :
0y0 = chip select
0y1 = Reserved

c. <wr_bl>
Write memory burst length:
0y000 = 1beat
0y001 = 4beats
Other = Reserved

d. <wr_sync>
Memory operation mode:
0y0 = asynchronous write operation
0y1 = Reserved

e. <rd_bl>
Read memory burst length:
0y000 = 1 beat
0y001 = 4 beats
Other = Reserved

f. <rd_sync>
Memory operation mode:
0y0 = asynchronous read operation
0y1 = Reserved

g. <mw>
The Reset value depends on setting state. The CS0 memory data bus width can be set for Boot.

3.11 NAND-Flash Controller (NDFC)

3.11.1 Overview

The NAND-Flash Controller (NDFC) is provided with dedicated pins for connecting with the NAND-Flash memory. The NDFC also has an ECC calculation function for error correction. It supports the Hamming Code ECC calculation method for the NAND-Flash memory of SLC (Single Level Cell) type that is capable of correction (Note1) a single-bit error for every 256 bytes and the Reed-Solomon ECC calculation method for the NAND-Flash memory of MLC (Multi-Level Cell) type that is capable of correction (Note 1) four error addresses for every 512 bytes.

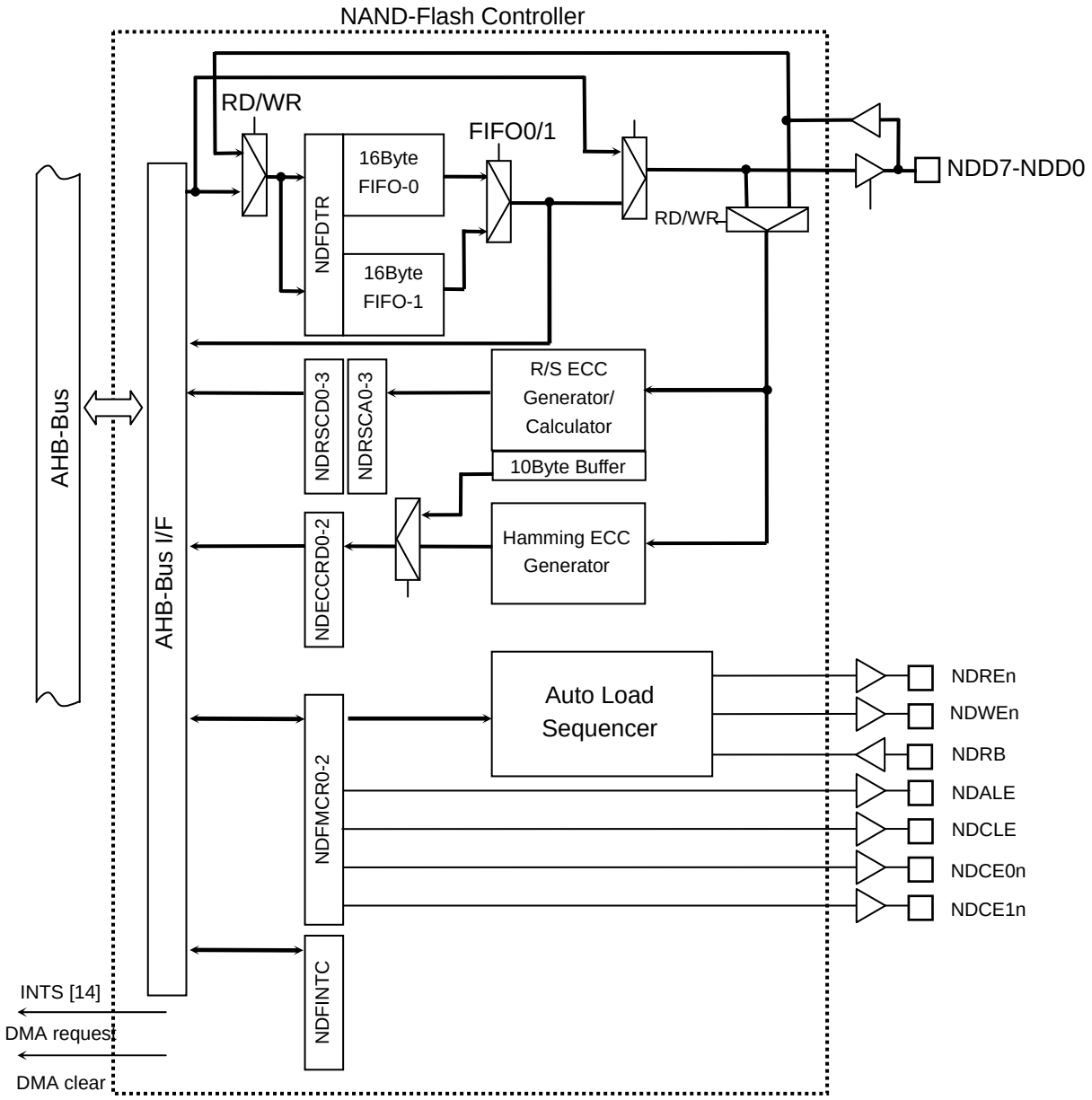
The NDFC has the following features:

a. Controls the NAND-Flash memory interface through registers.
b. Supports 8-bit NAND-Flash memory devices (Does not support 16-bit devices).
c. Supports page sizes of 512 bytes and 2048 bytes.
d. Includes an ECC generation circuit using Hamming codes (for SLC type).
e. Includes a 4-address (4-byte) error detection circuit using Reed-Solomon coding/encoding techniques (for MLC type).
f. Provides an Autoload function for high-speed data transfer by using two 32bit × 4word FIFOs together with a DMA controller.

Note 1: Error correction needs software processing.

Note 2: The WPn (Write Protect) pin of the NAND Flash is not supported. When this function is needed, prepare it on an external circuit.

3.11.2 Block Diagram



3.11.3 Operation Description

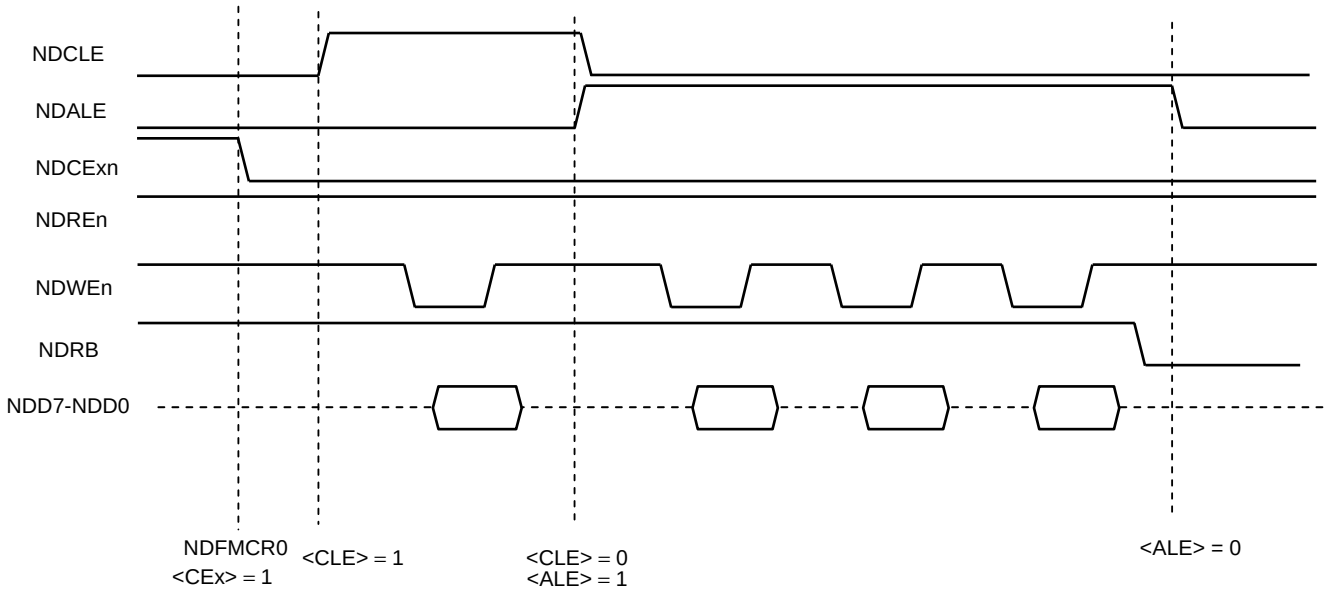
a. Setting the commands and addresses to the NAND-Flash memory

The commands and addresses for executing instructions such as Page Read and Page Write to the NAND-Flash memory are set by software.

The NDCExn, NDCLE, and NDALE pins are configured by the NDFMCR0<CEx, CLE, ALE> register.

Reading/Writing of NAND-Flash memory are executed by reading/writing the NDFDTR register.

The setting the AC timing can be adjusted by using the NDFMCR2 <SPLW[2:0], SPHW[2:0], SPLR[2:0], SPHR[2:0]> register.



b. Reading data from the NAND-Flash memory in page units and writing data to the built-in RAM

In this section, a high-speed data read function with a smaller burden to the CPU is implemented by using the built-in DMA controller in addition to two 32bit × 4word FIFOs contained in the NDFC and the Autoload function.

Note: Please use the DMA function for the data read that uses the Autoload function.

Because the Autoload function at data read starts automatically after detection of a rising edge of the NDRB pin in the state of NDFMCR1<ALS>= 1, the settings of steps (1) and (2) below must be performed after command setting to the NAND-Flash before address setting.

(1) Assign the NDFC to an arbitrary channel of the DMA controller and set the relevant registers.

The following is an example in which the NDFC is assigned to DMAC channel 0:

DMACC0SrcAddr ← Address of NDFDTR
DMACC0DestAddr ← Address of the built-in RAM
DMACC0Control ← <Swidth[2:0]> = 0y010 (32 bits),
<Dwidth[2:0]> = 0y010 (32 bits),
<SBSize[2:0]> = 0y001 (4 beats),
<TransferSize[11:0]> = 0x80 (512 Bytes/ 4 Bytes)
DMACC0Configuration ← <FlowCntrl[13:11]> = 0y010 (Peripheral to Memory),
<ITC> = 1 (DMA termination interrupt is enabled.)

(2) Write 0 to the NDFMCR1<SELAL> register and 1 to the NDFMCR1<ALS> register.

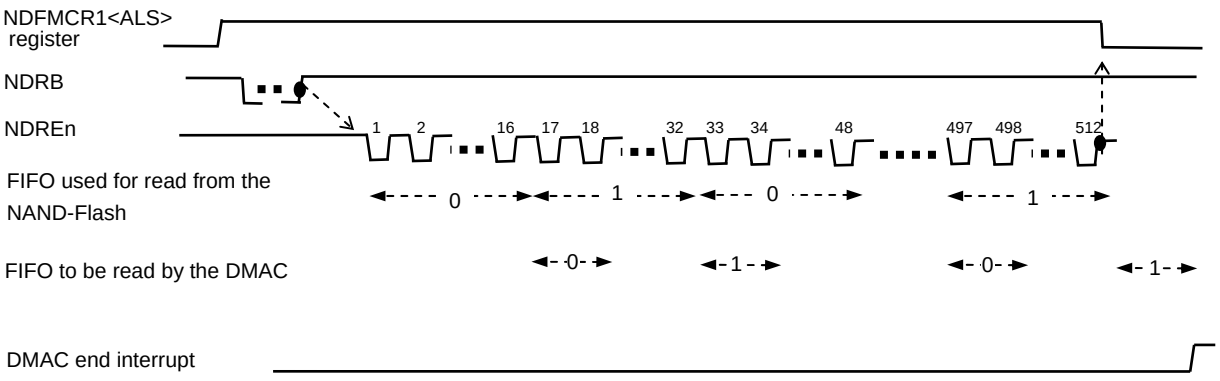
When Step (2) is performed, the NDFC begins detecting a rising edge of the R/B pin, and after detecting a rising edge, starts a read cycle of 1-byte data. Each time the NDFC reads 1-byte data, it stores the read data in the first-stage 16-byte FIFO (FIFO-0) and generates the ECC by entering the data to either Hamming Code ECC calculator or Reed-Solomon ECC calculator depending on the setting of the NDFMCR1<ECCS> register.

When FIFO-0 is filled up with data, the FIFO-1) takes over the data storage for continued data read. In addition, the NDFC asserts a DMA transfer request to the DMAC at the fill-up of FIFO-0 to request the transfer of the FIFO-0 data to the built-in RAM.

Data can be read efficiently at a higher speed by switching between two 16-byte FIFOs in this way.

When a total of 512 bytes of data has been read, the DMAC asserts a DMA termination interrupt and the CPU uses the interrupt to start the next process.

The following shows a conceptual timing chart of the data read timing by DMA.



If DMAC cannot read the data from a FIFO of the NDFC when both FIFO-0 and FIFO-1 are full, the autoload function is suspended for that duration.

c. Data writing from the built-in RAM to the NAND-Flash memory in page units

The following is a description of data writing using the Autoload function that is performed similarly to data reading from the NAND-Flash. For execution, perform the settings described in steps (1) and (2) below.

Note: Please use the DMA function for the data read that uses the autoload function.

(1) Assign the NDFC to an arbitrary channel of the DMA controller and set the relevant registers.
The following is an example in which the NDFC is assigned to DMAC channel 0:

DMACC0SrcAddr ← Address of the built-in RAM
DMACC0DestAddr ← Address of NDFDTR
DMACC0Control ← <Swidth[2:0]> = 0y010 (32 bits),
<Dwidth[2:0]> = 0y010 (32 bits),
<SBSize[2:0]> = 0y001 (4 beats),
<TransferSize[11:0]> = 0x80 (512 Byte/4 Byte)
DMACC0Configuration ← <FlowCntrl> = 0y001(Memory to Peripheral),
<ITC> = 1 (DMA termination interrupt is enabled.)

(2) Write 1 to the NDFMCR1<SELAL> register and 1 to the NDFMCR1<ALS> register.

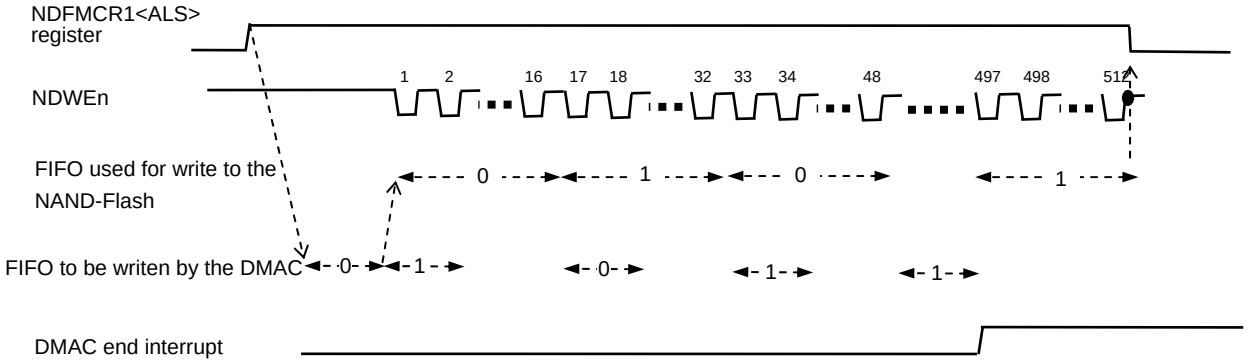
When step (2) is performed, the NDFC asserts a DMA request, because both FIFO-0 and FIFO-1 are empty, to have the DMA controller transfer data from the built-in RAM to FIFO-0 and FIFO-1.

When the data transfer from the DMA controller to FIFO-0 and FIFO-1 is terminated, the NDFC uses the FIFO-0 data to start a data write cycle to NAND-Flash. Each time the NDFC writes data, it generates the ECC by entering the data to either Hamming Code ECC calculator or Reed-Solomon ECC calculator depending on the setting of the NDFMCR1<ECCS> register. When FIFO-0 becomes empty, the FIFO-1 takes over the data extraction for continued data write.

In addition, the NDFC asserts a DMA transfer request to the DMAC at the time of FIFO-0's becoming empty to request the data transfer from the built-in RAM to FIFO-0.

Data can be written efficiently at a higher speed by switching between two 16-byte FIFOs in this way.

When a total of 512 bytes of data has been written, the DMAC asserts a DMA termination interrupt and the CPU uses the interrupt to start the next process. The following shows a conceptual timing chart of the data write timing by DMA.



Note: Write operation to the NAND-Flash memory is not terminated by the Autoload function of the NDFC at the time of assertion of a DMAC end interrupt. Ensure that the NDFMCR1<ALS> = 0 during the DMAC end interrupt processing and then execute the next process (processing of the ECC).

If DMAC cannot be write the data to FIFO of the NDFC when FIFO-0 and FIFO-1 are full, the Autoload function is suspended for that duration.

d. ECC read or write to or from the redundant area

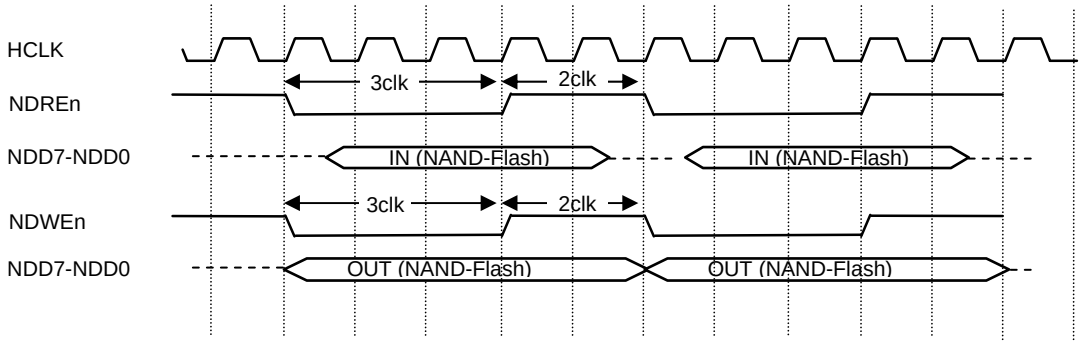
The Autoload function cannot be used. Execute read or write to or from NAND-Flash by software as in the case of setting a command or address.

e. Waveform adjusting function for NDREn and NDWEn

When setting of a command and address, data read, or data write is performed to the NDFDTR register, the NDFC generates waveforms for the NDREn and NDWEn pins.

At this time, the Low width and High width for the NDREn and NDWEn pins can be adjusted. Adjustment should be done in accordance with the AC specifications including the NDFC operation clock, HCLK (up to 100 MHz) and the NAND-Flash access time. (For details, refer to the electrical characteristics.)

The following figure shows a timing chart example in which continuous accesses are made when NDFMCR2<SPLW[2:0]> = 0y011, NDFMCR2<SPLR[2:0]> = 0y011, NDFMCR2<SPHW[2:0]> = 0y010, and NDFMCR2<SPHR[2:0]> = 0y010. (The data drive time becomes longer at data write.)



3.11.4 ECC Control

This section describes ECC control. NAND-Flash memory devices may inherently include error bits. It is therefore necessary to implement the error correction processing using ECC (Error Correction Code).

Figure 3.11.1 shows a basic flowchart for ECC control.

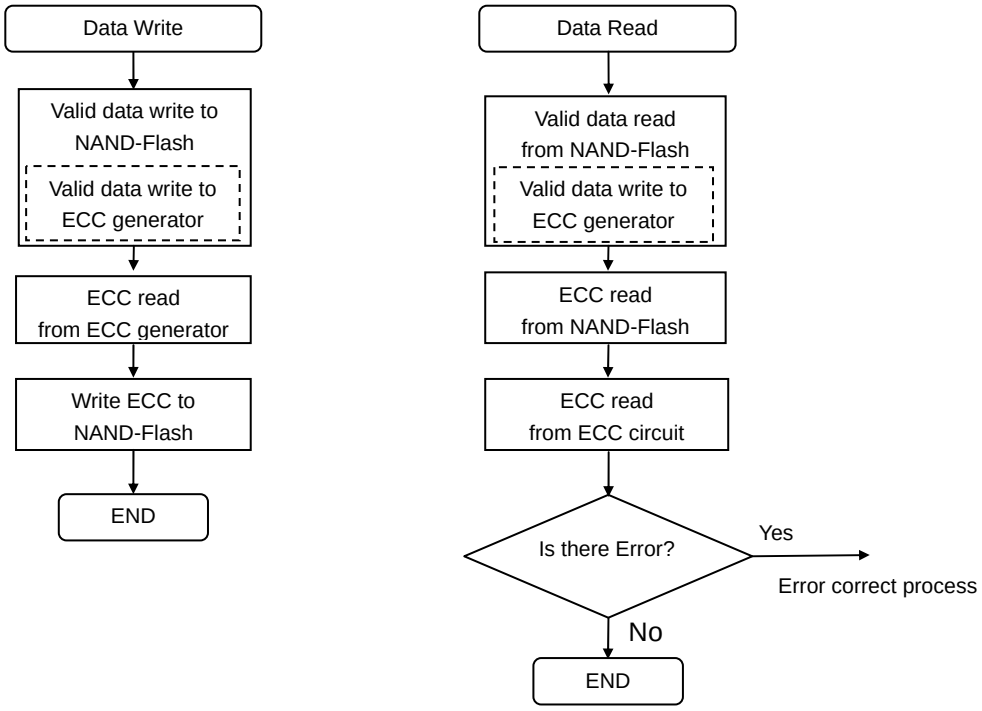


Figure 3.11.1 Basic Flow of ECC Control

Write:

1. When data is written to the actual NAND-Flash memory, the ECC generator in the NDFC simultaneously generates ECC for the written data.
2. The ECC is written to the redundant area in the NAND-Flash separately from the valid data.

Read:

1. When data is read from the NAND-Flash memory, the ECC generator in the NDFC simultaneously generates ECC for the read data as in the case of data writing.
2. The ECC written to the redundant area from the NAND-Flash memory is read. The ECC at the time of data writing and that at the time of data reading are used to calculate error bits for correction.

3.11.4.1 Difference between Hamming Code ECC Calculation Method and Reed-Solomon ECC Calculation Method

The NDFC includes an ECC generator supporting 2LC and 4LC.

The ECC calculation using Hamming codes (supporting 2LC) generates 22 bits of ECC for every 256 bytes of valid data and is capable of detecting and correcting a single-bit error for every 256 bytes. Error bit detection calculation and correction must be implemented by software.

When using Smart Media, Hamming codes should be used.

The ECC calculation using Reed-Solomon codes (supporting 4LC) generates 80 bits of ECC for every 1 byte to 512 bytes of valid data and is capable of detecting and correcting error bits at 4-symbol for every 512 bytes. Although the Reed-Solomon ECC calculation method needs error bit correction to be implemented by software as in the case of the Hamming Code ECC calculation method, error bit detection calculation is supported by hardware.

The differences between Hamming Code ECC calculation method and Reed-Solomon ECC calculation method are summarized in Table 3.11.1

Table 3.11.1 Differences between Hamming Code ECC Calculation Method and Reed-Solomon ECC Calculation Method

	Hamming	Reed Solomon
Maximum number of correctable errors	1-bit	4-symbol (All the 8 bits at one symbol are correctable.)
Number of ECC bits	22 bits/256 bytes	80 bits / up to 512 bytes
Error bit detection method	Supported by software.	Detected by hardware.
Error bit correction method	Supported by software.	Supported by software.
Error bit detection time	Supported by software, so it depends on how the software is made.	See the table below.
Others	Supports SmartMedia.	−

Number of Error Bits	Reed-Solomon Error Bit Detection Time (Units: HCLK)	Remarks
4	813 (max)	These values indicate the total number of clocks for detecting error bit(s) but do not include the register read/write time by the CPU.
3	648 (max)	
2	358 (max)	
1	219 (max)	
0	1	

3.11.4.2 Error Correction Methods

Hamming ECC

- The ECC generator generates 44 bits of ECC for a page containing 512 bytes of valid data. The error correction process must be performed in units of 256 bytes (22 bits of ECC). The following explains how to implement error correction on 256 bytes of valid data using 22 bits of ECC.
- If the NAND-Flash memory to be used has a large-capacity page size (e.g. 2048 bytes), the error correction process must be repeated several times to cover the entire page.

1. The calculated ECC and the ECC in the redundant area (Note 1) are rearranged, so that the lower 2 bytes of each ECC represent line parity (LPR15: 0) and the upper 1 byte (of which the upper 6 bits are valid) represents column parity (CPR7: 2).
2. The two rearranged ECCs are XORed.
3. If the XOR result is 0, indicating an ECC match, the error correction process ends normally (no error). If the XOR result is other than 0, it is checked whether or not the error data can be corrected.
4. If the XOR result contains only one ON bit, it is determined that a single-bit error exists in the ECC data itself and the error correction process terminates here (error not correctable).
5. If every two bits of bits 0 to 15 and bits 18 to 23 of valid data in the XOR result are either 0y01 or 0y10, it is determined that the error data is correctable and error correction is performed accordingly. If the XOR result contains either 0y00 or 0y11, it is determined that the error data is not correctable and the error correction process terminates abnormally.

	Example of Correctable XOR Result	Example of Uncorrectable XOR Result
Binary	10 01 10 00 Column parity 10 10 01 10 Line parity 01 01 10 10	10 11 10 00 Column parity 10 10 01 10 Line parity 01 01 10 10

6. For correction of the data, the line information in error is created from the line parity of the XOR result and the bit information is created from the column parity and then the error bit is inverted. The error correction is now completed.

Example: When the XOR result is 0y10_01_10_00_10_10_01_10_01_01_10_10

Convert two bytes of line parity into one byte. (10→1, 01→0)

Convert six bits of column parity into three bits. (10→1、01→0)

Line parity: 10 10 01 10 01 01 10 10

⇩ ⇩ ⇩ ⇩ ⇩ ⇩ ⇩ ⇩

1 1 0 1 0 0 1 1 = 0xD3

In this case, an error exists at address 0xD3. Note that this address is not an absolute address but a relative address in 256 bytes. Due care must be used when correcting this error.

Column parity: 10 01 10

⇩ ⇩ ⇩

1 0 1 = 5 → Error in bit 5

Error correction is performed by inverting the data in bit 5 at address 0xD3.

Reed-Solomon ECC

- The ECC generator generates 80 bits of ECC for up to 518 bytes of valid data. If the NAND-Flash memory to be used has a large-capacity page size (e.g. 2048 bytes), the error correction process must be repeated several times to cover the entire page.
- Basically no calculation is needed for error correction. If error detection is performed properly, the NDFC only needs to refer to the error address and error bit. However, it may be necessary to convert the error address, as explained below.

1. If the error address indicated by the NDRSCAn register is in the range of 0x000 to 0x007, this error exists in the ECC area and no correction is needed in this case. (It is not able to correct the error in the ECC area. Note, however, that if the error exists in the ECC area, this LSI has only the ability to correct errors up to 4 symbols including the error in the ECC area.
2. If the error address indicated by the NDRSCAn register is in the range of 0x008 to 0x207, the error address is obtained by subtracting this address from 0x207.

Example 1: When NDRSCAn = 0x005 and NDRSCDn = 0x04 = 0y0000_0100

Because the error address is in the range of 0x000 to 0x007, no correction is needed.

(Although an error exists in bit 2, no correction is needed.)

Example 2: When NDRSCAn = 0x083 and NDRSCDn = 0x81 = 0y1000_0001

Error correction is performed by inverting the data in bits 7 and 0 at address 0x184 (0x207 – 0x083).

Note : If the error address (after conversion) is in the range of 0x000 to 0x007, it indicates that an error bit exists in redundant area (ECC). In this case, no error correction is needed. If the number of error bits is not more than 4 symbols, the Reed-Solomon ECC calculation method calculates each error bit precisely even if it is in the redundant area (ECC).

3.11.5 Description of Registers

The following lists the SFRs:

Table 3.11.2 SFR List

Base address = 0xF201_0000

Register Name	Address (base+)	Description
NDFMCR0	0x0000	NAND-Flash Control Register 0
NDFMCR1	0x0004	NAND-Flash Control Register 1
NDFMCR2	0x0008	NAND-Flash Control Register 2
NDFINTC	0x000C	NAND-Flash Interrupt Control Register
NDFDTR	0x0010	NAND-Flash Data Register
NDECCRD0	0x0020	NAND-Flash ECC Read Register 0
NDECCRD1	0x0024	NAND-Flash ECC Read Register 1
NDECCRD2	0x0028	NAND-Flash ECC Read Register 2
NDRSCA0	0x0030	NAND-Flash Reed-Solomon Calculation Result Address Register 0
NDRSCD0	0x0034	NAND-Flash Reed-Solomon Calculation Result Data Register 0
NDRSCA1	0x0038	NAND-Flash Reed-Solomon Calculation Result Address Register 1
NDRSCD1	0x003C	NAND-Flash Reed-Solomon Calculation Result Data Register 1
NDRSCA2	0x0040	NAND-Flash Reed-Solomon Calculation Result Address Register 2
NDRSCD2	0x0044	NAND-Flash Reed-Solomon Calculation Result Data Register 2
NDRSCA3	0x0048	NAND-Flash Reed-Solomon Calculation Result Address Register 3
NDRSCD3	0x004C	NAND-Flash Reed-Solomon Calculation Result Data Register 3

1. NDFMCR0 (NAND-Flash Control Register 0)

Address = (0xF201_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read as undefined. Write as zero.
[11]	RSECCL	R/W	0y0	Reed Solomon ECC-Latch 0y0: Disable (Enable 80-bit F/F update.) 0y1: Enable (Disable 80-bit F/F update.)
[10]	RSEDN	R/W	0y0	Reed-Solomon operation select 0y0: Read 0y1: Write
[9]	RSESTA	WO	0y0	Reed-Solomon error calculation start 0y0: – 0y1: Start
[8]	RSECGW	R/W	0y0	Reed-Solomon ECC-Generator write enable 0y0: Disable 0y1: Enable
[7]	WE	R/W	0y0	Write operation enable 0y0: Disable 0y1: Enable
[6]	ALE	R/W	0y0	NDALE pin control 0y0: Output 0 0y1: Output 1
[5]	CLE	R/W	0y0	NDCLE pin control 0y0: Output 0 0y1: Output 1
[4]	CE0	R/W	0y0	NDCE0n pin control 0y0: Output 1 0y1: Output 0
[3]	CE1	R/W	0y0	NDCE1n pin control 0y0: Output 1 0y1: Output 0
[2]	ECCE	R/W	0y0	ECC circuit enable 0y0: Disable 0y1: Enable
[1]	BUSY	RO	0y0	NAND-Flash status Read: 0y0: Ready 0y1: Busy Write: Invalid
[0]	ECCRST	WO	0y0	ECC circuit reset 0y0: – 0y1: Reset

[Description]

a. <RSECCL>

The <RSECCL> bit is used only for Reed-Solomon codes. When Hamming codes are used, this bit should be set to 0.

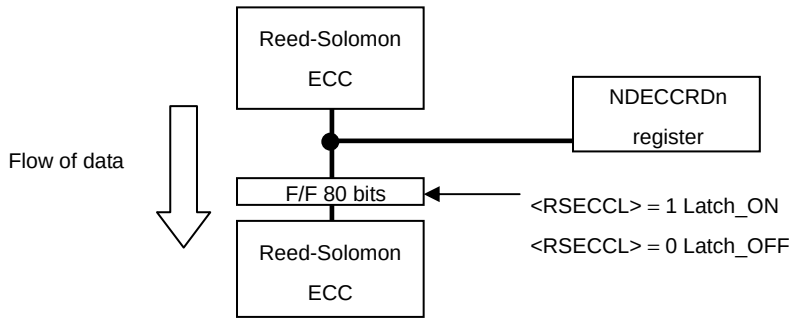
The Reed-Solomon processing unit is comprised of two circuits: an ECC generating circuit and a circuit to calculate the error address and error bit position from the ECC.

No special care is needed if ECC generation and error calculation are performed in series. If these operations need to be performed in parallel, the intermediate code used for error calculation must be latched while the calculation is being performed.

The <RSECCL> bit is provided to enable the latch operation for the intermediate code generated from the ECC for written data and the ECC for read data to calculate the error address and error bit position.

When <RSECCL> is set to 1, the intermediate code is latched so that no ECC is transferred to the error calculator even if the ECC generator updates the ECC, thus allowing the ECC generator to generate the ECC for another page while the ECC calculator is calculating the error address and error bit position. At this time, the ECC generator can perform both Write and Read operations.

When <RSECCL> is set to 0 the latch is released and the contents of the ECC calculator are updated sequentially as the data in the ECC generator is updated.



b. <RSEDN>

The <RSEDN> bit is used only for Reed-Solomon codes. When Hamming codes are used, this bit should be set to 0.

For a write operation, this bit should be set to 1 (for write) to generate ECC. The ECC read from the NDECCRDn register is written to the redundant area of the NAND-Flash memory. For a read operation, this bit should be set to 0 (for Read). Then, valid data is read from the NAND-Flash memory and the ECC written in the redundant area of the NAND-Flash memory is read to generate an intermediate code for calculating the error address and error bit position.

c. <RSESTA>

The <RSESTA> bit is used only for Reed-Solomon codes.

The error address and error bit position are calculated using an intermediate code generated from the ECC for written data and the ECC for read data Writing 1 to <RSESTA> starts this calculation.

d. <RSECGW>

The <RSECGW> bit is used only for Reed-Solomon codes. When Hamming codes are used, this bit should be set to 0.

Since the valid data part and the ECC are processed differently in this circuit, reading the valid data part and reading the ECC should be managed separately by software.

To read valid data from the NAND-Flash memory, set <RSECGW> to 0 (Disable). To read the ECC written in the redundant area of the NAND-Flash, set <RSECGW> to 1 (Enable).

Note 1: Valid data that use the DMAC and ECC cannot be read continuously. After valid data has been read, data transfer should be stopped once to change the <RSECGW> bit from 0 to 1 before ECC is read.

Note 2: Immediately after ECC is read from the NAND-Flash memory, accesses (read/write) to the NAND-Flash memory or error bit calculation cannot be performed for a duration of 20 system clocks that is used for internal processing. Wait processing or other by software is needed.

e. <WE>

The <WE> bit is used for both Hamming and Reed-Solomon codes. This bit is used to control activation of the NDWEn pin. This is a protective register to prevent the NDWEn pin from being activated inadvertently for the NAND-Flash memory.

f. <ALE>, <CLE>, <CE0>, <CE1>

The <ALE>, <CLE>, <CE0> and <CE1> bits are used for both Hamming and Reed-Solomon codes. These pins are used to control the pins of the NAND-Flash memory.

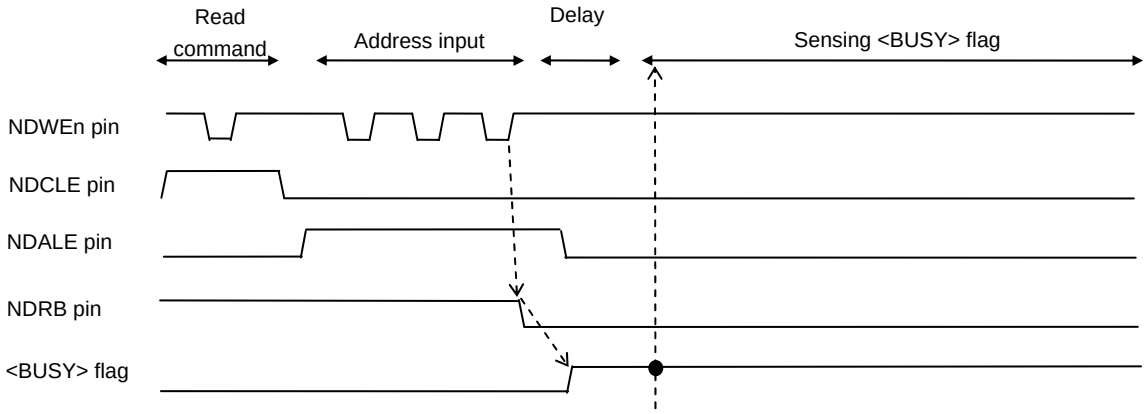
g. <ECCE>

The <ECCE> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to control the ECC circuit. To reset the ECC (to write 1 to <ECCRST>), this bit must have been enabled (1).

h. <BUSY>

The <BUSY> bit is used for both Hamming and Reed-Solomon codes. This bit is used to check the state of the NAND-Flash memory (NDRB pin). It is set to 1 when the NAND-Flash is “busy” and to 0 when it is “ready”. Since the NDFC incorporates a noise filter of several clocks, a change in the NDR/B pin state is reflected on the <BUSY> flag after some delay.



i. <ECCRST>

The <ECCRST> bit is used for both Hamming and Reed-Solomon codes.

To reset the Hamming ECC, set NDFMCR1<ECCS> = 0 (to reset the Reed-Solomon ECC, set NDFMCR1<ECCS> = 1), then write 1 once to this bit, and the ECC in this circuit is reset (reset is released automatically). The contents of the NDECCRDn register are also reset at the same time.

When you reset ECC, set <ECCE> to 1.

2. NDFMCR1 (NAND-Flash Control Register 1)

Address = (0xF201_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:12]	STATE[3:0]	RO	0y0000	Read: Status of the Reed-Solomon ECC calculator (Valid after calculation processing is started.) 0y0000: Calculation ended with no error. 0y0001: Calculation ended with errors of more than 5 symbols (uncorrectable). 0y0010, 0y0011: Calculation ended with errors of 4 symbols or less (correctable). 0y0100-0y1111: Calculating Write: Invalid
[11:10]	SERR[1:0]	RO	Undefined	Read: Number of errors in the Reed-Solomon ECC calculator (Valid after calculation processing ended.) 0y00: 1-address error 0y01 : 2-address error 0y10: 3-address error 0y11: 4-address error Write: Invalid
[9]	SELAL	R/W	0y0	Autoload function select 0y0: Data read from the NAND-Flash 0y1: Data write to the NAND-Flash
[8]	ALS	R/W	0y0	Autoload start (at write time) 0y0: – 0y1: Start Autoload status (at read time) 0y0: Before or after execution 0y1: Being executed
[7:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	ECCS	R/W	0y0	ECC circuit select 0y0: Hamming 0y1: Reed-Solomon
[0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <STATE[3:0]>, <SERR[1:0]>,

The <STATE3:0> and <SEER1:0> bits are used only for Reed-Solomon codes. When Hamming codes are used, these bits have no meaning.

These bits are used as flags to indicate the states of error address and error bit calculation results.

b. <SELAL>

The <SELAL> bit is used for both Hamming and Reed-Solomon codes.

This bit is used to select data read or data write for the NAND-Flash when the Autoload function is executed.

c. <ALS>

The <ALS> bit is used for both Hamming and Reed-Solomon codes.

This is the register that controls the function to transfer data read/write for the NAND-Flash at high speed by using the DMAC. Writing 1 to <ALS> enables the 16-byte FIFO0/FIFO1.

In addition, a read operation allows the user to know the status of the Autoload function. (When 512-byte read or write is executed by the Autoload function, this register is cleared to 0.)

d. <ECCS>

The <ECCS> bit is used to select whether to use Hamming codes or Reed-Solomon codes. This bit is set to 0 for using Hamming codes and to 1 for using Reed-Solomon codes. It is also necessary to set this bit for resetting ECC.

3. NDFMCR2 (NAND-Flash Control Register 2)

Address = (0xf201_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:15]	–	–	Undefined	Read as undefined. Write as zero.
[14:12]	SPLW	R/W	0y000	NDWEn Low pulse width setting 0y000: Reserved 0y001: 1 cycle of HCLK 0y010: 2 cycles of HCLK 0y011: 3 cycles of HCLK 0y100: 4 cycles of HCLK 0y101: 5 cycles of HCLK 0y110-0y111: Reserved
[11]	–	–	Undefined	Read as undefined. Write as zero.
[10:8]	SPHW	R/W	0y000	NDWEn High pulse width setting 0y000: Reserved 0y001: 1 cycle of HCLK 0y010: 2 cycles of HCLK 0y011: 3 cycles of HCLK 0y00: 4 cycles of HCLK 0y101: 5 cycles of HCLK 0y110-0y111: Reserved
[7]	–	–	Undefined	Read as undefined. Write as zero.
[6:4]	SPLR	R/W	0y000	NDREn Low pulse width setting 0y000: Reserved 0y001: 1 cycle of HCLK 0y010: 2 cycles of HCLK 0y011: 3 cycles of HCLK 0y100: 4 cycles of HCLK 0y101: 5 cycles of HCLK 0y110-0y111: Reserved
[3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	SPHR	R/W	0y000	NDREn High pulse width setting 0y000: Reserved 0y001: 1 cycle of HCLK 0y010: 2 cycles of HCLK 0y011: 3 cycles of HCLK 0y100: 4 cycles of HCLK 0y101: 5 cycles of HCLK 0y110-0y111: Reserved

[Description]

a. <SPLW>, <SPHW>, <SPLR>, <SPHR>

These are registers to set the Low and High pulse width of the NDREn and NDWEn pins.

The pulse width is given by the set value × the period of HCLK. Setting 0y000, 0y110 and 0y111 are prohibited.

4. NDFINTC (NAND-Flash Interrupt Control Register)

Address = (0xF201_0000) + (0x000C)

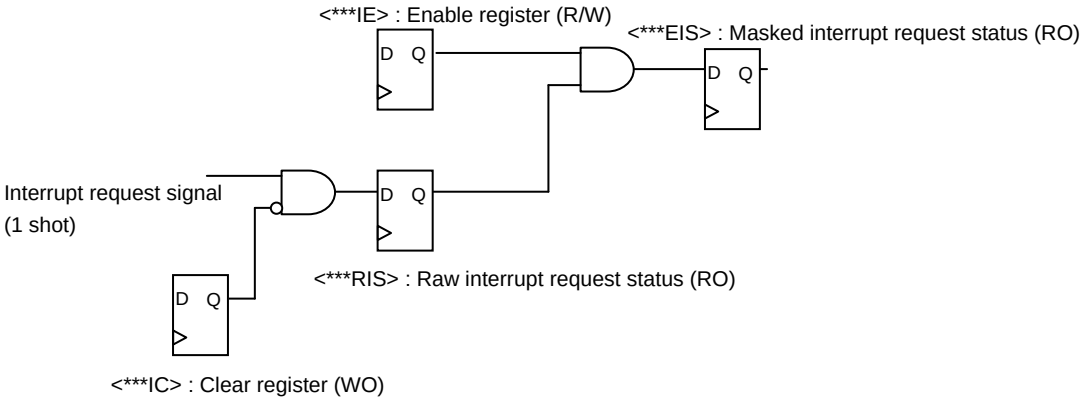
Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	RSEIC	WO	0y0	Read: Reed-Solomon calculator end interrupt clear register 0y0: – 0y1: Clear Write: Invalid
[6]	RSEEIS	RO	0y0	Read: Reed-Solomon calculator end interrupt masked status 0y0: Interrupt not requested 0y1: Interrupt requested Write: Invalid
[5]	RSERIS	RO	0y0	Read: Reed-Solomon calculator end interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested Write: Invalid
[4]	RSEIE	R/W	0y0	Reed-Solomon calculator end interrupt enable register 0y0: Disable interrupt requests 0y1: Enable interrupt requests
[3]	RDYIC	WO	0y0	NAND-Flash ready interrupt clear register 0y0: – 0y1: Clear
[2]	RDYEIS	RO	0y0	Read: NAND-Flash ready interrupt masked status 0y0: Interrupt not requested 0y1: Interrupt requested Write: Invalid
[1]	RDYRIS	RO	0y0	Read: NAND-Flash ready interrupt raw status 0y0: Interrupt not requested 0y1: Interrupt requested Write: Invalid
[0]	RDYIE	R/W	0y0	NAND-Flash ready interrupt enable register 0y0: Disable interrupt requests 0y1: Enable interrupt requests

[Description]

a. <xxxIC>, <xxxEIS>, <xxxRIS>, <xxxIE>

These are 4-bit registers to support two types of interrupts: a READY interrupt that occurs when the status of the monitored NDRB pin changes from Busy to Ready and a Reed-Solomon calculation end interrupt that occurs when Reed-Solomon calculation of address and data ends. The NDFC asserts one interrupt request obtained by ORing these two interrupt requests to the interrupt controller. Therefore, the register contents check during interrupt processing and the processing appropriate to the individual interrupt source are required.

The following figure shows the relationship between these registers:



5. NDFDTR (NAND-Flash Data Register)

Address = (0xF201_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	DATA[31:0]	R/W	Undefined	Data register

[Description]

a. <DATA[31:0]>

This register is accessed when reading or writing data to or from the NAND-Flash memory or setting commands and addresses to the memory.

When data is written to this register, the data is written to the NAND-Flash memory. When a read operation is made to this register, data is read from the NAND-Flash memory. One-word transfer can be used through the DMA operation.

Note: Although this register is readable and writable, it contains no F/F. If reading is done after writing, the written data is not held because the operations for writing and reading are different.

6. NDECCRD0 (NAND-Flash ECC Read Register 0)

Address = (0xF201_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	CODE0[31:0]	RO	0x00000000	Register to store ECC

7. NDECCRD1 (NAND-Flash ECC Read Register 1)

Address = (0xF201_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	CODE1[31:0]	RO	0x00000000	Register to store ECC

8. NDECCRD2 (NAND-Flash ECC Read Register 2)

Address = (0xF201_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:0]	CODE2[15:0]	RO	0x0000	Register to store ECC

[Description]

a. <CODE0[31:0]>, <CODE1[31:0]>, <CODE2[15:0]>

This register is used to read the ECC calculated in this circuit.

When 0 is written to NDFMCR0<ECCE> after read/write ends, ECC is prepared in this register (when the value of NDFMCR0<ECCE> changes from 1 to 0 the ECC in this register is updated).

The Hamming ECC calculation generates 22 bits of ECC for every 256 bytes of valid data and the Reed-Solomon ECC calculation generates 80 bits of ECC for every 1 byte to 512 bytes of valid data.

Three 32-bit width registers are provided to store 80 bits.

The table below shows the format for storing ECC.

Note: Before reading ECC from the NAND Flash ECC register, be sure to set NDFMCR0<ECCE> to 0. The ECC in the NAND Flash ECC register is updated when NDFMCR0<ECCE> changes from 1 to 0. Also note that when the ECC in the ECC generator is reset by NDFMCR0<ECCRST>, the contents of this register are not reset.

Register Name	Hamming	Reed-Solomon
NDECCRD0<15:0>	[15:0] Line parity (for the first 256 bytes)	[15:0] R/S ECC 79:64
NDECCRD0<31:16>	[23:18] Column parity (for the first 256 bytes)	[31:16] R/S ECC 63:48
NDECCRD1<15:0>	[15:0] Line parity (for the second 256 bytes)	[15:0] R/S ECC 47:32
NDECCRD1<31:16>	[23:18] Column parity (for the second 256 bytes)	[31:16] R/S ECC 31:16
NDECCRD2<15:0>	Not used	[15:0] R/S ECC 15:0

The table below shows examples of writing ECC on the redundant area of the NAND-Flash memory.

When using Hamming codes with SmartMedia, the addresses of the redundant area are specified by the physical format of SmartMedia. For details, refer to the SmartMedia Physical Format Specifications.

	Reed-Solomon	NAND-Flash Address
NDECCRD0	[31:0] R/S ECC 79:48	Upper 8 bits [79:72]→ address 518 8 bits [71:64] → address 519 8 bits [63:56] → address 520 Lower 8 bits [55:48] → address 521
NDECCRD1	[31:0] R/S ECC 47:16	Upper 8 bits [47:40] → address 522 8 bits [39:32] → address 523 8 bits [31:24] → address 524 Lower 8 bits [23:16] → address 525
NDECCRD2	[15:0] R/S ECC 15:0	Upper 8 bits [15:8] → address 526 Lower 8 bits [7:0] → address 527

9. NDRSCA0 (NAND-Flash Reed-Solomon Calculation Result Address Register 0)

Address = (0xF201_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:10]	–	–	Undefined	Read as undefined.
[9:0]	AL	RO	0x000	Register to store Reed-Solomon error 0 address

10. NDRSCD0 (NAND-Flash Reed-Solomon Calculation Result Data Register 0)

Address = (0xF201_0000) + (0x0034)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7:0]	DATA	RO	0x00	Register to store Reed-Solomon error 0 data

[Description]

a. <AL>,<DATA>

If an error is found at only one address, the error address is stored in the NDRSCA0 register and the error data in the NDRSCD0 register. If errors are found at two addresses, the error addresses are stored in the NDRSCA0 and NDRSCA1 registers and the error data in the NDRSCD0 and NDRSCD1 registers. Valid error addresses are stored in this way when error bits are found at four or less addresses.

For the number of error addresses, read the contents of NDFMCR1<SEER[1:0]>.

- NDRSCAx (NAND-Flash Reed-Solomon Calculation Result Address Register-x) (x = 0 to 3)
- NDRSCDx (NAND-Flash Reed-Solomon Calculation Result Data Register-x) (x = 0 to 3)

As for the above registers, the structure and description are same as NDRSCA0 and NDRSCD0.

Please refer to the description of NDRSCA0 and NDRSCD0. About the name and address of registers, please refer to Table 3.11.2.

3.11.6 Examples of Accessing the NAND-Flash

The following example shows the example of NAND Flash memory accessing. This example is showing the set-up procedure. Please note that we do not warrant the operation shown below. Please use it as a guide in programming.

(1) Page write (2LC type)

```
——— Main Program ———
  ;
  ; ***** Initialize NDFC *****
  ;       Condition: 8bit-bus, CE0, SLC, 512 Byte/Page, Hamming
  ;
  NDFMCR0       ←       0x0000_0010       ; NDCE0n pin = 0,  ECC-disable
  NDFMCR1       ←       0x0000_0000       ; ECC = Hamming
  NDFMCR2       ←       0x0000_3343       ; NDWEn  L = 3clks,H =3clks,
                                          ; NDREn  L = 4clks,H = 3clks
  NDFINTC       ←       0x0000_0000       ; ALL Interrupt Disable

  ; ***** Setting Command, Address to NAND-Flash *****
  ;
  NDFMCR0       ←       0x0000_00b0       ;NDCE0n pin = 0,  NDCLE = 1, NDALE = 0
  NDFDTR        ←       0x80              ; Write Command (1st-cycle of Page-Program)
  NDFMCR0       ←       0x0000_00d0       ;NDCE0n pin = 0,  NDCLE = 0, NDALE = 1
  NDFDTR        ←       0x??              ; Write Address (n-times)
  NDFMCR0       ←       0x0000_0095       ;NDCE0n pin = 0,  NDCLE = 0, NDALE = 0
                                          ; ECC Enable and Reset
  ; ***** Writing 512Byte Valid data *****
  ;
  Have the DMAC and INTC support the Autoload function of 512-byte write data. (Details are omitted.)
                                                          (Including INTTC interrupt enable)
  NDFMCR1       ←       0x0000_0300       ; <SELALS> = 1, Start Auto-Load
  ;

———— INTTC Interrupt Processing Program ————

  NDFMCR1       →       Read and check    ; Check that <ALS> = 0 (end). If not, perform polling.
  NDFMCR0       ←       0x0000_0090       ; ECC-Disable
  Return to the main program

——— Main Program ———

  ; ***** Reading ECC from NDFC  *****
  NDECCRD0      →       Read              ; ECC for the first 256 bytes
  NDECCRD1      →       Read              ; ECC for the second 256 bytes

   ; ***** Writing Dummy data & ECC code*****
   ;
   NDFDTR       ←       0x??              ; Write dummy data (1 byte x 8 times)
   NDFDTR       ←       ECC               ; Write ECC (1 byte x 3 times)
   ;                                      Write to D520: LPR7:0       For second 256 bytes
   ;                                      Write to D521: LPR15:8      For second 256 bytes
   ;                                      Write to D522: CPR5:0+11b   For second  256 bytes

   ;
   NDFDTR       ←       0x??              ; Write dummy data (1 byte x 2 times)
   NDFDTR       ←       ECC               ; Write ECC (1byte x 3 times)
   ;                                      Write to D525:LPR7:0        For first 256 bytes
   ;                                      Write to D526:LPR15:8       For first 256 bytes
   ;                                      Write to D527:CPR5:0+11b    For first 256 bytes
```

```
; ***** Set Page program command *****
;
NDFMCR0     ←     0x0000_00b0          ;NDCEn pin = 0,  NDCLE = 1, NDALE = 0
NDFDTR      ←     0x10                 ; Write Command(2nd cycle of Page-Program)
NDFMCR0     ←     0x0000_0010          ;NDCEn pin = 0,  NDCLE = 0, NDALE = 0

; ***** Wait till Page-Program End *****
;
; Wait for the page program to end. Whether or not the program has ended can be checked by two methods:
; 1) write a read status command to read the status from the NDD7 to NDD0 pins (polling method), and
: 2) use a Ready interrupt by detection of NDRB pin rising edge. The following describes a case in which the
; second method is used.
;
NDFINTC     ←     0x0000_0009          ; Clear/enable RDY interrupt
```

Have the INTC enable an NDFC interrupt (Details are omitted)

—— INTNDFC Interrupt Processing Program ——

End processing
Return to the main program

(2) Page read (2LC type)

——— Main Program ———

```
;
; ***** Initialize NDFC *****
;       condition: 8bit-bus, CE0, SLC, 512 Bytes/Page, Hamming
;
NDFMCR0          ←    0x0000_0010        ; NDCEn pin = 0,  ECC-disable
NDFMCR1          ←    0x0000_0000        ; ECC = Hamming
NDFMCR2          ←    0x0000_3343        ; NDWEn  L = 3clks,H = 3clks,
                                         ; NDREn  L = 4clks,H = 3clks
NDFINTC          ←    0x0000_0000        ; ALL Interrupt Disable

; ***** Setting Command, Address to NAND-Flash *****
;
NDFMCR0          ←    0x0000_00b0        ;NDCEn pin = 0,  NDCLE = 1, NDALE = 0
NDFDTR           ←    0x00               ; Write Command (1st-cycle of Page-Read)

 ; ***** Reading 512Byte Valid data *****
 Have the DMAC and INTC support the Autoload function of 512-byte read data. (Details are omitted.)
                                         (Including INTTC interrupt enable)
NDFMCR1          ←    0x0000_0100        ; <SELALS> = 0, Start Auto-Load

NDFMCR0          ←    0x0000_00d0        ; NDCEn pin = 0,  NDCLE = 0, NDALE = 1
NDFDTR           ←    0x??               ; Write Address (n-times)
NDFMCR0          ←    0x0000_00b0        ; NDCEn pin = 0,  NDCLE = 1, NDALE = 0
NDFDTR           ←    0x030              ; Read Command(2nd-cycle of Page-Read)
NDFMCR0          ←    0x0000_0015        ; NDCEn pin = 0,  NDCLE = 0, NDALE = 0
                                         ; ECC Enable and Reset
```

——— INTTC Interrupt Processing Program ———

```
NDFMCR0          ←    0x0000_0010        ; Disable ECC

 ; ***** Reading Dummy data & ECC from NAND-Flash  *****
 ;
NDFDTR           →    Read               ; Read dummy data (1 byte x 8 times)
NDFDTR           →    Read               ; Read ECC    (1 byte x 3 times)
NDFDTR           →    Read               ; Read dummy data (1 byte x 2 times)
NDFDTR           →    Read               ; Read ECC    (1 byte x 3 times)
;
; ***** Reading ECC code from NDFC  *****
;
NDECCRD0              →    Read          ; ECC for the first 256 bytes
NDECCRD1              →    Read          ; ECC for the second 256 bytes
```

Software processing

The ECC generated for the read operation and the ECC read from the memory are compared. If any error is found, the error processing routine is executed to correct the error data. For details, see Section 3.11.4.2 “Error Correction Methods”.

Return to the main program

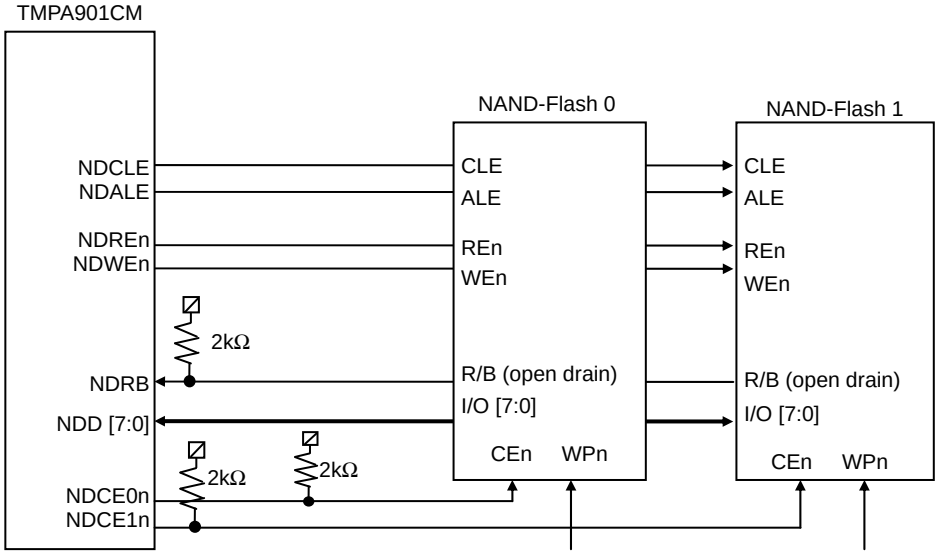
(3) Page write (4LC type)

```
——— Main Program ———
    ;
    ; ***** Initialize for NDFC *****
    ;           condition: 8bit-bus, CE0, MLC, 512 Bytes/Page, Reed Solomon
    ;
    NDFMCR0     ←   0x0000_0410     ; NDCEn  pin = 0,  ECC-disable
    NDFMCR1     ←   0x0000_0002     ; ECC=Reed-Solomon
    NDFMCR2     ←   0x0000_3343     ; NDWEn  L = 3clks,H = 3clks,
                                    ; NDWEn  L = 4clks,H = 3clks
    NDFINTC     ←   0x0000_0000     ; ALL Interrupt Disable

    ; ***** Setting Command, Address to NAND-Flash *****
    ;
    NDFMCR0     ←   0x0000_04b0     ; NDCEn pin = 0,  NDCLE = 1, NDALE = 0
    NDFDTR      ←   0x80            ; Write command (1st-cycle of Page-Program)
    NDFMCR0     ←   0x0000_04d0     ; NDCEn pin = 0,  NDCLE = 0, NDALE = 1
    NDFDTR      ←   0x??            ; Write Address (n-times)
    NDFMCR0     ←   0x0000_0495     ; NDCEn pin = 0,  NDCLE = 0, NDALE = 0
                                    ; ECC enable and reset

    ; ***** Writing 512Byte Valid data *****
    Have the DMAC and INTC support the Autoload function of 512-byte write data. (Details are omitted.)
                                        (Including INTTC interrupt enable)
    NDFMCR1     ←   0x0000_0302     ; <SELALS> = 1, Start Auto-Load
    ;
——— INTTC Interrupt Processing
Program——

    NDFMCR1     →   Read and check  ; Check that <ALS> = 0 (end). If not, perform polling.
    NDFMCR0     ←   0x0000_0490     ; Disable ECC
    Return to the main program

    ——— Main Program ———
    ; ***** Reading ECC from NDFC  *****
    NDECCRD0    →   Read            ; ECC (1/3)
    NDECCRD1    →   Read            ; ECC (2/3)
    NDECCRD2    →   Read            ; ECC (3/3)

    ; ***** Writing Dummy data & ECC *****
    ;
    NDFDTR      ←   ECC             ; Write ECC  (1 byte x 10 times)
    NDFDTR      ←   0x??            ; Write dummy data (1 byte x 6 times)

    ; ***** Set Page program command *****
    ;
    NDFMCR0     ←   0x0000_04b0     ; NDCEn pin = 0,  NDCLE = 1, NDALE = 0
    NDFDTR      ←   0x10            ; Write Command(2nd-cycle of Page-Program)
    NDFMCR0     ←   0x0000_0410     ; NDCEn pin = 0,  NDCLE = 0, NDALE = 0

    ; ***** Wait till Page-Program End *****
    ;
    ; wait for the page program to end. Whether or not the program has ended can be checked by two
    ; methods: 1) write a read status command to read the status from the NDD7 to NDD0 pins (polling)
    ; method), and 2) use a Ready interrupt by detection of NDRB pin rising edge. The following describes a
    ; case in which the second method is used.
    ;
    NDFINTC     ←   0x0000_0009     ; Clear/enable RDY interrupt

    Have the INTC enable an NDFC interrupt. (Details are omitted.)

——— INTNDFC Interrupt Processing Program———

    End processing
    Return to the main program
```

(4) Page Read (4LC type)

```
——— Main Program ———
    ;
    ; ***** Initialize for NDFC *****
    ;           condition: 8bit-bus, CE0, MLC, 512 Bytes/Page, Reed Solomon
    ;
    NDFMCR0     ←   0x0000_0010     ; NDCEn pin = 0,  ECC-disable
    NDFMCR1     ←   0x0000_0002     ; ECC = Reed-Solomon
    NDFMCR2     ←   0x0000_3343     ; NEWEn  L = 3clks,H = 3clks,
                                    ; NDWEn  L = 4clks,H = 3clks
    NDFINTC     ←   0x0000_0000     ; ALL Interrupt Disable

    ; ***** Setting Command, Address to NAND-Flash *****
    ;
    NDFMCR0     ←   0x0000_00b0     ; NDCEn pin = 0,  NDCLE = 1, NDALE = 0
    NDFDTR      ←   0x00            ; Write Command(1st-cycle of Page-Read)

    ; ***** Reading 512Byte Valid data *****
    Have the DMAC and INTC support the Autoload function of 512-byte read data. (Details are omitted.)
                                        (Including INTTC interrupt enable)
    NDFMCR1     ←   0x0000_0102     ; <SELALS> = 0, Start Auto-Load

    NDFMCR0     ←   0x0000_00d0     ; NDCEn pin = 0,  NDCLE = 0, NDALE = 1
    NDFDTR      ←   0x??            ; Write Address (n-times)
    NDFMCR0     ←   0x0000_00b0     ; NDCEn pin = 0,  NDCLE = 1, NDALE = 0
    NDFDTR      ←   0x030           ; Read Command(2nd-cycle of Page-Read)
    NDFMCR0     ←   0x0000_0015     ; NDCEn pin = 0,  NDCLE = 0, NDALE = 0
                                    ; ECC Enable and Reset, <RSECGW> = 0

——— INTTC Interrupt Processing Program ———

    NDFMCR0     ←   0x0000_0114     ; ECC-Enable, <RSECGW>=1
    Return to the main program

——— Main Program ———
    ; ***** Reading Dummy data & ECC from NAND-Flash  *****
    ;
    NDFDTR      →   Read            ; Read ECC (1 byte x 10 times)
    ;
    ; ***** Calculation Error Address and Data *****
    ;
    NDFINTC     ←   0x0000_0090     ; Clear/enable R/S calculation end interrupt
    Have the INTC enable an NDFC interrupt. (Details are omitted.)
    NDFMCR0     ←   0x0000_0310     ; Disable ECC, <RSECGW> = 1, <RSESTA> = 1

    ——— INTNDFC Interrupt Processing Program
    ———
    NDFMCR1     →   Read and check  ; Check the <STATE> and <SERR> flags.
    Software processing
        If any error is found, the error processing routine is executed to correct the error data.
        For details, see Section 3.11.4.2 “Error Correction Methods”.

    Return to the main program
```

3.11.7 Example of Connection with the NAND-Flash



Note 1: The pull-up resistor value for the NDRB pin and the NDCE[1:0]n pins must be set appropriately according to the NAND Flash memory to be used and the capacity of the board (typical: 2 kΩ).

Note 2: The WPn (Write Protect) pin of the NAND Flash is not supported. When this function is needed, prepare it on an external circuit.

Figure 3.11.2 Example of Connection with the NAND-Flash

3.12 16-Bit Timers/PWM

3.12.1 General Description of Functions

The TMPA901CM contains six channels of 16-bit timers. They operate in the following two modes:

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

The circuit consists of three blocks, each associated with two channels. Of the three blocks, Block 1 and Block 2 support PWM (Pulse Width Modulation) output.

	Block 1		Block 2		Block 3	
	Timer0	Timer1	Timer2	Timer3	Timer4	Timer5
Free-Running	○	○	○	○	○	○
Periodic timer	○	○	○	○	○	○
PWM	○	N/A	○	N/A	N/A	
	PWM0OUT (PC3)	×	PWM2OUT (PC4)	×	×	×
Interrupt source signal	INTS[2]		INTS[3]		INTS[4]	

Since all blocks are of the same specifications (except for the PWM function and Interrupt source) only the circuit of Block 1 is described here.

3.12.2 Block Diagrams

Each timer block, containing two channels of timer circuits, is comprised of two programmable 16-bit free-running decrement counters. The TIMCLK input is used for counter operation. This clock can be selected from the internal system clock divided by two ($f_{PCLK}/2$) and fs (32.768 kHz).

Figure 3.12.1 shows a diagram of the timer block (Timer 0 and Timer 1).

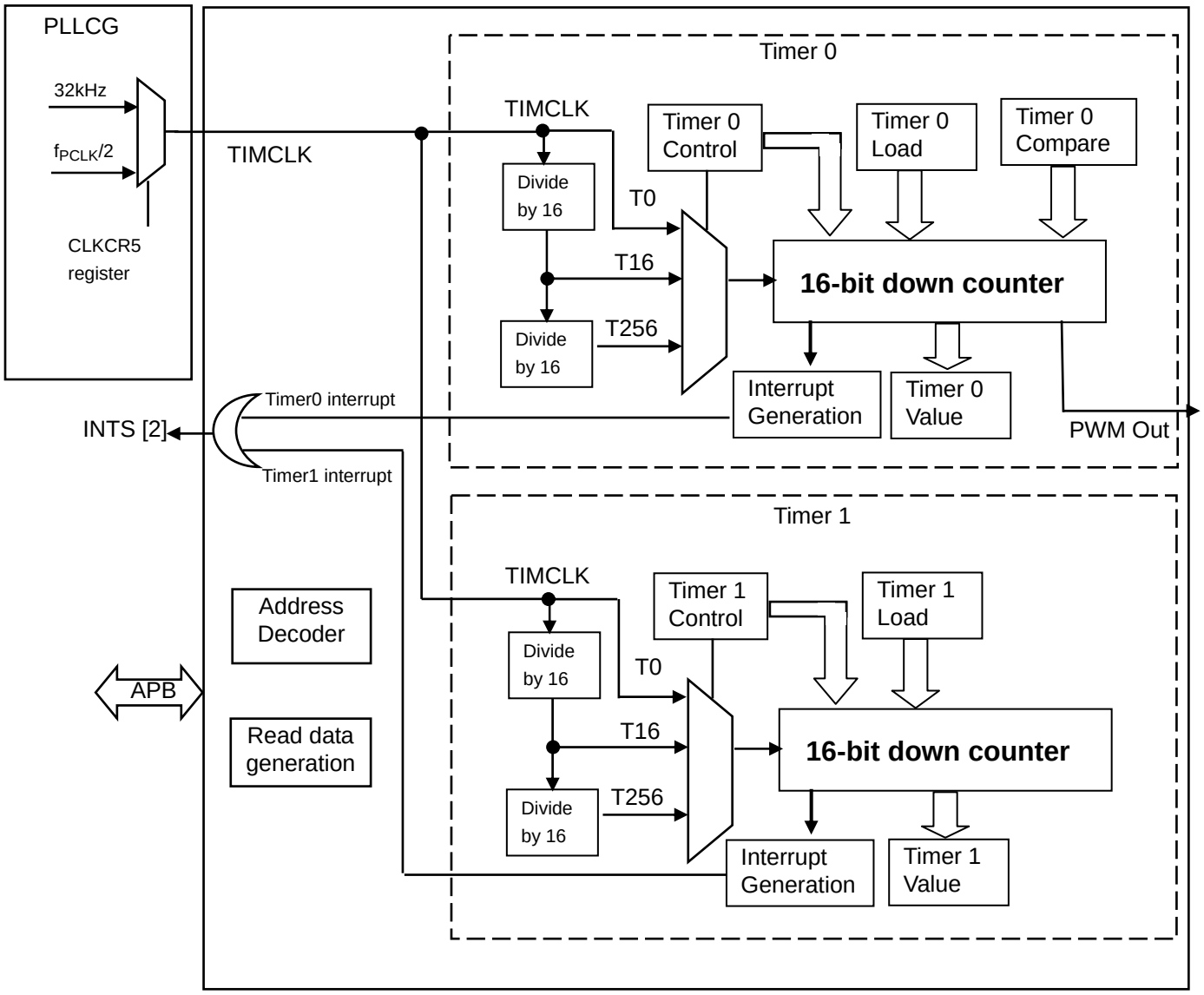


Figure 3.12.1 Timer Block Diagram (Timer 0 and Timer 1)

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

T0: $f_{PCLK}/2$

T16: $f_{PCLK}/2$ divided by 16, generated by a 4-bit prescaler.

T256: $f_{PCLK}/2$ divided by 256, generated by an 8-bit prescaler.

3.12.3 Operation Descriptions

The following descriptions are based on setting examples for Timer 0. The timers of other channels operate identically to Timer 0.

1) Free-running mode

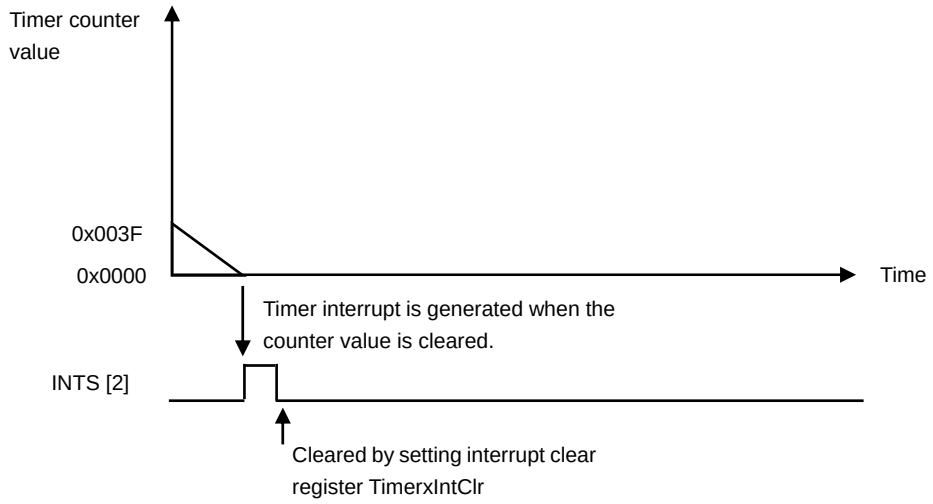
When the timer starts counting, the counter value decrements from the initially set value. When the counter value reaches 0, an interrupt is generated.

For the One-shot-operation (Timer0Control<TIM0OSCTL> = 1), the interrupt is generated once.

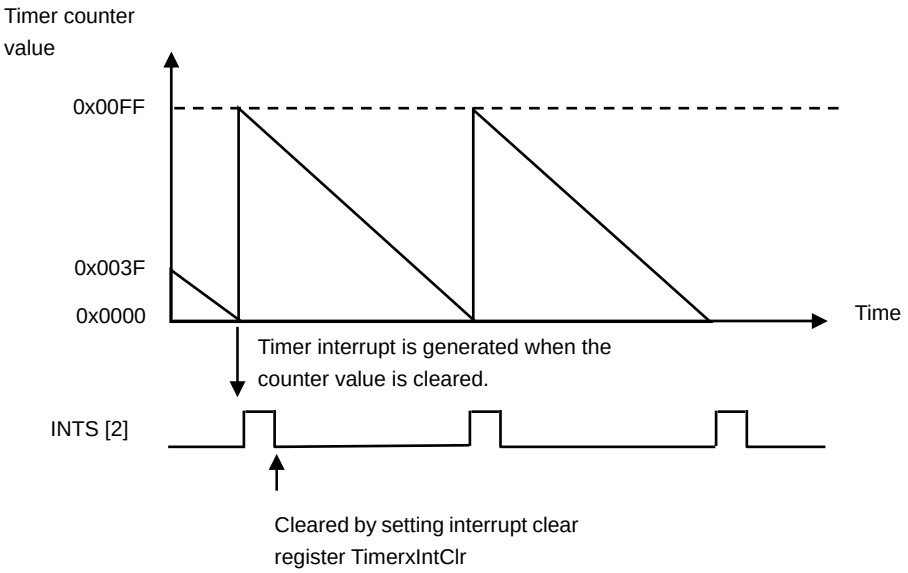
Upon reaching 0, the counter is reloaded with the maximum value and continues decrementing if Wrapping-operation is enabled (Timer0Control<TIM0OSCTL> = 0). The maximum value is 0x000000FF for the 8-bit counter and 0x0000FFFF for the 16-bit counter.

The following shows an example where the 8-bit timer counter and timer value is set to 0x0000003F.

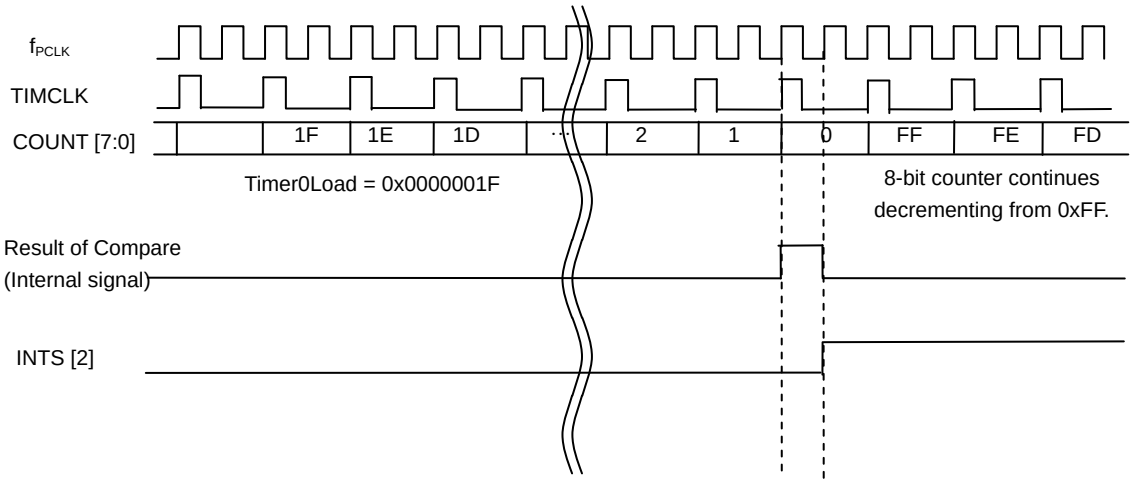
(1) One-shot-operation



(2) Wrapping-operation



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



Example of settings for free-running mode (Wrapping-Operation)

Bits / Register	MSB [31:8]	7	6	5	4	3	2	1	LSB 0	Function
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stops Timer 0.
Timer0Load	0x000000	0	0	0	1	1	1	1	1	[15:0]: Timer 0 period = 0x0000001F
Timer0Control	0x000000	1	0	1	0	0	0	0	0	[7]: Enables Timer 0 (Starts counting). [6]: Selects free-running mode. [5]: Enables timer interrupts. [3:2]: Selects input clock T0. [1]: Selects 8-bit counter. [0]: Selects Wrapping-operation.

×: Don' t care

2) Periodic timer mode

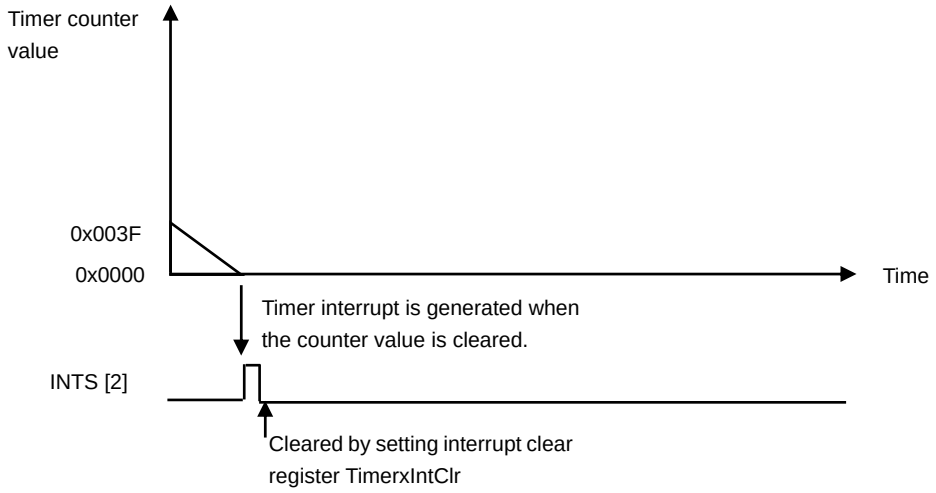
When the timer starts counting, the counter value decrements from the initially set value. When the counter value reaches 0, an interrupt is generated.

If setting to the One-shot-operation (Timer0Control<TIM0OSCTL> = 1), the interrupt is generated once.

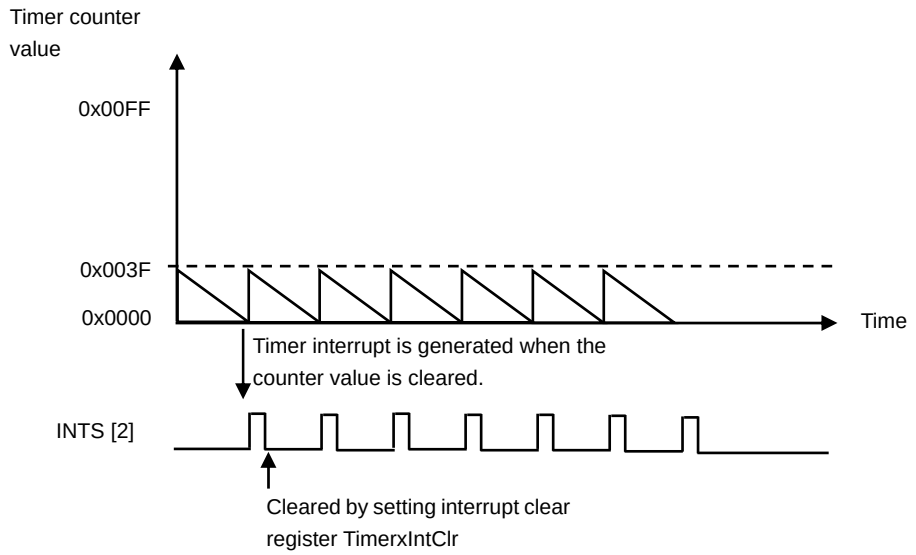
Upon reaching 0, the counter is reloaded with the initially set value and continues decrementing if Wrapping-operation is enabled (Timer0Control<TIM0OSCTL> = 0). Therefore, interrupts are generated at fixed intervals.

The following shows an example where the 8-bit counter and timer value is set to 0x0000003F.

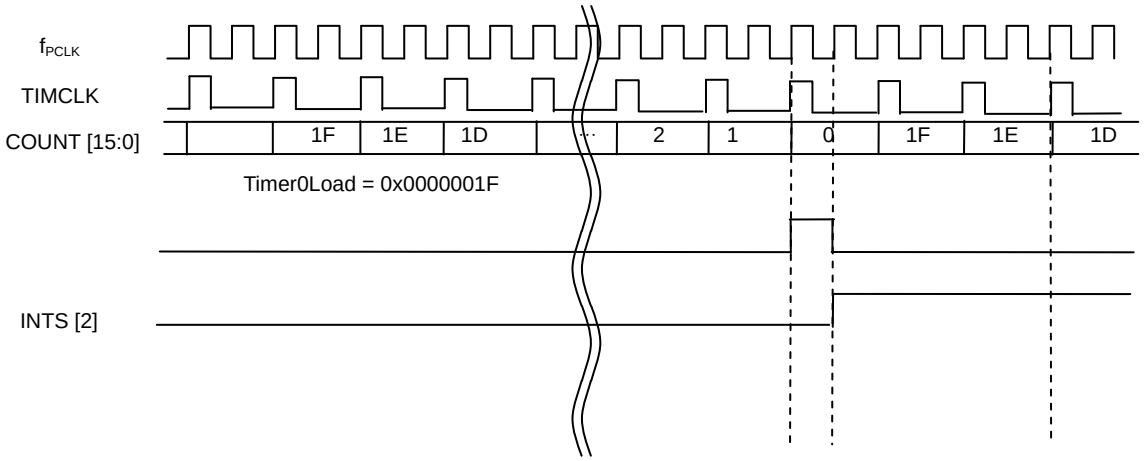
(1) One-shot-operation



(2) Wrapping-operation



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



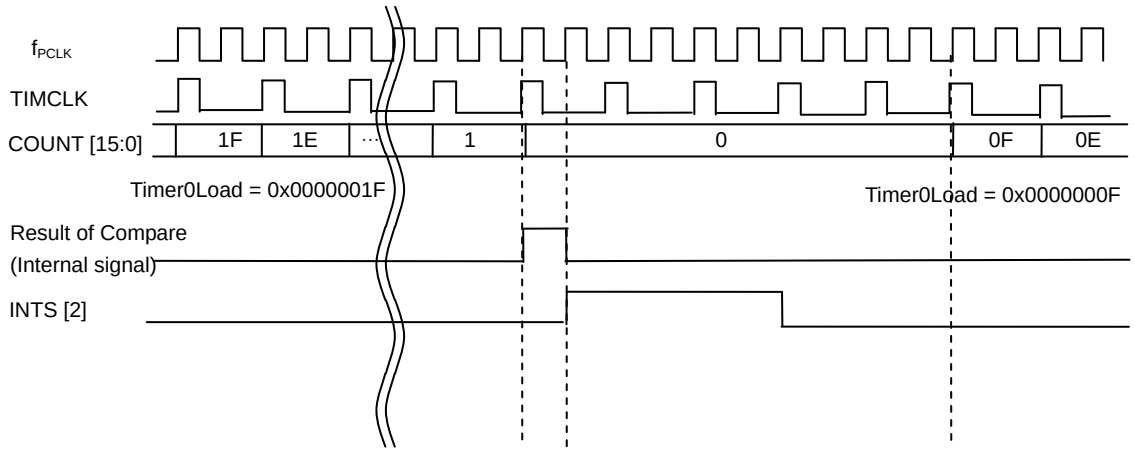
Example of settings for periodic timer mode (Wrapping-Operation)

Bits / Register	MSB [31:8]	7	6	5	4	3	2	1	LSB 0	Function
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stops Timer 0.
Timer0Load	0x000000	0	0	0	1	1	1	1	1	[15:0]: Timer 0 period = 0x0000001F
Timer0Control	0x000000	1	1	1	0	0	0	1	0	[7]: Enables Timer 0 (Starts counting). [6]: Selects periodic timer mode. [5]: Enables timer interrupts. [3:2]: Selects input clock T0. [1]: Selects 16-bit counter. [0]: Selects Wrapping-operation.

×: Don' t care

- One-shot operation

When One-shot operation is selected, a new value must be set in the Timer0Load register before the timer can be restarted. If Timer0Control<TIM0EN> is set to 1 without setting a new value in the Timer0Load register, the timer cannot be restarted.



Example of settings for Free-Running mode (One-shot operation)

Bits / Register	MSB [31:8]	7	6	5	4	3	2	1	LSB 0	Function
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stops Timer 0.
Timer0Load	0x000000	0	0	0	1	1	1	1	1	[15:0]: Timer 0 period = 0x0000001F
Timer0Control	0x000000	1	0	1	0	0	0	0	1	[7]: Enables Timer 0 (Starts counting). [6]: Selects Free-running mode. [5]: Enable timer interrupts. [3:2]: Selects input clock T0. [1]: Selects 8-bit counter. [0]: Selects one-shot operation.
Timer0IntClr	×	×	×	×	×	×	×	×	×	[32:0]: Writing any value clears the interrupt.

×: Don' t care

- PWM function support

Block 1 and Block 2 are provided with two channels of the 16-bit PWM function. The two channels of PWM output are output on the PWM0OUT (PC3) and PWM2OUT (PC4) pins.

The PWM0OUT output is inverted when the value of the decrement counter matches the value set in the Timer0Compare1 register or when the counter value set in Timer0Mode<PWM Period> decrements to 0.

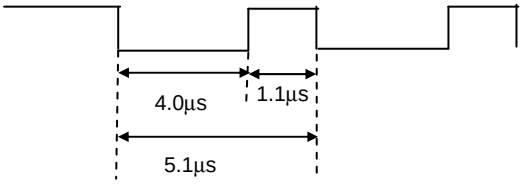
The Timer0Compare1 register can be set in a range of duty 0% to 100%. When the decrement counter value reaches 0, the counter resumes counting down from "2^n-2".

The two channels have the same specifications and the above explanation also applies to Timer 2.

Note1: When using PWM function, be sure to select "periodic timer mode", "16-bit counter" and "wrapping operation" in the control register.

Note2: The status of PWM output is kept, if PMD function is stopped by setting TIMER0 Control<TIM0EN> to "0" to stop the timer. If 1 is set to " Timer0Control<TIM0EN> " in this state and the timer is made to work again, the output of PWM starts from the state when the timer stopped before. Therefore, if PWM output is High when timer is stopped, the phase reverses because it is output from High level. Output data is initialized by disabling the PWM function, therefore, Timer0Mode<PWM Mode> must be set**"** disable **"**to make the PWM function work again.

Example: Outputting the following PWM waveform on the PWM0OUT pin by using Timer 0 with f_{PCLK} = 100 MHz and TIMCLK = 50 MHz



(1) To realize the PWM period of 5.1μs with T0 = 0.02 μs:

$5.1\mu s \div 0.02\ \mu s = 255 = 2^n - 1$

Therefore, n=8.

(2) Since the Low level period is 4.0 μs, the value to be set in Timer0Compare1 is calculated as follows with T0 = 0.02 μs:

$(5.1\mu s - 4.0\ \mu s) / 0.02\ \mu s = 55 = 0x37$

Bits / Register	MSB [31:8]	7	6	5	4	3	2	1	LSB 0	Function
Timer0Control	0x000000	0	×	×	0	×	×	×	×	[7]: Stops Timer 0.
Timer0Mode	0x000000	0	1	0	0	0	0	0	0	[6], [5:4]: Selects PWM mode Sets PWM period to 2^8-1.
Timer0Compare1	0x000000	0	0	1	1	0	1	1	1	[7:0]: Sets the compare value 0x37.
Timer0CmpEn	0x000000	0	0	0	0	0	0	0	1	[0]: Enables compare.
Timer0Control	0x000000	1	1	1	0	0	0	1	0	[7]: Enables Timer 0 (Starts counting). [6]: Selects periodic timer mode. [5]: Enables timer interrupts. [3:2]: Selects input clock T0. [1]: Selects16-bit counter without exception. [0]: Selects wrapping operation.

×: Don' t care

The following describes PWM minimum resolutions and duty.

Table 3.12.1 PWM Minimum Resolutions (TIMCLK = 50 MHz)

PWM period / Prescaler	2^8-1	2^9-1	$2^{10}-1$	$2^{16}-1$
T0	5.1 μs	10.22 μs	20.46 μs	1.31 ms
T16	81.6 μs	163.52 μs	327.36 μs	20.97 ms
T256	1.305 ms	2.62 ms	5.24 ms	335.54 ms

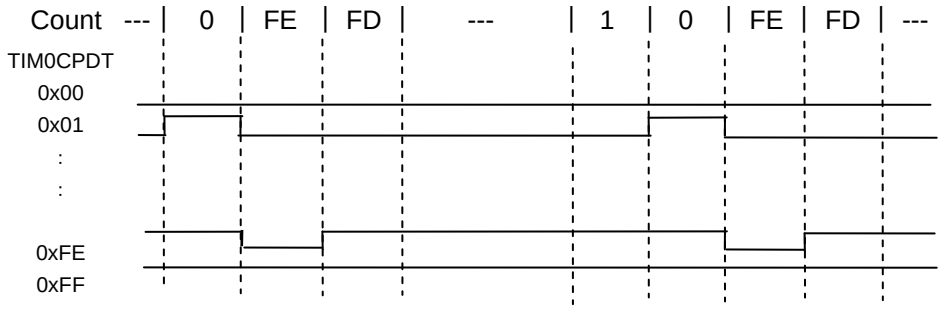


Table 3.12.2 PWM output waveform example

Example: Duty settings when the PWM counter period is 2^8-1 (255 counts)

The initial value of PWM output is always Low output. Duty 0% is always Low output, and duty 100% is always High output.

Timer0Compare1 = 0x00: Duty = $0/255 \times 100 = 0$ %

Timer0Compare1 = 0x01: Duty = $1/255 \times 100 = 0.39$ %

:

:

Timer0Compare1 = 0xFE: Duty = $254/255 \times 100 = 99.6$ %

Timer0Compare1 = 0xFF: Duty = $255/255 \times 100 = 100$ %

- When the PWM period is $2^n - 1$, setting 2^n-1 to Timer0Compare1 sets the flip-flop for PWM output to High . To start PWM output by modifying only the PWM period from this state, PWM mode must be disabled once to modify the setting.
- The following requirements must be met when PWM mode is used.
  $0 \leq$ (Setting value of TimerxCompare1) $\leq 2^n-1$

3.12.4 Register Descriptions

The following lists the SFRs.

Table 3.12.2 SFR (1/3) Base address = 0xF004_0000

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 3.12.3 SFR (2/3) Base address = 0xF004_1000

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 3.12.4 SFR (3/3) Base address = 0xF004_2000

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
–	0x001C	Reserved
–	0x0020	Reserved
–	0x0040	Reserved
–	0x0060	Reserved
–	0x0064	Reserved
–	0x0068	Reserved
–	0x00A0	Reserved
–	0x00C0	Reserved
–	0x00E0	Reserved
–	0x00E4	Reserved
–	0x00E8	Reserved
–	0x00EC	Reserved
:	:	:
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

1. Timer0Load Register

Address = (0xF004_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	TIM0SD[15:0]	R/W	0x0000	Set the interval value of Timer 0.

[Description]

a. <TIMxSD[15:0]>

This register is used to set the timer period.

The counter is a decrement counter and the counter value can be set in a range of 0x0001-0xFFFF. (The value to be set here should be the desired counter value decremented by one.

Note: 0x0000 setting is prohibited. 0x0000 setting is started decrement from 0xFFFF.

When the 8-bit counter is used, the upper 8 bits are ignored.

When the counter runs in periodic timer mode and Wrapping-operation is enabled, the value set in this register is reloaded into the counter when the counter value reaches 0x0000.

The value written in this register is immediately reflected in the counter.

To renew the counter value when the counter value reaches 0x0000, the Timer0BGLoad register described later can be used.

- TimerxLoad (Timer x Load value register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0Load.

  Please refer to the description of Timer0Load.

  For the name and address of these registers, please refer to Table 3.12.2.

2. Timer0Value Register

Address = (0xF004_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:0]	TIM0CD[15:0]	RO	0xFFFF	Current counter value of Timer 0

[Description]

a. <TIMxCD[15:0]>

This register is used to read the current timer value.

It indicates the current value of the decrement counter.

- TimerxValue (Timerx value register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0Value.

  Please refer to the description of Timer0Value.

  For the name and address of these registers, please refer to Table 3.12.2.

3. Timer0Control Register

Address = (0xF004_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	TIM0EN	R/W	0y0	Timer 0 enable bit 0y0: Disable 0y1: Enable
[6]	TIM0MOD	R/W	0y0	Timer 0 mode setting 0y0: Free-running mode 0y1: Periodic timer mode
[5]	TIM0INTE	R/W	0y1	Timer 0 interrupt control 0y0: Disable interrupts 0y1: Enable interrupts
[4]	–	–	Undefined	Read as undefined. Write as zero.
[3:2]	TIM0PRS	R/W	0y00	Timer 0 prescaler setting 0y00: No division 0y01: Divide by 16 0y10: Divide by 256 0y11: Setting prohibited
[1]	TIM0SIZE	R/W	0y0	8-bit/16-bit counter select for Timer 0 0y0: 8-bit counter 0y1: 16-bit counter
[0]	TIM0OSCTL	R/W	0y0	One-shot/Wrapping operation select for Timer 0 0y0: Wrapping operation 0y1: One-shot operation

[Description]

a. <TIMxEN>

This bit is used to enable or disable timer operation.

0y0: Disable

0y1: Enable

Re-enabling timer operation after the timer is stopped in the middle of counting

If the timer is stopped in the middle of counting, the timer retains the count value and resumes decrementing from this value when re-enabled. However, if a new value is set in the TimerxLoad register before timer operation is re-enabled, the timer starts decrementing from the value set in the TimerxLoad register.

b. <TIMxMOD>

This bit is used to switch timer operation modes.

c. <TIMxINTE>

This bit is used to control masking of timer interrupts.

d. <TIMxPRS>

This bit is used to set the prescale value for dividing the timer source clock.

e. <TIMxSIZE>
This bit is used to select the 8-bit or 16-bit counter.

f. <TIMxOSCTL>
This bit is used to select one-shot or wrapping operation.

- TimerxControl (Timerx Control register) (x = 0 to 5)
  The structure and description of these registers are same as Timer0Control.
  Please refer to the description of Timer0Control.
  For the name and address of these registers, please refer to Table 3.12.2.

4. Timer0IntClr Register

Address = (0xF004_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	TIM0INTCLR	WO	Undefined	Timer 0 interrupt clear

[Description]

a. <TIMxINTCLR>

This register is used to clear timer interrupts.

Writing any value in this register causes the corresponding interrupt to be cleared.

(The bus widths of 8, 16 and 32 bits are supported.)

- TimerxIntClr (Timerx Interrupt Clear register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0IntClr.

  Please refer to the description of Timer0IntClr.

  For the name and address of these registers, please refer to Table 3.12.2.

5. Timer0RIS Register

Address = (0xF004_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined.
[0]	TIM0RIF	RO	0y0	Timer 0 interrupt flag 0y0: No interrupt 0y1: Interrupt requested

[Description]

a. <TIMxRIF>

This register indicates the interrupt status of the internal counter, regardless of the interrupt enabled/disabled status specified in IMxCR<TIMxINTE>.

- TimerxRIS (Timerx Interrupt Raw Flag register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0RIS.

  Please refer to the description of Timer0RIS.

  For the name and address of these registers, please refer to Table 3.12.2.

6. Timer0MIS Register

Address = (0xF004_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined.
[0]	TIM0MIF	RO	0y0	Timer 0 interrupt flag 0y0: No interrupt 0y1: Interrupt requested

[Description]

a. <TIMxMIF>

This register indicates the masked interrupt status, reflecting the interrupt enabled/disabled status specified in TIMxCR< TIMxINTE>.

(This register is always 0 when TIMxCR< TIMxINTE> = 0.)

- TimerxMIS (Timerx Interrupt Masked Flag register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0MIS.

  Please refer to the description of Timer0MIS.

  For the name and address of these registers, please refer to Table 3.12.2.

7. Timer0BGLoad Register

Address = (0xF004_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	TIM0BSD[15:0]	R/W	0x00	Set the interval value of the background counter for Timer 0.

[Description]

a. <TIMxBSD[15:0]>

This register is used to set the value of the background counter for the TimerxLoad register.

When the counter runs in periodic timer mode and Wrapping-operation is enabled, this register is used to reload the counter value.

Unlike a write to the TimerxLoad register, a write to the TimerxBGLoad register is not immediately reflected in the counter. The counter is reloaded with the new value when it reaches 0x0000.

The Read/Write operating relation of the TimerxLoad register and the TimerxBGLoad register is shown below.

- When writing :

When writing to the TimerxLoad register, same data is written into both the TimerxLoad register and the TimerxBGLoad register. However, when writing to the TimerxBGLoad register, the data is written into the TimerxBGLoad register only.

- When reading :

When reading either the TimerxLoad register or the TimerxBGLoad register, the data is read from the TimerxBGLoad register and the latest setting value of the timer period can be read.

- TimerxBGLoad (Timer x Back Ground Counter Data register) (x = 0 to 5)

The structure and description of these registers are same as Timer0BGLoad.

Please refer to the description of Timer0BGLoad.

For the name and address of these registers, please refer to Table 3.12.2.

8. Timer0Mode Register

Address = (0xF004_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined. Write as zero.
[6]	PWM Mode	R/W	0y0	PWM mode select: 0y0: PWM disabled 0y1: PWM enabled
[5:4]	PWM Period	R/W	0y00	PWM mode period select: 0y00: 2^8 - 1 0y01: 2^9 - 1 0y10: 2^{10} - 1 0y11: 2^{16} - 1
[3:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <PWM Mode>

This register is used to enable or disable PWM mode.

b. <PWM Period>

This register is used to specify the PWM mode period.

- TimerxMode (Timerx mode register) (x = 0, 2)

  The structure and description of these registers are same as Timer0Mode.

  Please refer to the description of Timer0Mode.

  For the name and address of these registers, please refer to Table 3.12.2.

9. Timer0Compare1 Register

Address = (0xF004_0000) + (0x00A0)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	TIM0CPD	R/W	0x00	Set the value to be compared with the counter value of Timer 0: 0x0001-0xFFFF

[Description]

a. <TIMxCPD>

Set the value to be compared with the counter value of Timer.

- TimerxCompare1 (Timer x Compare Value register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0Compare1.

  Please refer to the description of Timer0Compare1.

  For the name and address of these registers, please refer to Table 3.12.2.

10. Timer0CmpIntClr1 Register

Address = (0xF004_0000) + (0x00C0)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	TIM0CMINTCLR	WO	Undefined	Timer 0 compare interrupt clear

[Description]

a. <TIMxCMINTCLR>
This register is used to clear timer compare interrupts.
Writing any value in this register causes the corresponding interrupt to be cleared.
(The bus widths of 8, 16 and 32 bits are supported.)

- TimerxCmpIntClr1 (Timer x Compare Interrupt Clear register) (x = 0 to 5)
The structure and description of these registers are same as Timer0CmpIntClr.
Please refer to the description of Timer0 CmpIntClr.
For the name and address of these registers, please refer to Table 3.12.2.

11. Timer0CmpEn Register

Address = (0xF004_0000) + (0x00E0)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	TIM0CPE	RO	0y0	Timer 0 compare operation enable 0y0: Disable 0y1: Enable

[Description]

a. <TIMxCPE>

This register is used to enable compare operation of the timer.

It is also used to mask interrupts.

- TimerxCmpEn (Timer x Compare Enable register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0CmpEn.

  Please refer to the description of Timer0CmpEn.

  For the name and address of these registers, please refer to Table 3.12.2.

12. Timer0CmpRIS Register

Address = (0xF004_0000) + (0x00E4)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined.
[0]	TIM0CRIF	RO	0y0	Timer 0 compare raw interrupt status before enabled compare operation 0y0: No interrupt 0y1: Interrupt requested

[Description]

a. <TIMxCRIF>

This register indicates the status of the raw compare interrupt before enabled compare operation, regardless of the interrupt enabled/disabled status specified in TIMxCPMIS.

- TimerxCmpRIS (Timer x Compare raw interrupt status register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0CmpRIS.

  Please refer to the description of Timer0CmpRIS.

  For the name and address of these registers, please refer to Table 3.12.2.

13. Timer0CmpMIS Register

Address = (0xF004_0000) + (0x00E8)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	TIM0CMIF	RO	0y0	Timer 0 compare interrupt flag 0y0: No interrupt 0y1: Interrupt requested

[Description]

a. <TIMxCMIF>

This register indicates the status of the masked compare interrupt.

- TimerxCmpMIS (Timerx Compare Masked interrupt status register) (x = 0 to 5)

  The structure and description of these registers are same as Timer0CmpMIS.

  Please refer to the description of Timer0CmpMIS.

  For the name and address of these registers, please refer to Table 3.12.2.

14. Timer0BGCmp Register

Address = (0xF004_0000) + (0x00EC)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	TIM0BGCPD	R/W	0x0000	Set the background value to be compared with the counter value of Timer 0: 0x0001-0xFFFF

When the compare value to be reloaded is written in the TimerxBGCmp register while the timer is running in periodic timer mode, the timer continues counting until the counter value reaches 0. Then, the value set in the TimerxBGCmp register is shifted to the TimerxCompare1 register.

The following requirements must be met when PWM mode is used.

$0 \leq$ (Setting value of TIMxBGCPD) $\leq 2^n - 1$

The Read/Write operating relation of the TimerxLoad register and the TimerxBGLoad register is shown below.

- When writing :

  When writing to the TimerxCompare1 register, same data is written into both the TimerxCompare1 register and the TimerxBGCmp register. However, when writing to the TimerxBGCmp register, the data is written into the TimerxBGCmp register only.

- When reading :

  When reading either the TimerxCompare1 register or the TimerxBGCmp register, the data is read from the TimerxBGCmp register and the latest setting value of the timer compare can be read.

- TimerxBGCmp (Timer x Back Ground Compare register) (x = 0, 2, 4)

  The structure and description of these registers are same as Timer0BGCmp.

  Please refer to the description of Timer0BGCmp.

  For the name and address of these registers, please refer to Table 3.12.2.

3.13 UART

This LSI contains two UART channels. The feature of each channel is shown below.

	Channel 0	Channel 1
Transmit FIFO	8-bit width / 16 location deep	
Receive FIFO	12-bit width /16location deep	
Transmit/Receive data format	DATA bits : 5,6,7,8bits can be selected PARITY: use / no use STOP bit:1bit / 2bits	
FIFO ON/OFF	ON (FIFO mode)/ OFF (characters mode)	
Interrupt	(1) Combined interrupt factors are output to interrupt controller. (2) The permission of each interrupt factor is programmable.	
baud rate generator	Generates a common transmit and receive internal clock from the UART internal reference clock input. Supports baud rates of up to 6.15Mbps at f_{PCLK} = 100MHz.	
DMA	support	N/A
IrDA 1.0 Function	(1) Max data rate: 115.2kbps(half-duplex) (2) support low power mode	N/A
Control pins	U0RXD U0TXD	U1RXD U1TXD U1CTSn
Hardware flow control	RTS support CTS support	CTS support

(1) UART transmit/receive data format

Transmit/receive data format			
START	DATA (LSB → MSB)	PARITY	STOP

(2) Receive FIFO data format

	Receive data (LSB → MSB)								Framing error flag	Parity error flag	Break error flag	Overrun error flag
Bit Number	0	1	2	3	4	5	6	7				
Receive 8-bit data	1	1	1	1	1	1	1	1				
Receive 7-bit data	1	1	1	1	1	1	1	0				
Receive 6-bit data	1	1	1	1	1	1	0	0				
Receive 5-bit data	1	1	1	1	1	0	0	0				

3.13.1 Block Diagrams

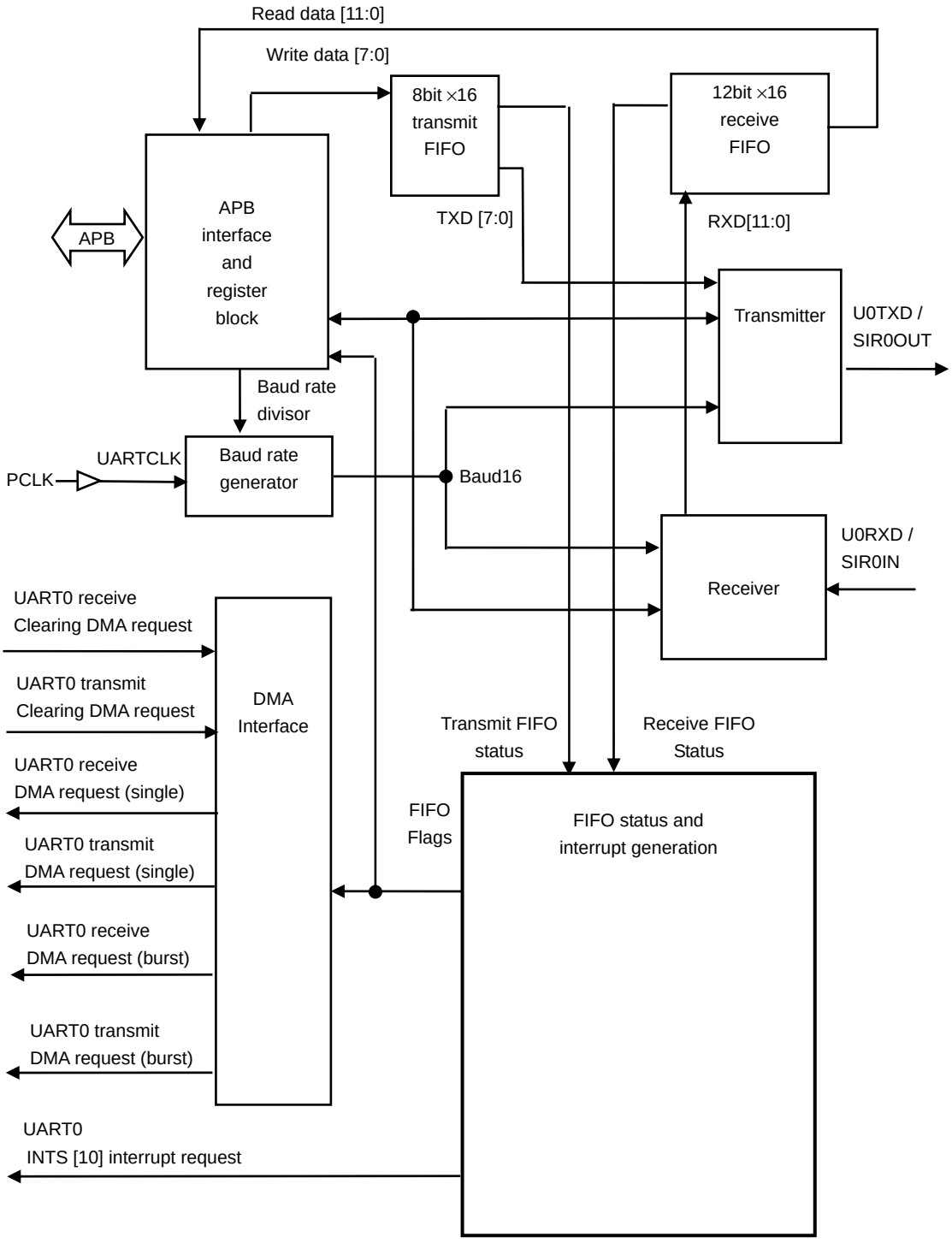


Figure 3.13.1 UART Channel 0 Block Diagram

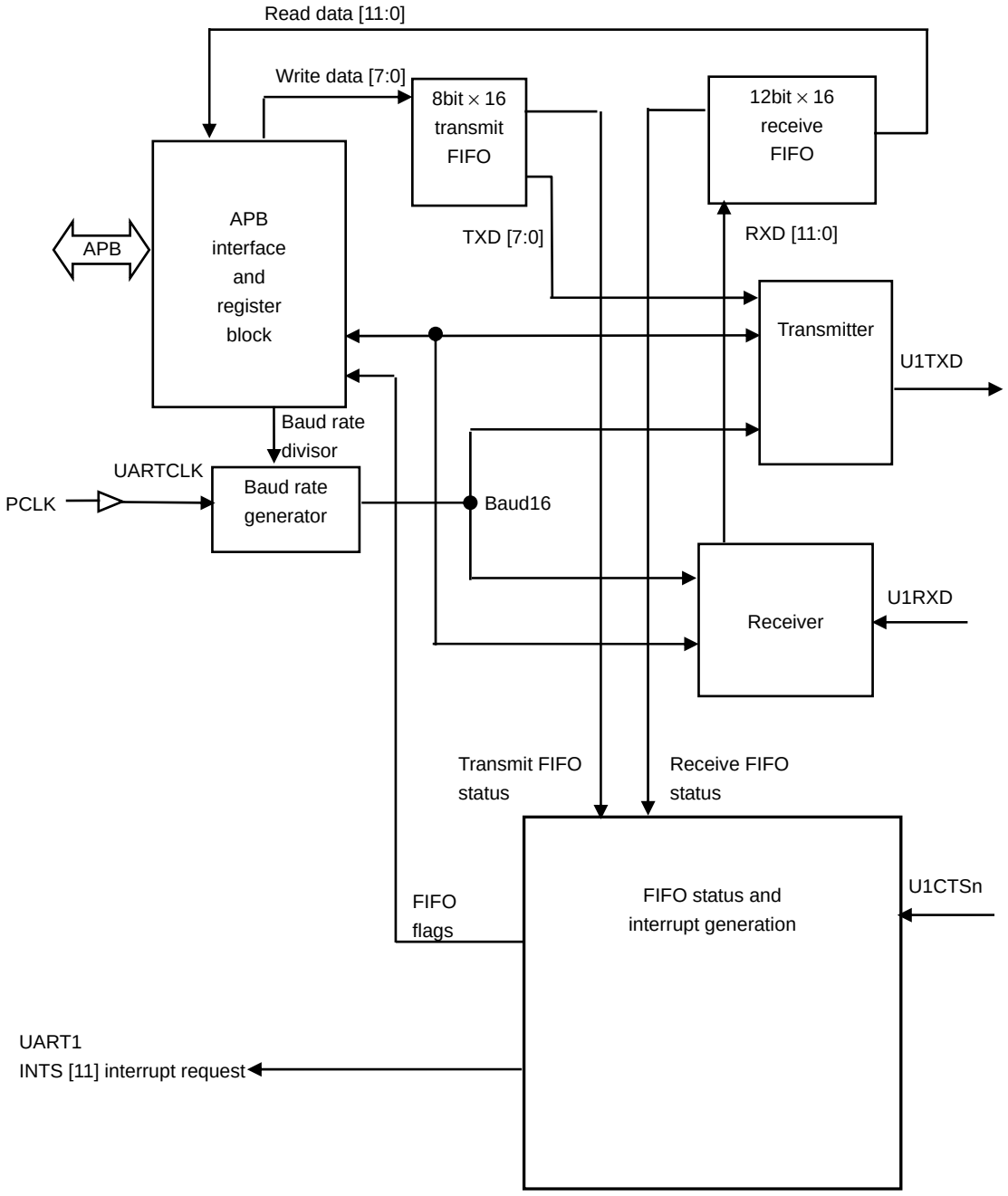


Figure 3.13.2 UART Channel 1 Block Diagram

3.13.1.1 Operation Description

(1) Baud rate generator

The baud rate generator contains the internal Baud16 clock circuit which controls the timing of UART transmit and receive, and the internal IrLPBaud16 circuit which generates the pulse width of the IrDA encoded transmit bit stream when in low-power mode.

(2) Transmit FIFO

The transmit FIFO is an 8-bit wide, 16-location 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, 16 locations deep, FIFO memory buffer. Received data and corresponding error bits are stored in the receive FIFO by the receive logic until they are read out by the CPU across the APB interface. The receive FIFO can be disabled 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 start bit has been detected. Error check for overrun, parity and frame and line break detection are also performed. Their error bit data is written to the receive FIFO.

(6) Interrupt generation logic

UART outputs a maskable combined interrupt for every interrupt sources.

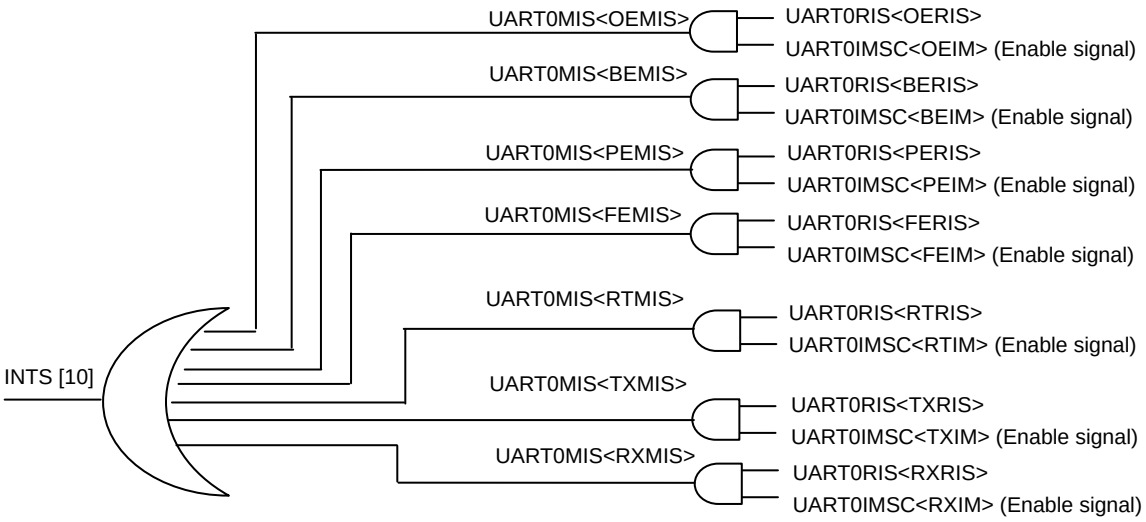
(7) Interrupt timing

Interrupt type	Interrupt timing
Overrun error	After receiving the stop bit of Overflow data
Break error	After receiving STOP bit
Parity error	After receiving parity data
Frame error	After receiving frame over bit
Receive timeout error	After 511 clocks (Baud16) from Receive FIFO data storage.
Transmit interrupt	After transmitting the last data (MSB data).
Receive interrupt	After receiving STOP bit

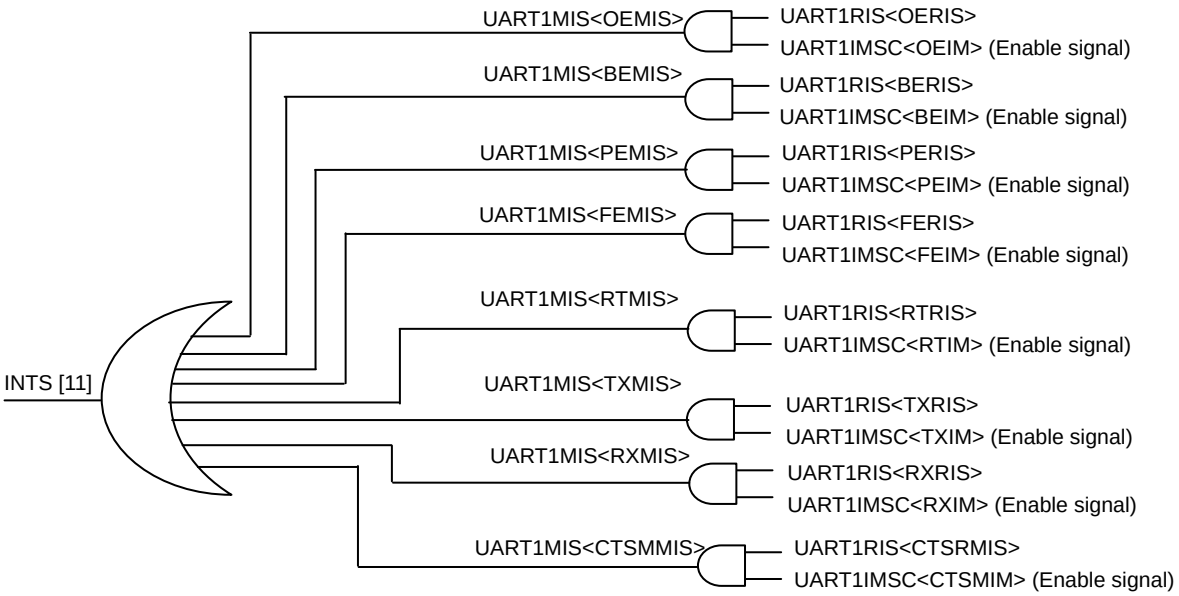
Note: The number of STOP bit can be selected as 1 bit or 2 bits by setting UARTxLCR_H<STP2>. The term STOP bit here means the last STOP bit.

(8) UART interrupt block

1) UART0 interrupt block



2) UART1 interrupt block



(9) DMA interface

The UART0 supports DMA controller. (The UART1 does not support it)

(10) IrDA circuit description

The IrDA is comprised of:

- IrDA SIR transmit encoder
- IrDA SIR receive decoder

Note: The transmit encoder output (SIROUT) has the opposite polarity to the receive decoder input (SIRIN). Please refer to Figure 3.13.4.

Figure 3.13.3 shows a block diagram of the IrDA circuit.

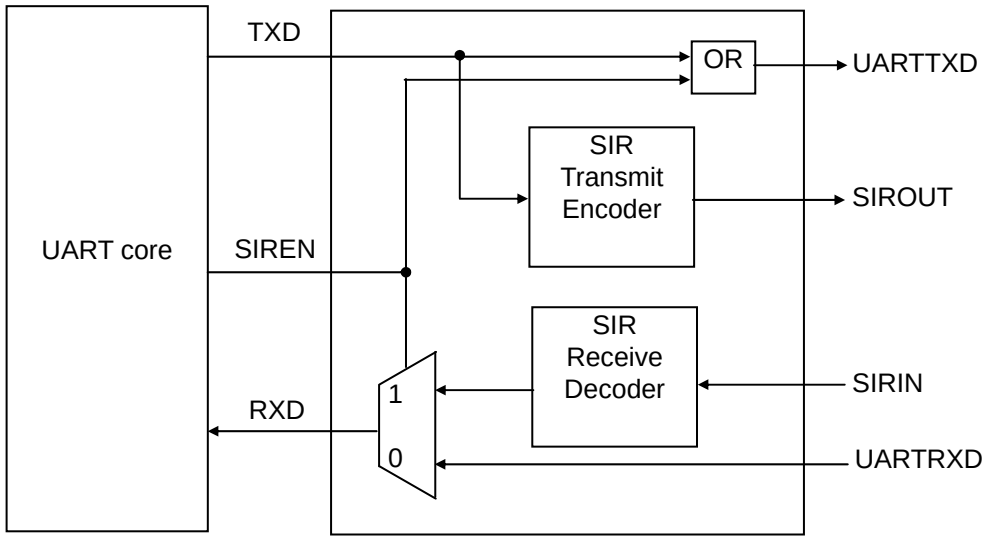


Figure 3.13.3 IrDA Circuit Block Diagram

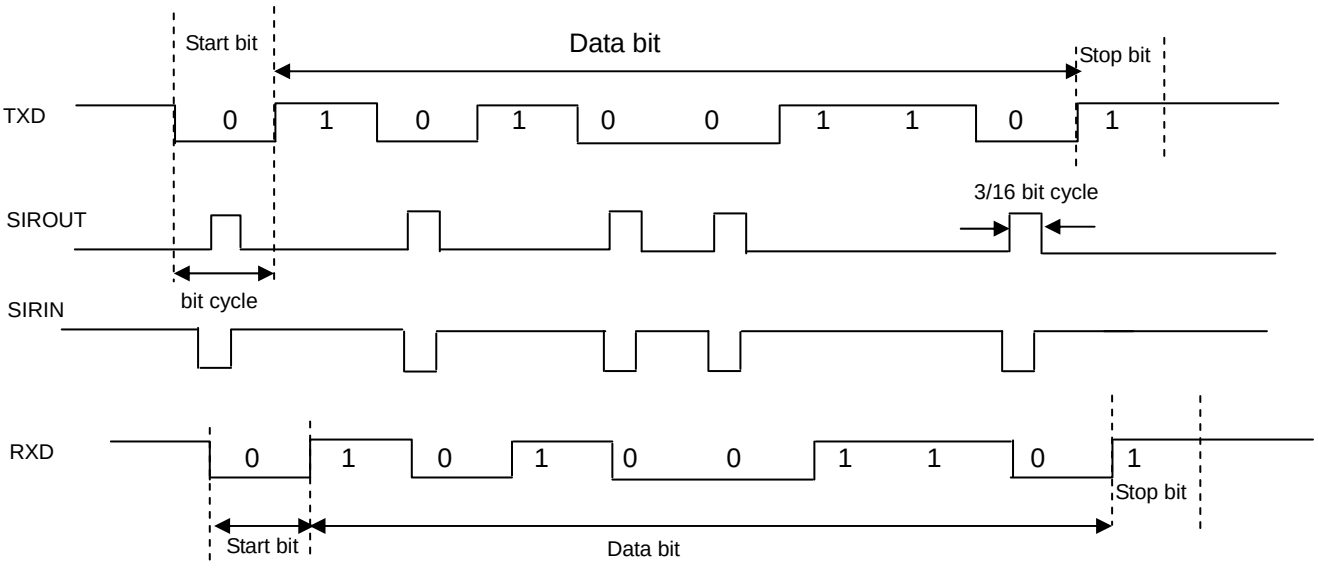


Figure 3.13.4 IrDA Data Modulation

(11) Hardware flow control

The hardware flow control feature is fully selectable, and enables you to control the serial data flow by using the UxRTSn output and UxCTSn input signals.

Figure 3.13.5 shows how the two devices can communicate with each other using hardware flow control.

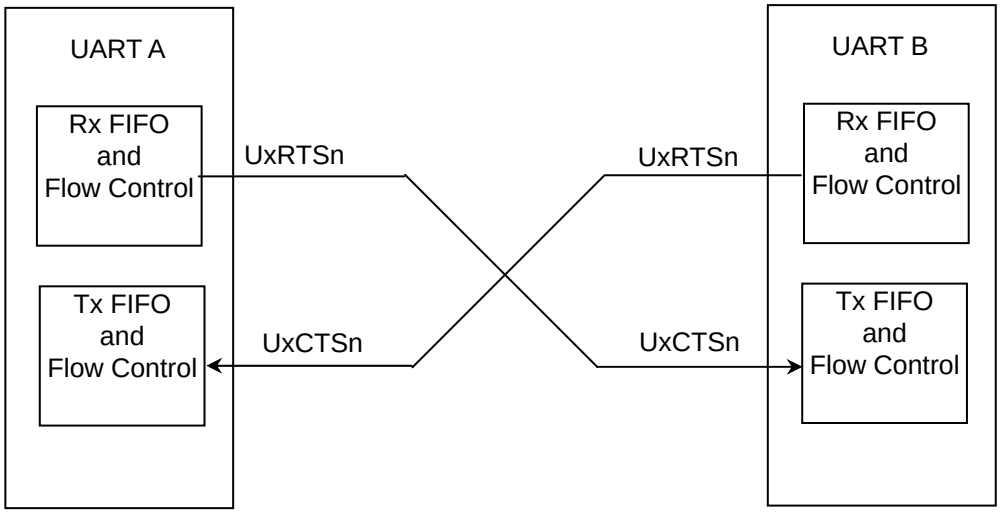


Figure 3.13.5 Hardware Flow Control

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 amount of data stored in the receive FIFO exceeds watermark level, the UxRTSn signal is deasserted, indicating that there is no more room to receive.

The UxRTSn signal is reasserted when data has been read out of the receive FIFO and it is filled to less than the watermark level.

Even if RTS flow control is disabled, communication can be enabled.

CTS flow control

If CTS flow control is enabled, then the transmitter checks the UxCTSn signal before transmitting. If the UxCTSn signal is asserted, it transmits the byte, otherwise transmission does not occur.

The data transmission continues while UxCTSn is asserted and the transmit FIFO is not empty. If the transmit FIFO is empty, no data is transmitted even when the UxCTSn signal is asserted.

If the UxCTSn signal is deasserted while CTS flow control is enabled, the current data transmission is completed before stopping.

Even if CTS flow control is disabled, communication can be enabled.

Table 3.13.1 Control Bits to Enable and Disable Hardware Flow Control

UARTxCR		nUARTRTS	Description
CTSEn	RTSEn		
1	1	0 (Note)	Both RTS and CTS flow controls enabled
1	0	1	Only CTS flow control enabled
0	1	0(Note)	Only RTS flow control enabled
0	0	1	Both RTS and CTS flow controls disabled

Note: During in the RTSEn=1(Enable), the nUARTRTS is set to 0(Enable) until the receive FIFO is filled up to the watermark level.

3.13.2 Register Descriptions

The following lists the SFRs.:

• UART0 Base address = 0xF200_0000

Register Name	Address (base +)	Description
UART0DR	0x000	UART0 Data register
UART0SR/ UART0ECR	0x004	UART0 Receive status register/ UART0 error clear register
–	0x008-0x014	Reserved
UART0FR	0x018	UART0 Flag register
–	0x01C	Reserved
UART0ILPR	0x020	UART0 IrDA low-power counter register
UART0IBRD	0x024	UART0 Integer baud rate register
UART0FBRD	0x028	UART0 Fractional baud rate register
UART0LCR_H	0x02C	UART0 Line control register
UART0CR	0x030	UART0 Control register
UART0IFLS	0x034	UART0 Interrupt FIFO level select register
UART0IMSC	0x038	UART0 Interrupt mask set/clear register
UART0RIS	0x03C	UART0 Raw interrupt status register
UART0MIS	0x040	UART0 Masked interrupt status register
UART0ICR	0x044	UART0 Interrupt clear register
UART0DMACR	0x048	UART0 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 control registers are reprogrammed. When the UART is disabled in the middle of transmit or receive operation, it stops after the transmission of the current character is completed.

•UART1

Base address = 0xF200_1000

Register Name	Address (base +)	Description
UART1DR	0x0000	UART1 Data register
UART1SR/ UART1ECR	0x0004	UART1 Receive status register/ UART1 error clear register
–	0x0008-0x0014	Reserved
UART1FR	0x0018	UART1 Flag register
–	0x001C	Reserved
–	0x0020	Reserved
UART1IBRD	0x0024	UART1 Integer baud rate register
UART1FBRD	0x0028	UART1 Fractional baud rate register
UART1LCR_H	0x002C	UART1 Line control register
UART1CR	0x0030	UART1 Control register
UART1IFLS	0x0034	UART1 Interrupt FIFO level select register
UART1IMSC	0x0038	UART1 Interrupt mask set/clear register
UART1RIS	0x003C	UART1 Raw interrupt status register
UART1MIS	0x0040	UART1 Masked interrupt status register
UART1ICR	0x0044	UART1 Interrupt clear register
–	0x0048	Reserved
–	0x004C-0x007C	Reserved
–	0x0080-0x008C	Reserved
–	0x0090-0x0FCC	Reserved
–	0x0FD0-0x0FDC	Reserved
–	0x0FE0	Reserved
–	0x0FE4	Reserved
–	0x0FE8	Reserved
–	0x0FEC	Reserved
–	0x0FF0	Reserved
–	0x0FF4	Reserved
–	0x0FF8	Reserved
–	0x0FFC	Reserved

Note: You must disable the UART before any of the control registers are reprogrammed. When the UART is disabled in the middle of transmit or receive operation, it stops after the transmission of the current character is completed.

1. UART0DR (UART0 Data Register)

Address = (0xF200_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read as undefined. Write as zero.
[11]	OE	RO	Undefined	Overrun error Read : 0y0: There is an empty space in the FIFO. 0y1: Overrun error flag Write: Invalid
[10]	BE	RO	Undefined	Break error Read : 0y0: No error detected 0y1: Error detected Write: Invalid
[9]	PE	RO	Undefined	Parity error Read : 0y0: No error detected 0y1: Error detected Write: Invalid
[8]	FE	RO	Undefined	Framing error Read : 0y0: No error detected 0y1: Error detected Write: Invalid
[7:0]	DATA	R/W	Undefined	Read: Receive data Write: Transmit data

2. UART1DR (UART1 Data Register)

Address = (0xF200_1000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read as undefined. Write as zero.
[11]	OE	RO	Undefined	Overrun error Read: 0y0: There is an empty space in the FIFO. 0y1: Overrun error flag Write: Invalid
[10]	BE	RO	Undefined	Break error Read: 0y0: No error detected 0y1: Error detected Write: Invalid
[9]	PE	RO	Undefined	Parity error Read: 0y0: No error detected 0y1: Error detected Write: Invalid
[8]	FE	RO	Undefined	Framing error Read: 0y0: No error detected 0y1: Error detected Write: Invalid
[7:0]	DATA	R/W	Undefined	Read: Receive data Write: Transmit data

[Description]

a. <OE>

This bit is set to 1 if data is received and the receive FIFO is already full. In this case, the received data is not stored in the FIFO and is discarded.

The bit is cleared to 0 once an empty space is made in the FIFO and a new data can be written to it.

b. <BE>

This bit is set to 1 if a break condition was detected, indicating that the receive data input (defined as start, data parity, and stop bits) was held Low for a period longer than a full-word transmission time.

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 of 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).

3. UART0SR/UART0ECR (UART0 Receive status register/ UART0 error clear register)

UART0SR and UART0ECR are mapped to same address.

These functions differ in read and write operations.

Address = (0xF200_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	–	Read as undefined.
[3]	OE	RO	0y0	Overrun error: 0y0: There is an empty space in the FIFO. 0y1: Overrun error flag
[2]	BE	RO	0y0	Break error 0y0: No error detected 0y1: Error detected
[1]	PE	RO	0y0	Parity error 0y0: No error detected 0y1: Error detected
[0]	FE	RO	0y0	Framing error 0y0: No error detected 0y1: Error detected

Address = (0xF200_1000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	WO	–	A write to this register clears framing, parity, break, and overrun errors. The data value has no significance. The address of this register is the same as that of the UART0SR register.

4. UART1SR/ UART1ECR (UART1 Receive status register/ UART1 error clear register)

Address = (0xF200_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	–	Read as undefined.
[3]	OE	RO	0y0	Overrun error: 0y0: There is an empty space in the FIFO. 0y1: Overrun error flag
[2]	BE	RO	0y0	Break error 0y0: No error detected 0y1: Error detected
[1]	PE	RO	0y0	Parity error 0y0: No error detected 0y1: Error detected
[0]	FE	RO	0y0	Framing error 0y0: No error detected 0y1: Error detected

Address = (0xF200_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	–	WO	–	A write to this register clears framing, parity, break, and overrun errors. The data value has no significance. The address of this register is the same as that of the UART1SR 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, 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 receive data must be read first from UARTxDR before the error status associated with that data is read from UARTxSR. This read sequence cannot be reversed because the status register UARTxSR is updated only when the data is read from the data register UARTxDR. The status information can also be read directly from the UARTxDR register.

[Description]

a. <OE>

This bit is set to 1 if data is received and the FIFO is already full. In this case, the received data is not stored in the FIFO and is discarded.

The bit is cleared to 0 once an empty space is made 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 receive data input (defined as start, data parity, and stop bits) was held Low for longer than a full-word transmission time.

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 of 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).

5. UART0FR (UART0 Flag register)

The <TXFE>, <RXFF>, <TXFF>, and <RXFE> bits differ depending on the state of the <FEN> of the UART0LCR_H register.

(1) Transmit FIFO

The transmit FIFO is an 8-bit wide, 16-location 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. The transmit FIFO can be disabled to act like a one-byte holding register.

(2) Receive FIFO

The receive FIFO is a 12-bit wide, 16-location deep, FIFO memory buffer. Received data and corresponding error bits are stored in the receive FIFO by the receive logic until they are read out by the CPU across the APB interface. The receive FIFO can be disabled to act like a one-byte holding register.

Address = (0xF200_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description	
				FIFO mode (FEN = 1)	Character mode (FEN = 0)
[31:9]	–	–	Undefined	Read as undefined.	Read as undefined.
[8]	Reserved	RO	Undefined	Read as undefined.	Read as undefined.
[7]	TXFE	RO	0y1	Transmit FIFO empty flag 0y0: Not empty 0y1: Empty	Transmit hold register empty flag 0y0: Not empty 0y1: Empty
[6]	RXFF	RO	0y0	Receive FIFO full flag 0y0: Not full 0y1: Full	Receive hold register full flag 0y0: Not full 0y1: Full
[5]	TXFF	RO	0y0	Transmit FIFO full flag 0y0: Not full 0y1: Full	Transmit hold register full flag 0y0: Not full 0y1: Full
[4]	RXFE	RO	0y1	Receive FIFO empty flag 0y0: Not empty 0y1: Empty	Receive hold register empty flag 0y0: Not empty 0y1: Empty
[3]	BUSY	RO	0y0	BUSY flag 0y0: The UART has stopped transmitting data 0y1: The UART is transmitting data. (BUSY).	BUSY flag: 0y0:The UART has stopped transmitting data.0 y1:The UART is transmitting data. (BUSY)
[2]	Reserved	RO	Undefined	Read as undefined.	Read as undefined.
[1]	Reserved	RO	Undefined	Read as undefined.	Read as undefined.
[0]	Reserved	RO	Undefined	Read as undefined.	Read as undefined.

6. UART1FR (UART1 Flag register)

Address = (0xF200_1000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description	
				FIFO mode (FEN = 1)	Character mode (FEN = 0)
[31:8]	–	–	Undefined	Read as undefined.	Read as undefined.
[7]	TXFE	RO	0y1	Transmit FIFO empty flag 0y0: Not empty 0y1: Empty	Transmit hold register empty flag 0y0: Not empty 0y1: Empty
[6]	RXFF	RO	0y0	Receive FIFO full flag 0y0: Not full 0y1: Full	Receive hold register full flag 0y0: Not full 0y1: Full
[5]	TXFF	RO	0y0	Transmit FIFO full flag 0y0: Not full 0y1: Full	Transmit hold register full flag 0y0: Not full 0y1: Full
[4]	RXFE	RO	0y1	Receive FIFO full flag 0y0: Not empty 0y1: Empty	Receive hold register empty flag 0y0: Not empty 0y1: Empty
[3]	BUSY	RO	0y0	BUSY flag 0y0: The UART has stopped transmitting data 0y1: The UART is transmitting data. (BUSY)	BUSY flag 0y0:The UART has stopped transmitting data 0y1:The UART is transmitting data. (BUSY)
[2:1]	–	–	Undefined	Read as undefined.	Read as undefined.
[0]	CTS	RO	Undefined	Clear To Send (CTS) flag 0y1: Modem status input = 0	Clear To Send (CTS) flag 0y1 : Modem status input = 0

[Description]

a. <RI>

Ring indicator (nUART1RI): This bit is set to 1 when the modem status input is 0.

b. <BUSY>

This bit is set to 1 when the UART is transmitting data. This bit remains set until the complete data, including all the stop bits, has been sent from the shift register.

c. <DCD>

Data carrier detect (U0DCDn): This bit is set to 1 when the modem status input is 0.

d. <DSR>

UART data set ready (U0DSRn): This bit is set to 1 when the modem status input is 0.

e. <CTS>

Clear to send (U0CTSn): This bit is set to 1 when the modem status input is 0.

7. UART0ILPR (UART0 IrDA low-power counter register)

Address = (0xF200_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	ILPDVSR	R/W	0x00	IrDA low-power divisor: 0x01 to 0xFF

[Description]

a. <ILPDVSR>

Low-power divisor (ILPDVSR) = ($f_{UARTCLK}$ / $f_{IrLPBaud16}$)

The UART0ILPR register is the IrDA low-power counter register. This is an 8-bit read/write register that stores the low-power counter divisor value used to generate the IrLPBaud16 signal by dividing down of UARTCLK. All the bits are cleared to 0 when reset.

Note 1: Set this register before the UART0CR<SIRLP> is set to 1.

Note 2: 0x00 value is invalid. (the IrLPBaud16 pulse is not generated)

8. UART0IBRD (UART0 Integer baud rate register)

Address = (0xF200_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	BAUD DIVINT	R/W	0x0000	Integer part of baud rate divisor: 0x0001 to 0xFFFF

9. UART1IBRD (UART1 Integer baud rate register)

Address = (0xF200_1000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	BAUD DIVINT	R/W	0x0000	Integer part of baud rate divisor: 0x0001 to 0xFFFF

[Description]

a. <BAUD DIVINT>

This register, when put together with the fractional baud rate divisor described next, provides the baud rate divisor BAUDDIV.

Note1: To update the contents of UARTxIBRD internally, the write to UARTxLCR_H must always be executed last. For details, refer to the description of UARTxLCR_H.

Note 2: Set this register before the UARTxCR<UARTEN> is set to 1.

Note3: 0x0000 setting is prohibited.

10. UART0FBRD (UART0 Fractional baud rate register)

Address = (0xF200_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:0]	BAUD DIVFRAC	R/W	0x00	Fractional part of baud rate divisor: 0x01 to 0x3F

11. UART1FBRD (UART1 Fractional baud rate register)

Address = (0xF200_1000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:0]	BAUD DIVFRAC	R/W	0x00	Fractional part of baud rate divisor: 0x01 to 0x3F

[Description]

a. <BAUDDIVFRAC>

The baud rate divisor is calculated as follows:

Baud rate divisor BAUDDIV = ($f_{UARTCLK}$)/ (16 × baud rate)

$f_{UARTCLK}$ is the frequency of UARTCLK.

The BAUDDIV is comprised of the integer value (BAUD DIVINT) and the fractional value (BAUD DIVFRAC).

Note1: To update the contents of UARTxFBRD internally, the write to UARTxLCR_H must always be executed last. For details, refer to the description of UARTxLCR_H.

Note 2: Set this register before the UARTxCR<UARTEN> is set to 1.

Note 3: The Integer part of baud rate divisor setting is invalid to 0. Therefore, the baud rate setting can not be set only the fractional baud rate divisor. And please pay attention that the minimum baud rate divisor setting are "1" as the integer and also "1" as the decimal fraction (which can not set to "0" when the integer baud rate value is "1").

Example: Calculating the divisor value

When the required baud rate is **230400** and $f_{UARTCLK}$ = **4 MHz**:

Baud rate divisor = $(\mathbf{4 \times 10^6})$/ (16 × **230400**) = **1.085**

Therefore, BRDI = **1** and BRDF = **0.085**

Fractional part is ((**0.085** × 64) + 0.5) = 5.94.

The integer part of this, 0x5, should be set as the fractional baud rate divisor value.

Generated baud rate divisor = **1** + **5**/64 = **1.078**

Generated baud rate = $(\mathbf{4 \times 10^6})$/ (16 × **1.078**) = 231911

Error = (231911 – 230400)/ 230400 × 100 = 0.656 %

The maximum error using a 6-bit UARTxFBRD register = 1/64 × 100 = 1.56 %

This error occurs when m = 1, and it is cumulative over 64 clock ticks.

Typical baud rate setting examples

$f_{UARTCLK}$ = 100 MHz

Programmed divisor (integer)	Programmed divisor (fraction)	Required bit rate (bps)	Generated bit rate (bps)	Error (%)
0x1	0x1	—	6153846 (fastest)	—
0xD	0x24	460800	460829.493	0.0064
0x1B	0x8	230400	230414.747	0.0064
0x36	0x10	115200	115207.373	0.0064
0x51	0x18	76800	76804.916	0.0064
0x6C	0x20	57600	57603.687	0.0064
0xA2	0x31	38400	38398.771	-0.0032
0x145	0x21	19200	19200.307	0.0016
0x1B2	0x2	14400	14399.885	-0.0008
0x28B	0x3	9600	9599.923	-0.0008
0xA2C	0xB	2400	2399.995	-0.0002
0x1458	0x15	1200	1200.001	0.0001
0xDDF2	0xC	110	109.99999	-1.00E-05

$f_{UARTCLK}$ = 96 MHz

Programmed divisor (integer)	Programmed divisor (fraction)	Required bit rate (bps)	Generated bit rate (bps)	Error (%)
0x1	0x1	—	5907692 (fastest)	—
0xD	0x1	460800	460984.394	0.0400
0x1A	0x3	230400	230353.929	-0.0200
0x34	0x5	115200	115211.521	0.0100
0x4E	0x8	76800	76800.000	0
0x68	0xB	57600	57597.120	-0.0050
0x9C	0x10	38400	38400.000	0
0x138	0x20	19200	19200.000	0
0x1A0	0x2B	14400	14399.820	-0.0012
0x271	0x1	9600	9599.760	-0.0025
0x9C4	0x1	2400	2399.985	-0.0006
0x1388	0x1	1200	1199.996	-0.0003
0xD511	0x1D	110	110.000	2.60E-06

$f_{UARTCLK}$ = 25MHz

Programmed divisor (integer)	Programmed divisor (fraction)	Required bit rate (bps)	Generated bit rate (bps)	Error (%)
0x1	0x1	—	1538461(fastest)	—
0x3	0x19	460800	460829.493	0.0064
0x6	0x32	230400	230414.747	0.0064
0xD	0x24	115200	115207.373	0.0064
0x14	0x16	76800	76804.916	0.0064
0x1B	0x8	57600	57603.687	0.0064
0x28	0x2C	38400	38402.458	0.0064
0x51	0x18	19200	19201.229	0.0064
0x6C	0x20	14400	14400.922	0.0064
0xA2	0x31	9600	9599.693	-0.0032
0x28B	0x3	2400	2399.981	-0.0008
0x516	0x5	1200	1200.005	0.0004
0x377C	0x23	110	110.000	-1.00E-05

12. UART0LCR_H (UART0 Line control register)

Address = (0xF200_0000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	SPS	R/W	0y0	Stick parity select: Refer to Table 3.13.2 for the 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 0y0: Character mode 0y1: FIFO mode
[3]	STP2	R/W	0y0	Stop bit select 0y0: 1 bit 0y1: 2 bits
[2]	EPS	R/W	0y0	Even parity select (Refer to Table 3.13.2 for the truth table.) 0y0: Odd 0y1: Even
[1]	PEN	R/W	0y0	Parity control (Refer to Table 3.13.2 for the truth table.) 0y0: Disable 0y1: Enable
[0]	BRK	R/W	0y0	Send break 0y0: No effect 0y1: Send break

13. UART1LCR_H (UART1 Line control register)

Address = (0xF200_1000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	SPS	R/W	0y0	Stick parity select: Refer to Table 3.13.2 for the 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 0y0: Character mode 0y1: FIFO mode
[3]	STP2	R/W	0y0	Stop bit select 0y0: 1 stop bit 0y1: 2 stop bits
[2]	EPS	R/W	0y0	Even parity select (Refer to Table 3.13.2 for the truth table.) 0y0: Odd 0y1: Even
[1]	PEN	R/W	0y0	Parity control (Refer to Table 3.13.2 for the truth table.) 0y0: Disable 0y1: Enable
[0]	BRK	R/W	0y0	Send break 0y0: No effect 0y1: Send break

[Description]

a. <SPS>

When bits 1, 2, and 7 of the UARTxLCR_H register are set, the parity bit is transmitted and checked as a 0. When bits 1 and 7 are set and bit 2 is 0, the parity bit is transmitted and checked as a 1. When this bit is cleared, the stick parity is disabled. Refer to Table 3.13.2 for the truth table of SPS, EPS, and PEN bits.

b. <WLEN>

This bit indicates the number of data bits transmitted or received in a frame.

c. <FEN>

When this bit is set to 1, transmit and receive FIFO buffers are enabled (FIFO mode).

When this bit is cleared to 0, the FIFOs are disabled (character mode) and they become 1-byte deep holding registers.

d. <STP2>

When this bit is set to 1, two stop bits are transmitted at the end of a frame. The receive logic does not check for the second stop bit being received.

e. <EPS>

When this bit is set to 1, even parity generation and checking are performed during transmission and reception. This function checks whether the number of 1s contained in the data bits and parity bit is even. When this bit is cleared to 0, odd parity check is performed to check whether the number of 1s is odd. This bit has no effect when parity is disabled by Parity Enable bit <PEN> being cleared to 0. Refer to Table 3.13.2 for the truth table.

f. <PEN>

When this bit is set to 1, parity check and generation are enabled. Otherwise, parity is disabled and no parity bit is added to data frames. Refer to Table 3.13.2 for the truth table of SPS, EPS, and PEN bits.

g. <BRK>

When this bit is set to 1, the UxTXD output remains LOW after the current character is transmitted. For generation of the transmit break condition, this bit must be asserted while at least one frame is or longer being transmitted. Even when the break condition is generated, the contents of the transmit FIFO are not affected.

Note: When you set UARTxLCR_H, UARTxIBRD and UARTxFBRD, UARTxLCR_H must be set at the end.
When you update only UARTxIBRD or UARTxFBRD, UARTxLCR_H register must be set again.

Table 3.13.2 is the truth table of the <SPS>, <EPS> and <PEN> bits of the UARTxLCR_H register.

Table 3.13.2 Truth table of 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

14. UART0CR (UART0 Control register)

Address = (0xF200_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[14]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[13:12]	–	–	Undefined	Read as undefined. Write as zero
[11]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[10]	Reserved	R/W	0y0	Read as 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]	Reserved	R/W	0y0	Write as zero.
[6:3]	Reserved	–	Undefined	Read as undefined. Write as zero
[2]	SIRLP	R/W	0y0	IrDA encoding mode select for transmitting 0 bits 0y0: 0 bits are transmitted as an active high pulse of 3/16th of the bit period. 0y1:0 bits are transmitted with a pulse width that is 3 times the period of the IrLPBaud16 input signal.
[1]	SIREN	R/W	0y0	SIR enable 0y0: Disable 0y1: Enable
[0]	UARTEN	R/W	0y0	UART enable 0y0: Disable 0y1: Enable

15. UART1CR (UART1 control register)

Address = (0xF200_1000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	CTSEN	R/W	0y0	CTS hardware flow control enable 0y0: Disable 0y1: Enable
[14:10]	–	–	Undefined	Read as 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]	Reserved	R/W	0y0	Write as zero.
[6:1]	Reserved	–	Undefined	Read as undefined. Write as zero.
[0]	UARTEN	R/W	0y0	UART enable 0y0: Disable 0y1: Enable

[Description]

a. <CTSEN>

When this bit is set to 1, CTS hardware flow control is enabled. Data is transmitted only after the UxCTSn signal has been asserted.

b. <RTSEN>

When this bit is set to 1, RTS hardware flow control is enabled. Data is transmitted only when there is an empty space in the receive FIFO.

c. <RTS>

This bit is the UART Request To Send (UxRTSn) modem status output signal. When this bit is set to 1, the output is 0.

d. <DTR>

This bit is the UART Data Transmit Ready (UxDTRn) modem status output signal. When this bit is set to 1, the output is 0.

e. <RXE>

When this bit is set to 1, the receive circuit of the UART is enabled. Data reception occurs for either UART function or SIR function according to the setting of <SIREN>. When the UART is disabled in the middle of receive operation, it completes current reception and the subsequent receptions are disabled.

f. <TXE>

When this bit is set to 1, the transmit circuit of the UART is enabled. Data transmission occurs for either UART function or SIR function according to the setting of <SIREN>. When the UART is disabled in the middle of transmit operation, it completes the current transmission before stopping.

g. <SIRLP>

<SIRLP> selects IrDA encoding mode. When this bit is cleared to 0, 0 bits of the IrDA transmission data are transmitted as an active high pulse (SIROUT) with a width of 3/16th of the bit period. When this bit is set to 1, 0 bits of the IrDA transmission data are transmitted with a pulse width that is 3 times the period of the IrLPBaud16 input signal. Setting this bit can reduce power consumption but might decrease transmission distances.

h. <SIREN>

When this bit is set to 1, the IrDA circuit is enabled. To use the UART, the <UARTEN> must be set to 1. When the IrDA circuit is enabled, the SIR0OUT and SIR0IN pins are enabled. The U0TXD pin remains in the marking state (set to 1). Signal transitions on the U0RXD pin or modem status input have no effect. When IrDA circuit is disabled, SIR0OUT remains cleared to 0 (no light pulse is generated) and the SIR0IN pin has no effect.

i. <UARTEN>

When this bit is set to 1, the UART is enabled. Data transmission and reception occur for either UART function or SIR function according to the setting of <SIREN>. When the UART is disabled in the middle of transmit or receive operation, it completes current transmission or reception before stopping.

16. UART0IFLS (UART0 Interrupt FIFO level select register)

Address = (0xF200_0000) + (0x0034)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:3]	RXIFLSEL	R/W	0y010	Receive interrupt FIFO level select (1 word = 12 bits): 0y000: When the 2nd word has been stored in receive FIFO 0y001: When the 4th word has been stored in receive FIFO 0y010: When the 8th word has been stored in receive FIFO 0y011: When the 12th word has been stored in receive FIFO 0y100: When the 14th word has been stored in receive FIFO 0y101to 0y111: Reserved
[2:0]	TXIFLSEL	R/W	0y010	Transmit FIFO level select (1 word = 8 bits): 0y000: When transmit FIFO has space for 2 words left 0y001: When transmit FIFO has space for 4 words left 0y010: When transmit FIFO has space for 8 words left 0y011: When transmit FIFO has space for 12 words left 0y100: When transmit FIFO has space for 14 words left 0y101 to 0y111: Reserved

17. UART1IFLS (UART1 Interrupt FIFO level select register)

Address = (0xF200_1000) + 0x0034

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:3]	RXIFLSEL	R/W	0y010	Receive interrupt FIFO level select (1 word = 12 bits): 0y000: When the 2nd word has been stored in receive FIFO 0y001: When the 4th word has been stored in receive FIFO 0y010: When the 8th word has been stored in receive FIFO 0y011: When the 12th word has been stored in receive FIFO 0y100: When the 14th word has been stored in receive FIFO 0y101 to 0y111: Reserved
[2:0]	TXIFLSEL	R/W	0y010	Transmit interrupt FIFO level select (1 word = 8 bits): 0y000: When transmit FIFO has space for 2 words left 0y001: When transmit FIFO has space for 4 words left 0y010: When transmit FIFO has space for 8 words left 0y011: When transmit FIFO has space for 12 words left 0y100: When transmit FIFO has space for 14 words left 0y101 to 0y111: Reserved

[Description]

The UARTxIFLS register is the interrupt FIFO level select register. This register is used to define the FIFO level at which UARTTXINTR and UARTRXINTR are generated.

The interrupts are generated based on a transition through a level rather than based on the level. For example, an interrupt is generated at a point when the third word has been stored in the receive FIFO which contained two words.

18. UART0IMSC (UART0 Interrupt mask set/clear register)

Address = (0xF200_0000) + (0x0038)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined. Write as zero.
[10]	OEIM	R/W	0y0	Overrun error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[9]	BEIM	R/W	0y0	Break error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[8]	PEIM	R/W	0y0	Parity error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[7]	FEIM	R/W	0y0	Framing error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[6]	RTIM	R/W	0y0	Receive timeout interrupt mask 0y0: Clear the mask 0y1: Set the mask
[5]	TXIM	R/W	0y0	Transmit FIFO interrupt mask 0y0: Clear the mask 0y1: Set the mask
[4]	RXIM	R/W	0y0	Receive FIFO interrupt mask 0y0: Clear the mask 0y1: Set the mask
[3]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[2]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[1]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[0]	Reserved	R/W	0y0	Read as undefined. Write as zero.

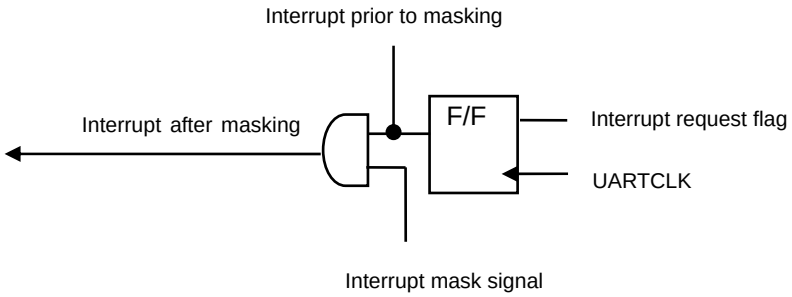
19. UART1IMSC (UART1 Interrupt mask set/clear register)

Address = (0xF200_1000) + (0x0038)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined. Write as zero.
[10]	OEIM	R/W	0y0	Overrun error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[9]	BEIM	R/W	0y0	Break error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[8]	PEIM	R/W	0y0	Parity error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[7]	FEIM	R/W	0y0	Framing error interrupt mask 0y0: Clear the mask 0y1: Set the mask
[6]	RTIM	R/W	0y0	Receive timeout interrupt mask 0y0: Clear the mask 0y1: Set the mask
[5]	TXIM	R/W	0y0	Transmit interrupt mask 0y0: Clear the mask 0y1: Set the mask
[4]	RXIM	R/W	0y0	Receive interrupt mask 0y0: Clear the mask 0y1: Set the mask
[3:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	CTSMIM	R/W	0y0	U0CTSn interrupt mask 0y0: Clear the mask 0y1: Set the mask
[0]	–	–	Undefined	Read as undefined. Write as zero.

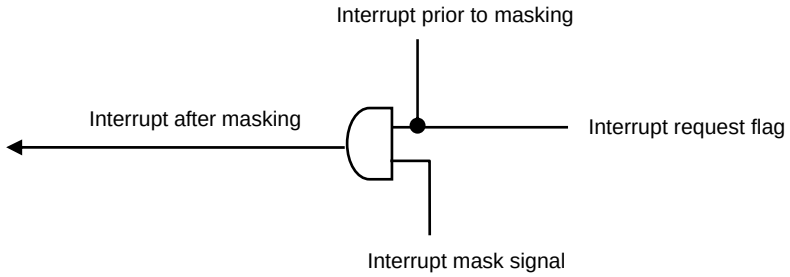
- UART interrupt generation block diagrams

(1) Block diagram of the break error (BE), parity error (PE) and framing error (PE) flags



- The interrupt request flag state changes in real time and is retained in the F/F. Each flag can be cleared by a write to the corresponding bit in the interrupt clear register.

(2) Block diagram of the overrun error (OE) flag



- The interrupt request flag state by the overrun error (OE) flag changes in real time and its state is not retained. And the OE flag is cleared by a read of the receive FIFO.

20. UART0RIS (UART0 Raw interrupt status register)

Address = (0xF200_0000) + (0x003C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined.
[10]	OERIS	RO	0y0	Overrun error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[9]	BERIS	RO	0y0	Break error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[8]	PERIS	RO	0y0	Parity error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[7]	FERIS	RO	0y0	Framing error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[6]	RTRIS	RO	0y0	Receive timeout raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[5]	TXRIS	RO	0y0	Transmit raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[4]	RXRIS	RO	0y0	Receive raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[3]	Reserved	RO	Undefined	Read as undefined.
[2]	Reserved	RO	Undefined	Read as undefined.
[1]	Reserved	RO	Undefined	Read as undefined.
[0]	Reserved	RO	Undefined	Read as undefined.

Note: All the bits, except the modem raw status interrupt bits (bits 3 to 0), are cleared to 0 when reset. The modem status bits are undefined after reset.

21. UART1RIS (UART1 Raw interrupt status register)

Address = (0xF200_1000) + (0x003C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined.
[10]	OERIS	RO	0y0	Overrun error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[9]	BERIS	RO	0y0	Break error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[8]	PERIS	RO	0y0	Parity error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[7]	FERIS	RO	0y0	Framing error raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[6]	RTRIS	RO	0y0	Receive timeout raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[5]	TXRIS	RO	0y0	Transmit raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[4]	RXRIS	RO	0y0	Receive raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[3:2]	–	–	Undefined	Read as undefined.
[1]	CTSRMIS	RO	Undefined	U0CTSn modem raw interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[0]	–	–	Undefined	Read as undefined.

Note: All the bits, except the modem raw status interrupt bits (bits 1), are cleared to 0 when reset. The modem status bits are undefined after reset.

22. UART0MIS (UART0 Masked interrupt status register)

Address = (0xF200_0000) + (0x0040)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined.
[10]	OEMIS	RO	0y0	Overrun error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[9]	BEMIS	RO	0y0	Break error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[8]	PEMIS	RO	0y0	Parity error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[7]	FEMIS	RO	0y0	Framing error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[6]	RTMIS	RO	0y0	Receive timeout masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[5]	TXMIS	RO	0y0	Transmit masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[4]	RXMIS	RO	0y0	Receive masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[3]	Reserved	RO	Undefined	Read as undefined.
[2]	Reserved	RO	Undefined	Read as undefined.
[1]	Reserved	RO	Undefined	Read as undefined.
[0]	Reserved	RO	Undefined	Read as undefined.

Note: All the bits, except the modem masked status interrupt bits (bits 3 to 0), are cleared to 0 when reset. The modem status bits are undefined after reset.

23. UART1MIS (UART1 Masked interrupt status register)

Address = (0xF200_1000) + (0x0040)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined.
[10]	OEMIS	RO	0y0	Overrun error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[9]	BEMIS	RO	0y0	Break error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[8]	PEMIS	RO	0y0	Parity error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[7]	FEMIS	RO	0y0	Framing error masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[6]	RTMIS	RO	0y0	Receive timeout masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[5]	TXMIS	RO	0y0	Transmit masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[4]	RXMIS	RO	0y0	Receive masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[3:2]	–	–	Undefined	Read as undefined.
[1]	CTSMMIS	RO	Undefined	U0CTSn masked interrupt status 0y0: Interrupt not requested. 0y1: Interrupt requested.
[0]	–	–	Undefined	Read as undefined.

24. UART0ICR (UART0 Interrupt clear register)

Address = (0xF200_0000) + (0x0044)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined. Write as zero.
[10]	OEIC	WO	Undefined	Overrun error interrupt clear 0y0: Invalid 0y1: Clear
[9]	BEIC	WO	Undefined	Break error interrupt clear 0y0: Invalid 0y1: Clear
[8]	PEIC	WO	Undefined	Parity error interrupt clear 0y0: Invalid 0y1: Clear
[7]	FEIC	WO	Undefined	Framing error interrupt clear 0y0: Invalid . 0y1: Clear.
[6]	RTIC	WO	Undefined	Receive timeout interrupt clear 0y0: Invalid 0y1: Clear
[5]	TXIC	WO	Undefined	Transmit interrupt clear 0y0: Invalid 0y1: Clear
[4]	RXIC	WO	Undefined	Receive interrupt clear 0y0: Invalid 0y1: Clear
[3]	Reserved	WO	Undefined	Write as zero.
[2]	Reserved	WO	Undefined	Write as zero.
[1]	Reserved	WO	Undefined	Write as zero.
[0]	Reserved	WO	Undefined	Write as zero.

Note: The UART0ICR register is a write-only interrupt clear register. When a bit of this register is set to 1, the associated interrupt is cleared. A write of 0 to any bit of this register is invalid.

25. UART1ICR (UART1 Interrupt clear register)

Address = (0xF200_1000) + (0x0044)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined. Write as zero.
[10]	OEIC	WO	Undefined	Overrun error interrupt clear 0y0: Invalid 0y1: Clear
[9]	BEIC	WO	Undefined	Break error interrupt clear 0y0: Invalid 0y1: Clear
[8]	PEIC	WO	Undefined	Parity error interrupt clear 0y0: Invalid 0y1: Clear
[7]	FEIC	WO	Undefined	Framing error interrupt clear 0y0 Invalid 0y1: Clear
[6]	RTIC	WO	Undefined	Receive timeout interrupt clear 0y0: Invalid 0y1: Clear
[5]	TXIC	WO	Undefined	Transmit interrupt clear 0y0: Invalid 0y1: Clear
[4]	RXIC	WO	Undefined	Receive interrupt clear 0y0: Invalid 0y1: Clear
[3:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	CTSMIC	WO	Undefined	U0CTSn interrupt clear 0y0: Invalid 0y1: Clear
[0]	–	–	Undefined	Read as undefined. Write as zero.

Note: The UART1ICR register is a write-only interrupt clear register. When a bit of this register is set to 1, the associated interrupt is cleared. A write of 0 to any bit of this register is invalid.

26. UART0DMACR (UART0 DMA control register)

Address = (0xF200_0000) + (0x0048)

Bit	Bit Symbol	Type	Reset Value	Description
[31:3]	–	–	–	Read as undefined. Write as zero.
[2]	DMAONERR	R/W	0y0	DMA on error 0y1: Available 0y0: Not available
[1]	TXDMAE	R/W	0y0	Transmit FIFO DMA enable 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	Receive FFO DMA enable 0y0: Disable 0y1: Enable

Note1: For example, if 19 characters have to be received and the watermark level is programmed to be four, then the DMA controller transfers four bursts of four characters and three single transfers to complete the stream.

Note2: The bus width must be set to 8-bits, if you transfer the data of tranmit/ receive FIFO by using DMAC.

[Description]

a. <DMAONERR>

When this bit is set to 1, the DMA receive request output, UARTxRXDMASREQ or UARTxRXDMABREQ, is disabled on assertion of a UART error interrupt.

3.14 I²C

3.14.1 Overview

This module operates in I²C bus mode compliant with the typical I²C bus standard (Philips specifications). (Note 1)

The main features are as follows:

- Contains one channel (ch0).
- Allows selection between master and slave.
- Allows selection between transmission and reception.
- Supports multiple masters (arbitration, clock synchronization recognition).
- Supports standard mode and fast mode (fastest baud rate in master mode: 89.91 kHz and 357.14 kHz, respectively, at f_{PCLK} = 100 MHz)
- Supports the addressing format of 7 bits only.
- Supports transfer data sizes of 1 to 8 bits.
- Provides one transfer (transmission or reception) complete interrupt (level-sensitive).
- Can enable or disable interrupts. (Interrupt source for I²C ch0: INTS[6])

This module also supports Toshiba's proprietary data format called "free data format".

Note 1: Compliant with the I²C bus standard (Philips specifications) in fast communication mode except those shown below.

Note 2: This module does not support some of the features in the I²C bus standard.

I²C bus feature	I²C specifications	This IP
Standard mode (up to 100 kHz)	Required	Supported
Fast mode (up to 400 kHz)	Required	Supported
High-speed mode (up to 3.4 Mbps)	Required	Not supported
7 -bit addressing	Required	Supported
10-bit addressing	Required	Not supported
START byte	Required	Not supported
Noise canceler	Required	Supported (digital)
Slope control	Required	Not supported
I/O at power off	Required	Not supported
Schmitt (VIH/VIL)	VDDx0.3 / VDDx0.7	Supported
Output current at VOL = 0.4V, VDD＞2 V	3 mA	Supported

3.14.1.1 I^2C Bus Mode

The I²C bus is connected to devices via the I2C0DA and I2C0CL pins and can communicate with multiple devices.

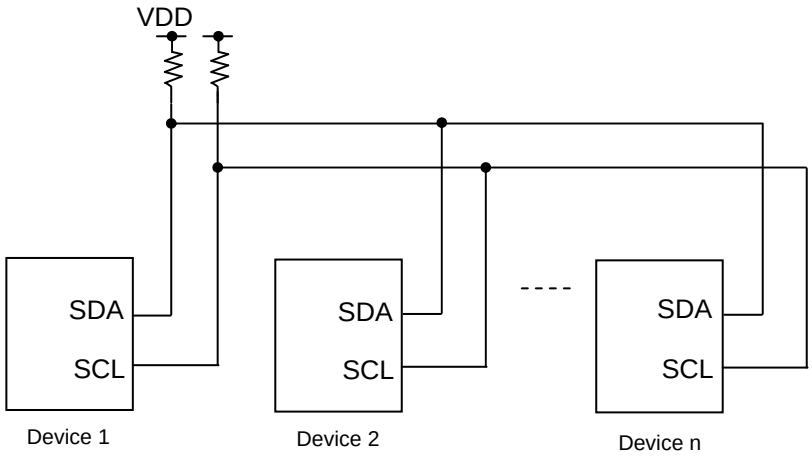


Figure 3.14.1 Device connections

This module operates as a master or slave device on the I²C 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 sends 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 operates as a receiver can output an acknowledge signal after reception of serial data and the device that operates as a transmitter can receive that acknowledge signal, regardless of whether the device is a master or slave. The master device can output a clock for the acknowledge signal.

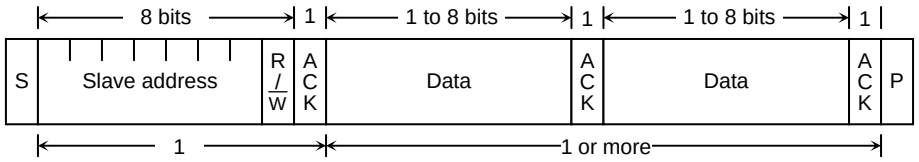
In multimaster mode in which multiple masters exist on the same bus, serial clock synchronization and arbitration lost to maintain consistency of serial data are supported.

3.14.2 Data Formats for I²C Bus Mode

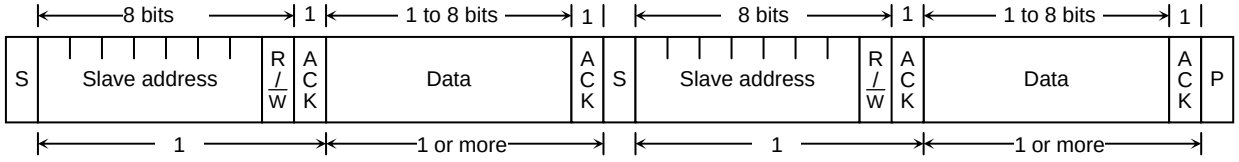
The data formats for I²C bus mode are shown below.

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

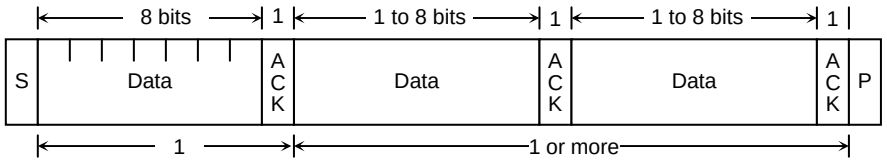
Figure 3.14.2 Data Format for I²C Bus Mode

3.14.2.2 Free Data Format

The free data format is for communication between one master and one slave.

In the free data format, slave addresses and direction bits are processed as data.

(a) Free data format (for transferring data from a master device to a slave device)



S: Start condition
R/W: Direction bit
ACK: Acknowledge bit
P: Stop condition

Figure 3.14.3 Free Data Format for I²C Bus Mode

3.14.3 Block Diagram

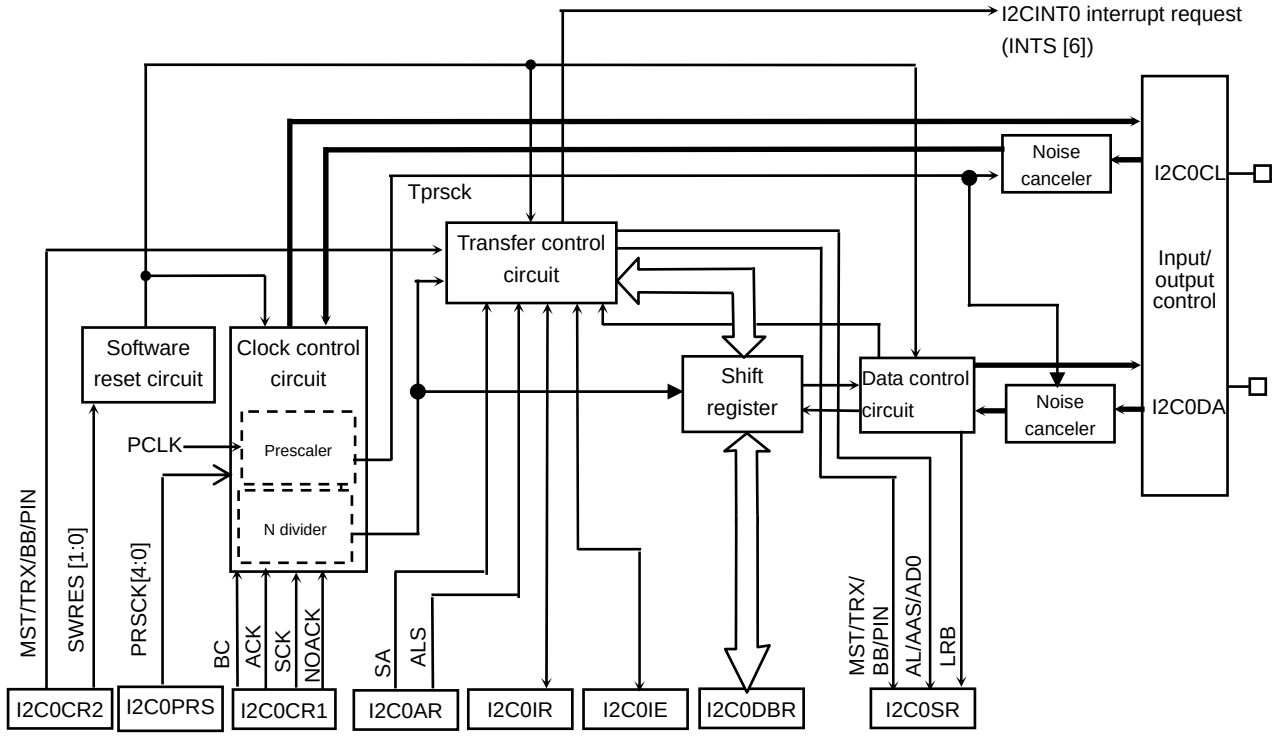


Figure 3.14.4 I^2C Channel 0

3.14.4 Operational Descriptions

3.14.4.1 Data Transfer Procedure in I²C Bus Mode

1. Device Initialization

After ensuring that the I2C0DA and I2C0CL pins are high (bus free), set I2C0CR2<I2CM> to 1 to enable I²C.

Next, set I2C0CR1<ACK> to 1, I2C0CR1<NOACK> to 0 and I2C0CR1<BC> to 0y000. These settings enable acknowledge operation, slave address match detection and general call detection and set the data length to 8 bits. Set t_{HIGH} and t_{LOW} in I2C0CR1<SCK>.

Then, set the slave address in I2C0AR<SA> and set I2C0AR<ALS> to 0 to select the addressing format.

Finally, set I2C0CR2<MST>, I2C0CR2<TRX> and I2C0CR2<BB> to 0, I2C0CR2<PIN> to 1 and I2C0CR2<SWRES[1:0]> to 0y00 to configure the device as a slave receiver.

Note: The initialization of I^2C must be completed within a certain period of time in which no start condition is generated by any device after all the devices connected to the bus have been initialized. If this constraint is not observed, another device may start a transfer before the initiaization of I^2C has been completed and data may not be received properly.

Programming example: Initializing the device

```
CHK _ PORT:  r1          ←  (GPIOCDATA)   ; Check whether the external pins are high.
             CMP            r1, #0xC0
             BNE            CHK _ PORT
             (I2C0CR2)  ←   0x18           ; Enable I2C.
             (I2C0CR1)  ←   0x16           ; Enable acknowledge operation and set I2C0CR1<SCK> = 0y110.
             (I2C0AR)   ←   0xA0           ; Set the slave address to 1010000 and select addressing format.
             (I2C0CR2)  ←   0x18           ; Select slave receiver mode.
```

2. Start Condition and Slave Address Generation

Check that the bus is free (I2C0SR<BB> = 0).

Set I2C0CR1<ACK> to 1 and write the slave address and direction bit to be transmitted to I2C0DBR. Writing 1 to I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB> and I2C0CR2<PIN> causes a start condition, the slave address and direction bit to be sent out on the bus. After a start condition is generated, it takes the t_{HIGH} period for the I2C0CL pin to fall.

Then, an I2CINT0 interrupt request is generated on the falling edge of the 9th clock of I2C0CL and I2C0SR<PIN> is cleared to 0. While I2C0SR<PIN> is 0, I2C0CL is pulled low. Only when the acknowledge signal is returned from the slave device, I2C0SR<TRX> is changed by hardware according to the direction bit upon generation of an I2CINT0 interrupt request.

Note 1: Before writing a slave address to I2C0DBR, make sure that the bus is free by software.

Note 2: After a slave address is written and before a start condition is generated, another master may initiate transfer operation. Therefore, after writing a slave address to I2C0DBR, check a bus free state again by software within 98.0 μs (the shortest transfer time in standard mode according to the I^2C bus standard) or 23.7μs (the shortest transfer time in fast mode according to the I^2C bus standard). A start condition should be generated only after a bus free state is confirmed.

Programming example: Generating a start condition

```
CHK _ BB:   r1          ←   (I2C0SR)      ; Check that the bus is free.
            AND             r1, #0x20
            CMP             r1, #0x00
            BNE             CHK _ BB
            (I2C0DBR)   ←   0xCB          ; Set the slave address to 0x65 and direction bit to 1.
            (I2C0CR2)   ←   0xF8          ; Set I2C0CR2<MST>, <TRX>, <BB>, <PIN> to 1.
```

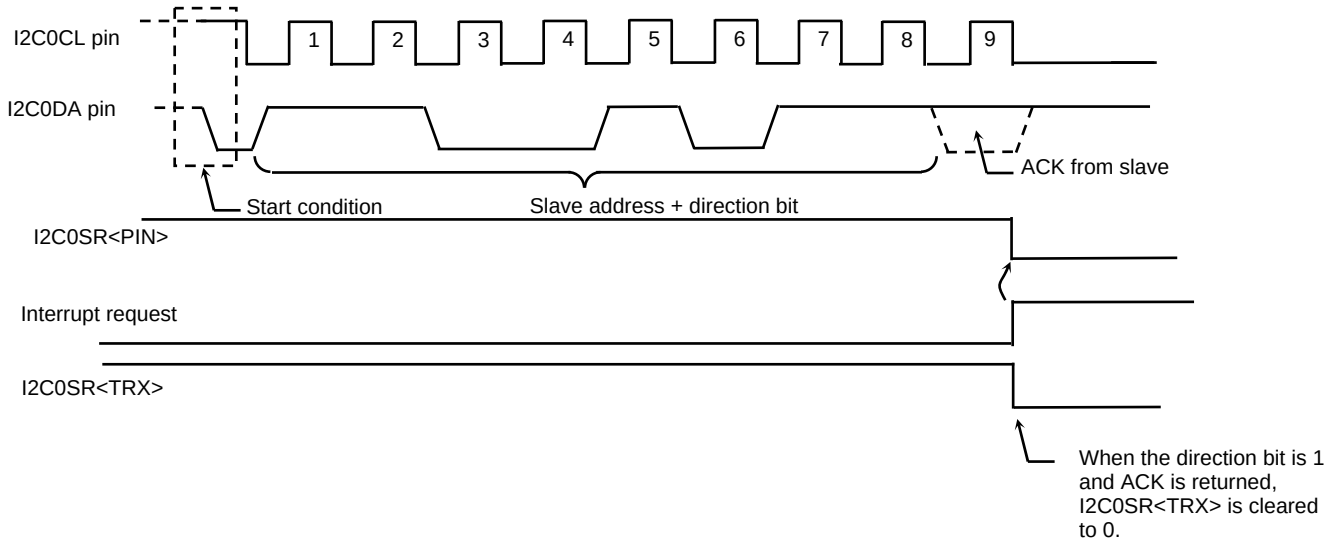


Figure 3.14.5 Start condition and Slave address generation

3. 1-Word Data Transfer

Check I2C0SR<MST> in the interrupt routine after a 1-word data transfer is completed, and determine whether master or slave mode is selected.

(1) When I2C0SR<MST> = 1 (Master mode)

Check I2C0SR<TRX> to determine whether transmitter or receiver mode is selected.

a. When I2C0SR<TRX>=1 (Transmitter mode)

Check the acknowledge status from the receiver with the I2C0SR<LRB> flag. When I2C0SR<LRB> is 0, the receiver is requesting the next data. Write the data to be transmitted to I2C0DBR.

If it is necessary to change the transfer data size, change I2C0CR1<BC>, set I2C0CR1<ACK> to 1, and then write the data to be transmitted to I2C0DBR.

After the transmit data is written, I2C0SR<PIN> is set to 1 and serial clocks are generated to transmit from I2C0CL the data from I2C0DA.

After the transmission is completed, an I2CINT0 interrupt request is generated. I2C0SR<PIN> is cleared to 0 and I2C0CL is pulled low. If more than one word of data needs to be transferred, repeat the procedure by checking I2C0SR<LRB>.

When I2C0SR<LRB> is 1, the receiver is not requesting the next data, so a stop condition should be generated to terminate the data transfer.

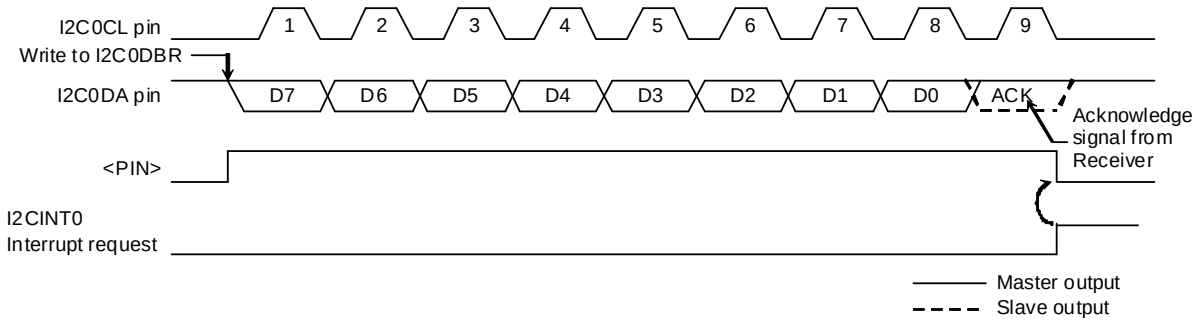


Figure 3.14.6 When I2C0CR1<BC> = 0y000 and I2C0CR1<ACK> = 1

b. When I2C0SR<TRX> = 0 (Receiver mode)

Writing dummy data (0x00) to I2C0DBR or setting I2C0CR2<PIN> to 1 causes clocks for 1-word transfer and acknowledge to be output.

After an I2CINT0 interrupt request is generated to indicate the end of receive operation, read the received data from I2C0DBR.

If it is necessary to change the receive data size, change I2C0CR1<BC>, set I2C0CR1<ACK> to 1 and then write dummy data (0x00) to I2C0DBR or set I2C0CR2<PIN> to 1.

(The data that is read immediately after slave address transmission is undefined.)

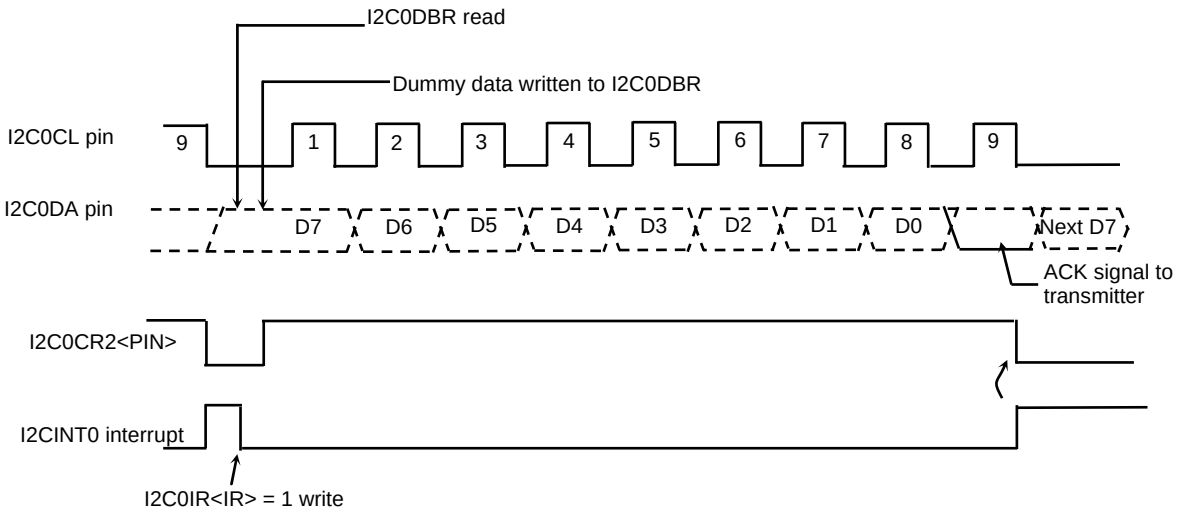


Figure 3.14.7 When I2C0CR1<BC> = 0y000 and I2C0CR1<ACK> = 1

c. When I2C0SR<TRX> = 0 (when receiving the last word)

The last word of the transfer is determined by pseudo communication without acknowledge. The flow of this operation is explained below.

To make the transmitter terminate transmission, perform the following operations before the last data bit is received:

1. Read the received data from I2C0DBR.
2. Clear I2C0CR1<ACK> to 0 and set I2C0CR1<BC> to 0y000.
3. Write dummy data (0x00) to I2C0DBR to set I2C0CR2<PIN> to 1.

After I2C0CR2<PIN> is set to 1, 1-word transfer with no acknowledge operation is performed. After 1-word transfer is completed, perform the following operations:

1. Read the received data from I2C0DBR.
2. Clear I2C0CR1<ACK> to 0 and set I2C0CR1<BC> to 0y001 (negative acknowledge).
3. Write dummy data (0x00) to I2C0DBR to set I2C0CR2<PIN> to 1.

When I2C0CR2<PIN> is set to 1, 1-bit transfer is performed. Since the master is acting as a receiver, the SDA line on the bus remains high. The transmitter receives this high-level signal as the negative acknowledge signal. The receiver can thus indicate the transmitter that the data transmission is completed. After 1-bit data is received and an interrupt request is generated, generate a stop condition to terminate the data transfer.

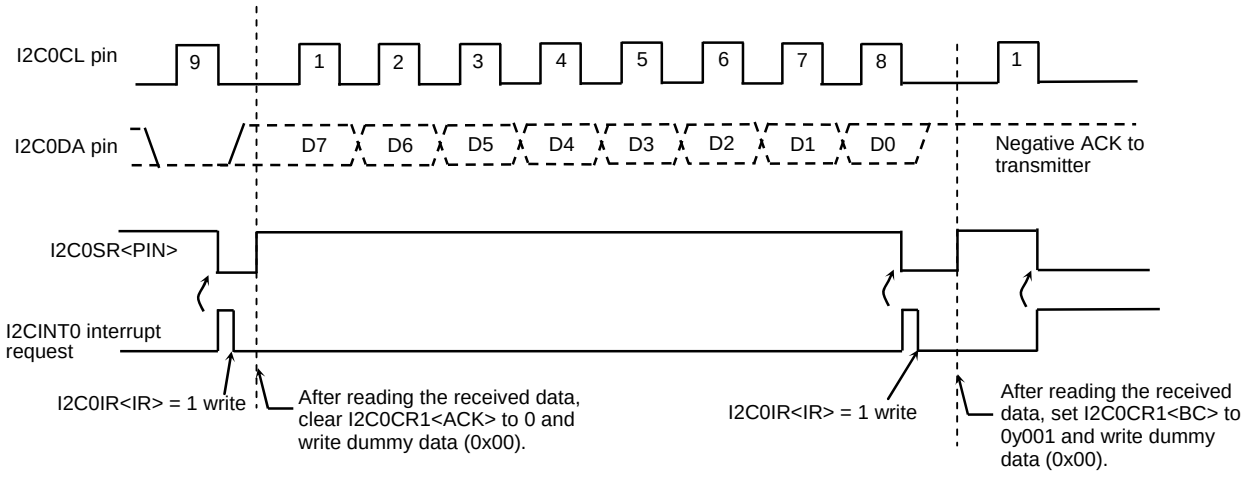


Figure 3.14.8 Terminating data transmission in master receiver mode

(2) When I2C0SR<MST> = 0 (Slave mode)

The following explains normal slave mode operations and the operations to be performed when I²C changes to slave mode after losing arbitration on the bus.

In slave mode, an I2CINT0 interrupt request is generated by the following conditions:

- When I2C0CR1<NOACK> is 0, after the acknowledge signal is output to indicate that the received slave address has matched the slave address set in I2C0AR<SA>
- When I2C0CR1<NOACK> is 0, after the acknowledge signal is output to indicate that a general call has been received
- When data transfer is completed after a matched slave address or general call is received.

I²C changes to slave mode if it loses arbitration while operating in master mode. After completion of the word transfer in which arbitration lost occurred, an I2CINT0 interrupt request is generated. Table 3.14.1 shows the I2CINT0 interrupt request and I2C0SR<PIN> operations when arbitration lost occurs.

Table 3.14.1 I2CINT0 interrupt request and I2C0SR<PIN> operations after arbitration lost

	When arbitration lost occurs during transmission of slave address as master	When arbitration lost occurs during transmission of data as master transmitter
I2CINT0 interrupt request	An I2CINT0 interrupt request is generated after the current word of data has been transferred.	
I2C0SR<PIN>	I2C0SR<PIN> is cleared to 0.	

When an I2CINT0 interrupt request occurs, I2C0SR<PIN> is reset to 0, and I2C0CL is pulled low. Either writing data to I2C0DBR or setting I2C0CR2<PIN> to 1 releases I2C0CL after the t_{LOW} period.

Check I2C0SR<AL>, I2C0SR<TRX>, I2C0SR<AAS> and I2C0SR<AD0>, and implement required operations, as shown in Table 3.14.2.

Table 3.14.2 Operations in slave mode

I2C0SR <TRX>	I2C0SR <AL>	I2C0SR <AAS>	I2C0SR <AD0>	Condition	Operation
1	1	1	0	The device loses arbitration during slave address transmission, and receives a slave address with direction bit set to 1 from another master.	Set the number of bits in 1 word to I2C0CR1<BC> and write the data to be transmitted to I2C0DBR.
	0	1	0	In slave receiver mode, the device receives a slave address with direction bit set to 1 from the master.	
		0	0	In slave transmitter mode, the device completes the transmission of 1-word data.	Check I2C0SR<LRB>. If it is set to 1, the receiver is not requesting the next data, so set I2C0CR2<PIN> to 1. Then, clear I2C0CR2<TRX> to 0 to release the bus. If I2C0SR<LRB> = 0, the receiver is requesting the next data, so set the number of bits in 1 word in I2C0CR1<BC> and write the data to be transmitted to I2C0DBR.
0	1	1	1/0	The device loses arbitration during slave address transmission, and receives a slave address with direction bit set to 0 from another master or receives a general call.	Write dummy data (0x00) to I2C0DBR to set I2C0SR<PIN> to 1, or write 1 to I2C0CR2<PIN>.
		0	0	The device loses arbitration when transmitting a slave address or data, and completes transferring the current word of data.	The device is set as a slave. Clear I2C0SR<AL> to 0 and write dummy data (0x00) to I2C0DBR to set I2C0SR<PIN> to 1.
	0	1	1/0	In slave receiver mode, the device receives a slave address with direction bit set to 0 from another master or receives a general call.	Write dummy data (0x00) to I2C0DBR to set I2C0SR<PIN> to 1, or write 1 to I2C0CR2<PIN>.
		0	1/0	In slave receiver mode, the device completes the receipt of 1-word data.	Set the number of bits in 1 word to I2C0CR1<BC>, read the received data from I2C0DBR and write dummy data (0x00).

Note: In slave mode, if I2C0AR<SA> is set to 0x00 and a START byte (0x01) of the I^2C bus standard is received, a slave address match is detected and I2C0SR<TRX> is set to 1. Do not set I2C0AR<SA> to 0x00.

4. Stop Condition Generation

When I2C0SR<BB> is 1, writing 1 to I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<PIN> and 0 to I2C0CR2<BB> initiates the sequence for generating a stop condition on the bus. Do not change the contents of I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB> and I2C0CR2<PIN> until a stop condition is generated on the bus.

If the I2C0CL line is pulled low by another device when the sequence for generating a stop condition is started, a stop condition will be generated after the I2C0CL line is released.

It takes the t_{HIGH} period for a stop condition to be generated after the I2C0CL line is released.

Programming example: Generating a stop condition

```
                I2C0CR2 ←   0xD8        ; Set I2C0CR2<MST>,<TRX>,<PIN> to 1 and
                                          I2C0CR2<BB> to 0.
CHK _ BB:       r1      ←   (I2C0SR)    ; Check that the bus is free.
                AND         r1, #0x20
                CMP         r1, #0x00
                BNE         CHK _ BB
```

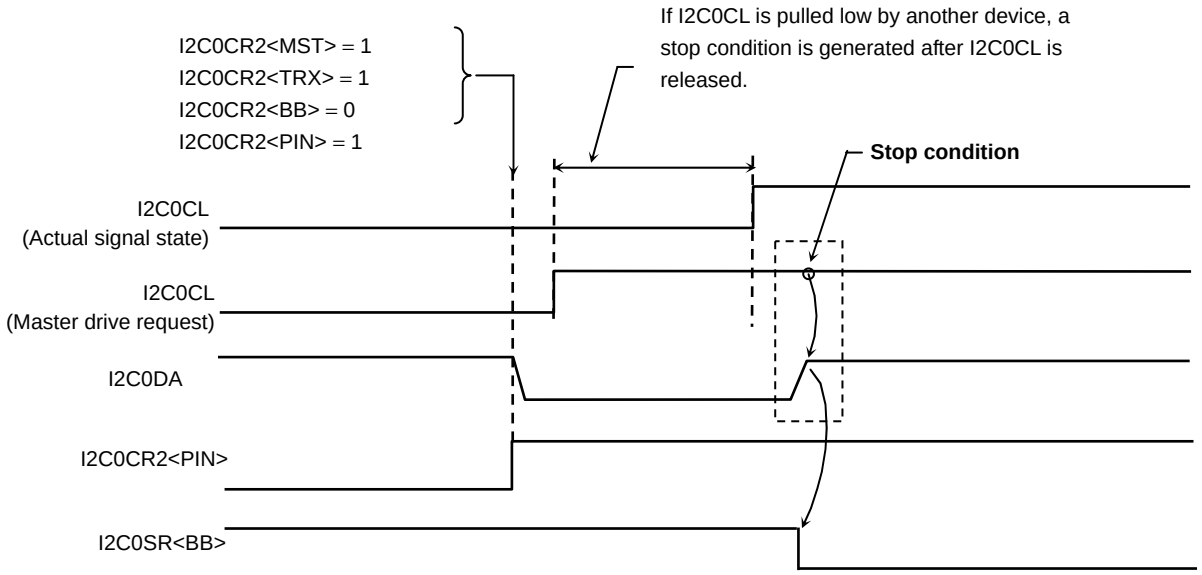


Figure 3.14.9 Stop condition generation

5. Restart Procedure

Restart is used to change the direction of data transfer without the master device terminating data transfer to the slave device. The restart procedure is explained below.

First, write 0 to I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB> and 1 to I2C0CR2<PIN>. The I2C0DA line remains high and the I2C0CL line is released.

Since this is not a stop condition, the bus remains busy for other devices.

Next, check I2C0SR<BB> until it is cleared to 0 to make sure that the I2C0CL line is released.

Then, check I2C0SR<LRB> until it becomes 1 to make sure that the I2C0CL line is not pulled low by another device.

After making sure that the bus is free by these steps, generate a start condition as explained earlier in "2. Start Condition and Slave Address Generation".

In order to satisfy the setup time requirement for restart, it is necessary to insert, by software, a wait period of 4.7 μs or longer in the case of standard mode and 0.6 μs or longer in the case of fast mode.

Note: When the master device is operating as a receiver, it is necessary to terminate the data transfer from the slave transmitter before the restart procedure can be started. To do so, the master device makes the slave device receive the negative acknowledge signal (high). Therefore, I2C0SR<LRB> is set to 1 before the restart procedure is started. The SCL line level cannot be determined by checking I2C0SR<LRB> = 1 in the restart procedure. The state of the I2C0CL line shold be checked by reading the port.

Programming example: Generating a restart condition

```
                (I2C0CR2) ←  0x18          ; Set I2C0CR2<MST>, <TRX>,<BB> to 0 and
                                           I2C0CR2<PIN> to 1.
CHK _ BB:       r1        ←  (I2C0SR)      ; Wait until I2C0SR<BB> is cleared to 0.
                AND          r1, #0x20
                CMP          r1, #0x00
                BNE          CHK _ BB
CHK _ LRB:      r1        ←  (I2C0SR)      ; Wait until I2C0SR<LRB> becomes 1.
                AND          r1, #0x01
                CMP          r1, #0x01
                BNE          CHK _ LRB

                             .              ; Wait by software
                             .
                (I2C0CR2) ←  0xF8           ; Set I2C0CR2<MST>, <TRX>, <BB>, <PIN> to 1.
```

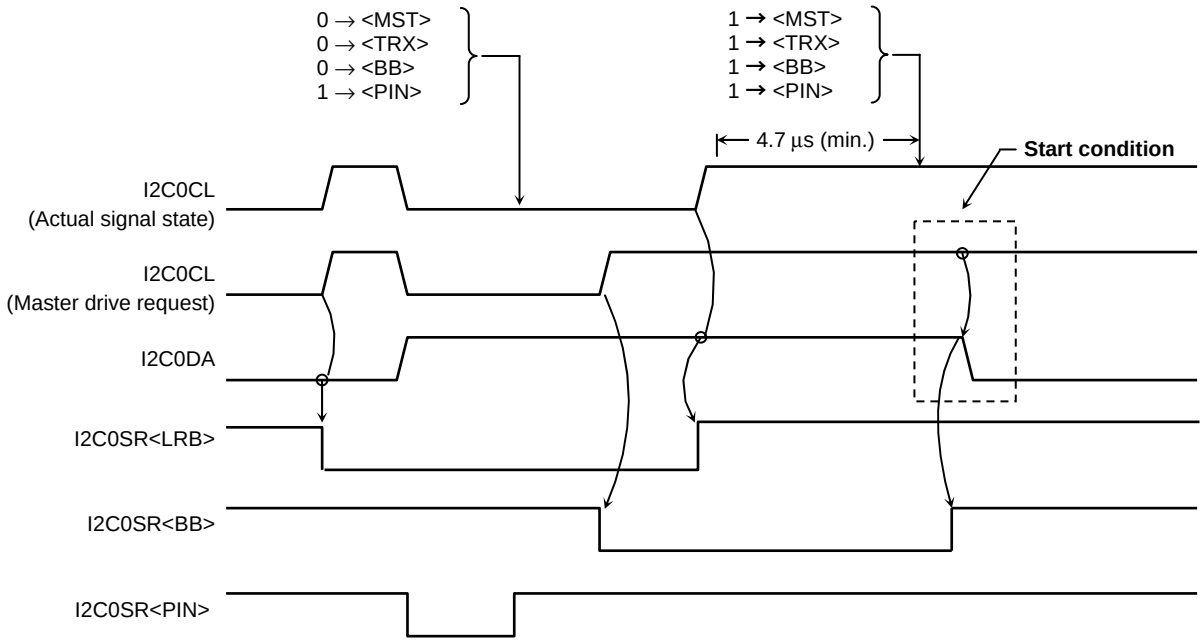


Figure 3.14.10 Restart timing chart

Note: When <MST> = 0, do not write 0 to <MST>. (Restart cannot be performed.)

3.14.5 Register Descriptions

The following lists the SFRs.

Base address = 0xF007_0000

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

1. I2C0CR1 (I^2C0 Control Register 1)

Address = (0xF007_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:5]	BC[2:0]	R/W	0y000	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	Acknowledge clock generation and recognition 0y0: Disable 0y1: Enable
[3]	NOACK	R/W	0y0	Slave address match detection and general call detection 0y0: Enable 0y1: Disable
[2:0]	SCK[2:0]	R/W	0y000	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

[Description]

a. <BC[2:0]>

These bits 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 specifies whether to disable or enable acknowledge clock generation and recognition.

0y0: Disable

0y1: Enable

c. <NOACK>

This bit specifies whether to enable or disable the slave address match detection and general call detection when this module is a slave.

0y0: Enable

0y1: Disable

When I2C0AR<ALS> = 1, this bit has no meaning.

When <NOACK> = 0, the slave address match detection and general call detection are enabled. When a slave address match or general call is detected, the slave pulls the SDA line low during the 9th (acknowledge) clock output from the master to return an acknowledge signal.

Setting <NOACK> = 1 disables the slave address match detection and general call detection. When a slave address match or general call is detected, the slave releases (holds high) the SDA line during the 9th (acknowledge) clock output from the master to return no acknowledge signal.

d. <SCK[2:0]>

These bits are used to set the rate of serial clock to be output from the master.

The prescaler clock divided according to I2C0PRS<PRSCK[4:0]> is used as the reference clock for serial clock generation. The prescaler clock is further divided according to I2C0CR1<SCK[2:0]> to generate the serial clock. The default setting of the prescaler clock is “divide by 1” (= f_{PCLK}).

Note: Refer to Section 3.14.5 6. I2C0PRS (I2C0 Prescaler Clock Set Register)” and Section 3.14.6.3 “Serial Clock”.

Writes to this register must be done before a start condition is generated or after a stop condition is generated or between the instant when an address or data transfer interrupt occurs and the instant when the internal interrupt is released. Do not write to this register during address or data transfer.

2. I2C0DBR (I^2C0 Data Buffer Register)

Address = (0xF007_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7:0]	DB[7:0]	RO	0x00	Read: Receive data is read (Note)

Address = (0xF007_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	DB[7:0]	WO	0x00	Write: Transmit data is written (Note)

Note: This register is initialized only after a hardware reset. It is not initialized by a software reset. (The most recent data is retained.)

[Description]

a. <DB[7:0]>

These bits are used to store data for serial transfer.

When this module is a transmitter, the data to be transmitted is written into DB[7:0] aligned on the left side.

When this module is a receiver, the received data is stored into DB[7:0] aligned on the right side.

When the master needs to transmit a slave address, the transfer target address is written to I2C0DBR<DB[7:1]> and the transfer direction is specified in I2C0DBR<DB[0]> as follows:

0y0: Master (transmission) → Slave/reception

0y1: Master (reception) ← Slave/transmission

When all the bits in the I2C0DBR register are written as 0, a general call can be sent out on the bus.

In both transmission and reception modes, a write to the I2C0DBR register releases the internal interrupt after the current transfer and initiates the next transfer.

Although I2C0DBR is provided as a transmit/receive buffer, it should be used as a dedicated transmit buffer in transmit mode and as a dedicated receive buffer in receive mode. This register should be accessed on a transfer-by-transfer basis.

Note: In receive mode, if data is written to I2C0DBR before the received data is read out, the received data will be corrupted.

3. I2C0AR (I^2C0 (Slave) Address Register)

Address = (0xF007_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:1]	SA[6:0]	R/W	0y0000000	Set the slave address.
[0]	ALS	R/W	0y0	Address recognition enable/disable 0y0: Enable (I^2C bus mode) 0y1: Disable (Free data format)

[Description]

a. <SA[6:0]>

These bits are used to set the slave device address (7 bits) when this module is a slave.

When slave address recognition is enabled in I2C0AR<ALS>, the data transfer operation to be performed is determined by the 7-bit address (plus one direction bit) that the master sends immediately after a start condition.

b. <ALS>

This bit is used to enable or disable slave address recognition.

0y0: Enable (I^2C bus mode)

0y1: Disable (Free data format)

When this module is a slave, this bit specifies whether or not to recognize the 8-bit data that the master sends immediately after a start condition as a 7-bit address plus one direction bit.

When <ALS> = 0, I^2 C bus mode is selected. When <ALS> = 1, transfer operation is performed based on the free data format.

When <ALS> = 0, the device compares the 7-bit address sent from the master against the slave address set in I2C0AR<SA[6:0]>. If the 7-bit address matches the slave address, the device uses the direction bit to determine whether to act as a transmitter or receiver. At this time, if I2C0CR1<NOACK> = 0, the device pulls the SDA line low during the 9th (acknowledge) clock output from the master.

Thereafter, the device continues to perform transmit or receive operation as a slave until a stop condition or a start condition by the restart procedure appears on the bus.

If the 7-bit address does not match the slave address, the device continues to leave the SDA line and SCL line high and does not participate in transfer operation until a stop condition or a start condition by the restart procedure appears on the bus.

If the 7-bit address plus one direction bit sent from the master are all 0s (indicating a general call) and I2C0CR1<NOACK> = 0, the device returns an acknowledge signal and acts as a slave receiver regardless of the slave address set in I2C0AR<SA[6:0]>.

When I2C0CR1<NOACK> = 1, the device does not return any acknowledge signal nor operate as a slave device even if the 7-bit address matches the slave address or a general call is detected.

When <ALS> = 1, the device receives the 7-bit address plus one direction bit sent from the master as data and pulls the SDA line low during the 9th (acknowledge) clock output from the master. Thereafter the device continues to perform receive operation as a slave until a stop condition or a start condition by the restart procedure appears on the bus (free format operation). In this case, the I2C0CR1<NOACK> value has no effect.

Writes to this register must be done before a start condition is generated or after a stop condition is generated. Writes cannot be performed during transfer.

4. I2C0CR2 (I^2C0 Control Register 2) (Write Only)

Address = (0xF007_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	MST	WO	0y0	Selects master or slave mode. 0y0: Slave 0y1: Master
[6]	TRX	WO	0y0	Selects transmit or receive operation. 0y0: Receiver 0y1: Transmitter
[5]	BB	WO	0y0	Selects whether to generate a start or stop condition. 0y0: Generate a stop condition. 0y1: Generate a start condition.
[4]	PIN	WO	0y1	Service request clear 0y0: No effect 0y1: Clear service request
[3]	I2CM	WO	0y0	I^2C operation control 0y0: Disable 0y1: Enable
[2]	–	–	Undefined	Read as undefined. Write as zero.
[1:0]	SWRES[1:0]	WO	0y00	Software reset A software reset is generated by writing 0y10 and then 0y01 to these bits.

[Description]

a. <MST>

This bit selects master or slave mode.

0y0: Slave

0y1: Master

Note: Refer to Section 3.14.6.3 “Serial CLock”.

b. <TRX>

This bit selects transmission or reception mode.

0y0: Reception

0y1: Transmission

Note: Refer to Section 3.14.6.3 “Serial Clock”.

c. <BB>

This bit is used to generate a start or stop condition.

0y0: Generate a stop condition.

0y1: Generate a start condition.

Note: Refer to Section 3.14.6.3 “Serial Clock”.

d. <PIN>

This bit is used to clear a service request for I²C communication.

0y0: Invalid

0y1: Clear service request

Note: Refer to Section 3.14.6.3 “Serial Clock”.

e. <I2CM>

This bit enables or disables I²C operation.

0y0: Disable

0y1: Enable

The <I2CM> bit cannot be cleared to 0 to disable I²C operation while transfer operation is being performed. Before clearing this bit, make sure that transfer operation is completely stopped by reading the status register.

f. <SWRES[1:0]>

Writing 0y10 and then 0y01 to these bits generates a software reset (reset width = one f_{PCLK} clock pulse).

If a software reset occurs, the SCL and SDA lines are forcefully released (driven high) to abort any ongoing transfer operation. All the settings except I2C0CR2<I2CM> are initialized. (I2C0DBR is not initialized.)

When generating a software reset, be sure to write 0 to I2C0CR2[7:4].

5. I2C0SR (I^2C0 Status Register) (Read Only)

Address = (0xF007_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	MST	RO	0y0	Master/slave selection state monitor 0y0: Slave 0y1: Master
[6]	TRX	RO	0y0	Transmit/receive selection state monitor 0y0: Receiver 0y1: Transmitter
[5]	BB	RO	0y0	Bus state monitor 0y0: The bus is free. 0y1: The bus is busy.
[4]	PIN	RO	0y1	Service request state and SCL line state monitor 0y0: Service request present, SCL line = low 0y1: No service request, SCL line = free
[3]	AL	RO	0y0	Arbitration lost detection monitor 0y0: Invalid 0y1: Detected
[2]	AAS	RO	0y0	Slave address match detection monitor 0y0: Invalid 0y1: Detected
[1]	AD0	RO	0y0	General call detection monitor 0y0: Invalid 0y1: Detected
[0]	LRB	RO	0y0	Last received bit monitor 0y0: The bit received last is 0. 0y1: The bit received last is 1.

[Description]

a. <MST>

This bit monitors whether master or slave mode is selected.

0y0: Slave

0y1: Master

b. <TRX>

This bit monitors whether transmission or reception mode is selected.

0y0: Reception

0y1: Transmission

c. <BB>

This bit monitors the bus status.

0y0: The bus is free.

0y1: The bus is busy.

This bit is set to 1 after a start condition is detected on the bus. It is cleared to 0 on detection of a stop condition. When the device is operating as a slave, this bit is set to 1 to monitor the generation of a stop condition even if the device is not selected by the master and is not involved in transfer operation.

While this bit is set to 1, the start condition cannot be generated.

d. <PIN>

This bit monitors the service request state and SCL state.

0y0: Service request present, SCL line = low

0y1: No service request, SCL line = free

e. <AL>

This bit monitors the detection of arbitration lost.

0y0: Invalid

0y1: Detected

f. <AAS>

This bit monitors the detection of a slave address match.

0y0: Invalid

0y1: Detected

When the device is operating as a slave, this bit is set to 1 if the slave address sent from the master matches the slave address set in I2C0AR<SA[6:0]>. This bit is then cleared to 0 after the internal interrupt is released and remains 0 until a stop condition or a start condition by the restart procedure appears on the bus and it is again set to 1 by a slave address match in address transfer after that start condition.

g. <AD0>

This bit monitors the detection of a general call.

0y0: Invalid

0y1: Detected

This bit is set to 1 on detection of a general call (the SDA line is held low during address transfer after a start condition) and remains set until a stop condition or a start condition by the restart procedure appears on the bus. I2C0SR<AAS> is also set to 1 on detection of a general call. However, this bit is cleared to 0 at the next data transfer as described earlier.

h. <LRB>

This bit monitors the last received bit.

0y0: The bit received last is 0.

0y1: The bit received last is 1.

When acknowledge operation is enabled, this bit can be used to check whether or not the receiver has output an acknowledge signal (low) by reading the bit in the interrupt routine after the transfer. This monitor is effective regardless of whether the device is set as a transmitter or receiver.

Note: Rerer to Section 3.14.6.15 Register Values after a Software Reset ".

6. I2C0PRS (I^2C0 Prescaler Clock Set Register)

Address = (0xF007_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined. Write as zero.
[4:0]	PRSCK[4:0]	R/W	0y00001	Prescaler clock frequency for generating the serial clock 0y00000: p = Divide by 32 0y00001: p = Divide by 1 : 0y11111: p = Divide by 31

[Description]

a. <PRSCK[4:0]>

These bits are used to select the prescaler clock frequency for generating the serial clock.

0y00000: p = Divide by 32

0y00001: p = Divide by 1

:

0y11111: p = Divide by 31

Note: Refer to Section 3.14.5 “1. I2C0CR1 (I2C0 Control Register 1)” and Section 3.14.6.3 “Serial Clock”.

7. I2C0IE (I^2C0 Interrupt Enable Register)

Address = (0xF007_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	IE	R/W	0y0	I^2C interrupts 0y0: Disable 0y1: Enable

[Description]

a. <IE>

This bit is used to enable or disable I^2C interrupts.

0y0: Disable

0y1: Enable

8. I2C0IR (I^2C0 Interrupt Register)

Address = (0xF007_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	IS/IC	R/W	0y0	(Read) Indicates I^2C interrupt status (before being disabled). 0y0: No interrupt 0y1: Interrupt generated (Write) Clears the I^2C interrupt. 0y0: Invalid 0y1: Clear

[Description]

a. <IS/IC>

(Read)

This bit indicates the I^2C interrupt status prior to masking by I2C0IE<IE>.

0y0: No interrupt

0y1: Interrupt generated

(Write)

This bit is used to clear the I^2C interrupt.

0y0: Invalid

0y1: Clear

Writing 1 to this bit clears the I^2C interrupt output (I2CINT0).

Writing 0 is invalid.

3.14.6 Functions

3.14.6.1 Slave Address Match Detection and General Call Detection

For a slave device, the following setting is made for slave address match detection and general call detection.

I2C0CR1<NOACK> enables or disables the slave address match detection and general call detection in slave mode.

Clearing I2C0CR1<NOACK> to 0 enables the slave address match detection and general call detection.

Setting I2C0CR1<NOACK> to 1 disables the slave address match detection and general call detection. The slave device ignores slave addresses and general calls sent from the master and returns no acknowledgement. I2CINT0 interrupt requests are not generated.

In master mode, I2C0CR1<NOACK> is ignored and has no effect on operation.

Note: If I2C0CR1<NOACK> is cleared to 0 during data transfer in slave mode, it remains 1 and an acknowledge signal is returned for the transferred data.

3.14.6.2 Number of Clocks for Data Transfer and Acknowledge Operation

(1) Number of clocks for data transfer

The number of clocks for data transfer is set through I2C0CR1<BC> and I2C0CR1<ACK>.

Setting I2C0CR1<ACK> to 1 enables acknowledge operation.

The master device generates clocks for the number of data bits to be transferred, and then generates an acknowledge clock and an I2CINT0 interrupt request.

The slave device counts clocks for the number of data bits, and then counts an acknowledge clock and generates an I2CINT0 interrupt request.

Clearing I2C0CR1<ACK> to 0 disables acknowledge operation.

The master device generates clocks for the number of data bits to be transferred, and then generates an I2CINT0 interrupt request.

The slave device counts clocks for the number of data bits, and then generates an I2CINT0 interrupt request.

When acknowledge operation is enabled in receiver mode, the device pulls I2C0DA low during the acknowledge clock period from the master to request the transfer of the next word. Conversely, by holding I2C0DA high during the acknowledge clock period from the master, the receiver device can indicate that it is not requesting the next word.

During address transmission (before a start condition is generated), both the master and slave must be configured for 8-bit transfer with acknowledge enabled.

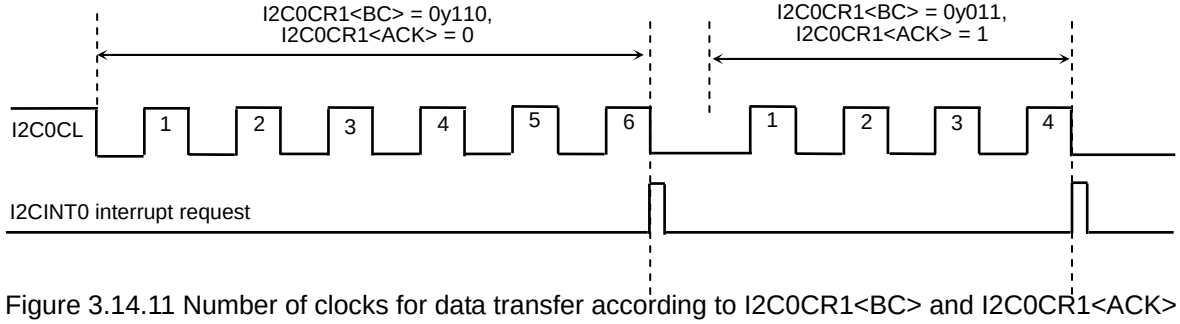


Figure 3.14.11 Number of clocks for data transfer according to I2C0CR1<BC> and I2C0CR1<ACK>

Table 3.14.3 shows the relationship between the number of clocks for data transfer and the I2C0CR1<BC> and I2C0CR1<ACK> settings.

Table 3.14.3 Number of clocks for data transfer

BC[2:0]	Acknowledge operation (I2C0CR1<ACK>)			
	0y0: Disabled		0y1: Enabled	
	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

I2C0CR1<BC> is cleared to 0y000 by a start condition.

This means that the slave address and direction bit are always transferred as 8-bit data. At other times, <BC> retains the set value.

Note: A slave address must be transmitted/received with I2C0CR1<ACK> set to 1. If I2C0CR1<ACK> is cleared, the slave address match detection and direction bit detection cannot be performed properly.

(2) Acknowledge output

When acknowledge operation is enabled, I2C0DA changes during the acknowledge clock period as explained below.

- Master mode

The master transmitter releases I2C0DA during the acknowledge clock period to receive the acknowledge signal from the slave receiver.

The master receiver pulls I2C0DA low during the acknowledge clock period and to generate the acknowledge signal.

- Slave mode

When the received slave address matches the slave address set in I2C0AR<SA> or when a general call is received, the slave pulls I2C0DA low during the acknowledge clock period to generate the acknowledge signal.

In data transfer after a slave address match or a general call, the slave transmitter releases I2C0DA during the acknowledge clock period to receive the acknowledge signal from the master receiver.

The slave receiver pulls I2C0DA low during the acknowledge clock period to generate the acknowledge signal.

Table 3.14.4 shows the I2C0CL and I2C0DA states when acknowledge operation is enabled.

Note: When acknowledge operation is disabled, no acknowledge clock is generated or counted and no acknowledge signal is output.

Table 3.14.4 I2C0CL and I2C0DA states when acknowledge is enabled

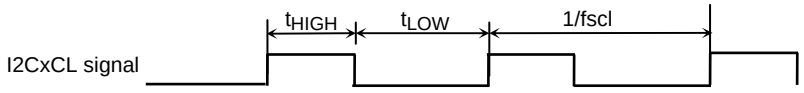
Mode	Pin	Condition	Transmitter	Receiver
Master	I2C0CL	–	Adds the acknowledge clock pulse.	Adds the acknowledge clock pulse.
	I2C0DA	–	Releases the pin to receive the acknowledge signal.	Pulls the pin low as the acknowledge signal.
Slave	I2C0CL	–	Counts the acknowledge clock pulse.	Counts the acknowledge clock pulse.
	I2C0DA	When a slave address match is detected or a general call is received.	–	Pulls the pin low as the acknowledge signal.
		During transfer after a slave address match is detected or a general call is received	Releases the pin to receive the acknowledge signal.	Pulls the pin low as the acknowledge signal.

3.14.6.3 Serial Clock

(1) Clock source

I2C0CR1<SCK> is used to set the high and low periods of the serial clock to be output in master mode.

SCK	t_{HIGH} (i / Tprsck)	t_{LOW}(j / Tprsck)
	i	j
0y000	8	12
0y001	10	14
0y010	14	18
0y011	22	26
0y100	38	42
0y101	70	74
0y110	134	138
0y111	262	266



t_{HIGH} = (i/Tprsck)
t_{LOW} = (j/Tprsck)
fscl = 1/(t_{HIGH} + t_{LOW})

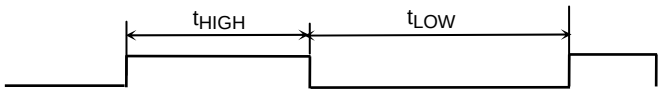
Figure 3.14.12 I2CxCL output

Note: The t_{HIGH} period may differ from the specified value if the rising edge becomes blunt depending on the combination of bus load capacitance and pull-up resistor. If the clock synchronization function for synchronizing clocks from multiple clocks is used, the actual clock period may differ from the specified setting.

In master mode, the hold time when a start condition is generated and the setup time when a stop condition is generated are defined as t_{HIGH} [s].

When I2C0CR2<PIN> is set to 1 in slave mode, the time to the release of I2C0CL is defined as t_{LOW} [s].

In both master and slave modes, the high level period must be 4/Tprsck [s] or longer and the low level period must be 5/Tprsck [s] or longer for externally input serial clocks, regardless of the I2C0CR1<SCK> setting.



t_{HIGH} > = (4/Tprsck)
t_{LOW} > = (5/Tprsck)

Figure 3.14.13 SCLK input

The serial clock rate to be output from the master is set through I2C0CR1<SCK[2:0]> and I2C0PRS<PRSCK[4:0]>. The prescaler clock which is divided according to I2C0PRS<PRSCK[4:0]> is used as the reference clock for generating the serial clock. 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 (= PCLK)".

<Serial transfer rate>

The serial clock rate (Fscl) is determined by prescaler setting value "p" (I2C0PRS <PRSCK[4:0]>, p = 1-32) and serial clock setting value "n" (2C0CR1<SCK[2:0]>, n = 0-7) based on the operating frequency (PCLK) as follows:

$$\text{Serial clock rate Fscl (kHz)} = \frac{\text{PCLK(MHz)}}{p \times (2^{n+2} + 16)} \times 1000$$

Note: The allowed range of prescaler setting value "p" (I2C0PRS<PRSCK[4:0]>) varies depending on the operating frequency (PCLK) and must satisfy the following condition:

50 ns < Prescaler clock width Tprsck (ns) ≤ 150ns

Note: Setting the prescaler clock width out of this range is prohibited in both master and slave modes.

The serial clock rate may not be constant due to the clock synchronization function.

SCK[2:0] = (n)			PRSCK [4:0] = (p)		
			0y00001 (divide by 1)	0y01101 (divide by 13)	0y00000 (divide by 32)
			(Ratio to PCLK)		
0	0	0	20	260	640
0	0	1	24	312	768
0	1	0	32	416	1024
0	1	1	48	624	1536
1	0	0	80	1040	2560
1	0	1	144	1872	4608
1	1	0	272	3536	8704
1	1	1	528	6864	16896

Writes to these bits must be done before a start condition is generated or after a stop condition is generated. Writes during transfer will cause unexpected operation.

<Prescaler clock width (= noise cancellation width)>

The prescaler clock width (Tprsck) (= noise cancellation width) is determined by prescaler setting value "p" (I2C0PRS<PRSCK[4:0]>, p = 1-32) based on the operating frequency (PCLK) as follows:

$$\text{Prescaler clock width Tprsck (ns) (= Noise cancellation width)} = \frac{1}{\text{PCLK (MHz)}} \times 1000 \times p$$

(2) Clock synchronization

The I²C bus is driven by the wired AND method, and a master device that first pulls down the clock line low invalidates the clock outputs from other masters on the bus. Masters who are keeping the clock line high need to detect this situation and act as required.

I²C has a clock synchronization function to ensure proper transfer operation even when multiple masters exist on the bus.

The clock synchronization procedure is explained below using an example where two masters simultaneously exist on the bus.

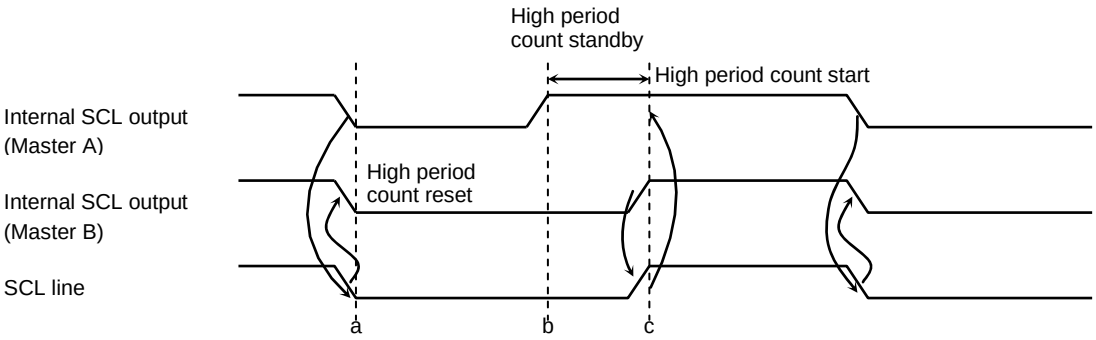


Figure 3.14.14 Example of clock synchronization

As Master A pulls I2C0CL low at point “a”, the SCL line of the bus becomes low. After detecting this situation, Master B resets the high level period count and pulls I2C0CL low.

Master A finishes counting the low level period at point “b” and sets I2C0CL to high. Since Master B is still holding the SCL line low, Master A does not start counting the high level period. Master A starts counting the high level period after Master B sets I2C0CL to high at point “c” and the SCL line of the bus becomes high.

Then, after counting the high level period, Master A pulls I2C0CL low and the SCL line of the bus becomes low.

The clock operation on the bus is determined by the master device with the shortest high level period and the master device with the longest low level period among master devices connected to the bus.

3.14.6.4 Master/Slave Selection

When I2C0CR2<MST> is set to 1, I²C is configured as a master device.

When I2C0CR2<MST> is cleared to 0, it is configured as a slave device.

I2C0SR<MST> is cleared to 0 by hardware when a stop condition or arbitration lost is detected on the bus.

3.14.6.5 Transmitter/Receiver Selection

When I2C0CR2<TRX> is set to 1, I²C is configured as a transmitter. When I2C0CR2<TRX> is cleared to 0, it is configured as a receiver.

In I²C data transfer in slave mode, I2C0SR<TRX> is set to 1 by hardware if the direction bit (R/W) sent from the master is 1, and is cleared to 0 if the direction bit is 0.

In master mode, I2C0SR<TRX> is cleared to 0 by hardware, after acknowledge is returned from the slave device, if the transmitted direction bit is 1, and is set to 1 if the direction bit is 0. If no acknowledge is returned, I2C0SR<TRX> remains unchanged.

I2C0SR<TRX> is cleared to 0 by hardware when a stop condition or arbitration lost is detected on the bus. Table 3.14.5 summarizes the operation of I2C0SR<TRX> in slave and master modes.

Note: When I2C0CR1<NOACK> = 1, the slave address detection and general call detection are disabled, and thus I2C0SR<TRX> remains unchanged.

Table 3.14.5 I2C0SR<TRX> operation in slave and master modes

Mode	Direction bit	Condition for state change	Changed <TRX> state
Slave mode	0	When the received slave address matches the slave address set in I2C0AR<SA>	0
	1		1
Master mode	0	When the ACK signal is returned.	1
	1		0

When I²C is used with the free data format, the slave address and direction bit are not recognized and bits immediately following a start condition are handled as data. Therefore, I2C0SR<TRX> is not changed by hardware.

3.14.6.6 Generation of Start and Stop Conditions

When I2C0SR<BB> = 0, writing 1 to I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<BB> and I2C0CR2<PIN> causes a start condition, the slave address written in the data buffer register and direction bit to be sent out on the bus. I2C0CR1<ACK> must be set to 1 before a start condition is generated.

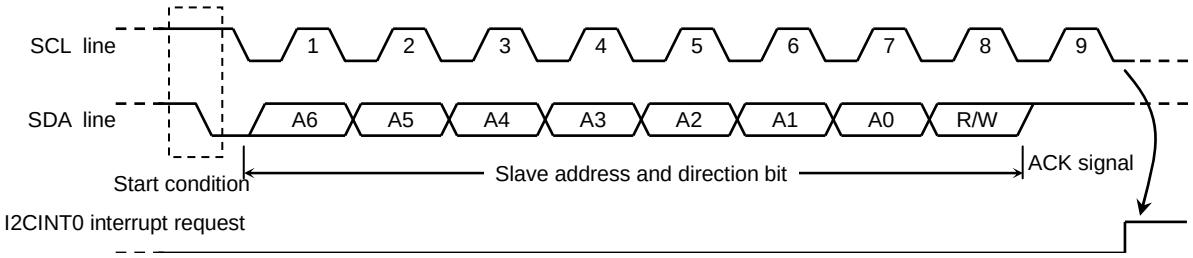
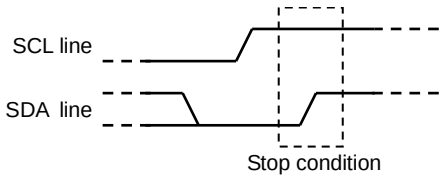


Figure 3.14.15 Start condition and slave address generation

When I2C0SR<BB> = 1, writing 1 to I2C0CR2<MST>, I2C0CR2<TRX>, I2C0CR2<PIN> and 0 to I2C0CR2<BB> initiates a sequence for sending out a stop condition on the bus.

At this time, if the SCL line is pulled low by another device, a stop condition is generated after the SCL line is released.



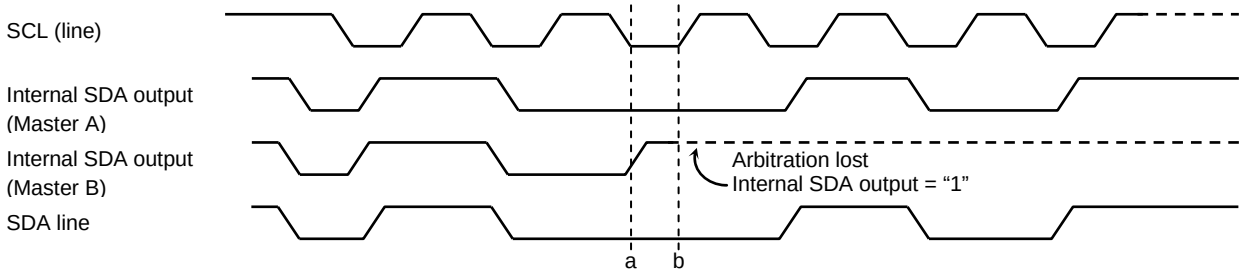


Figure 3.14.16 Stop condition generation

The bus status can be checked by reading I2C0SR<BB>. I2C0SR<BB> is set to 1 (bus busy) when a start condition is detected on the bus, and is cleared to 0 (bus free) when a stop condition is detected.

The following table shows typical setting examples according to the I2C0SR state.

Although the I2C0CR2<MST>, <TRX>, <BB> and <PIN> bits are given independent functions, they are used in typical combinations, as shown below, according to the I2C0SR setting.

I2C0SR			I2C0CR2				Operation
[7]MST	[5]BB	[4]PIN	[7]MST	[6]TRX	[5]BB	[4]PIN	
0	0	1	0	0	0	0	Wait for a start condition as a slave.
			1	1	1	1	Generate a start condition.
1	1	0	1	1	0	1	Generate a stop condition.
			0	0	0	1	Release the internal interrupt for restart.

When writing to these bits, be careful not to inadvertently change I2C0CR2<I2CM>.

3.14.6.7 Interrupt Service Request and Cancel

In master mode, after the number of bits specified by I2C0CR1<BC> and I2C0CR1<ACK> have been transferred, an I2CINT0 interrupt request is generated.

In slave mode, an I2CINT0 interrupt request is also generated by the following conditions in addition to the above condition:

- When I2C0CR1<NOACK> is 0, after the acknowledge signal is output to indicate that the received slave address has matched the slave address set in I2C0AR<SA>
- When I2C0CR1<NOACK> is 0, after the acknowledge signal is output to indicate that a general call has been received.
- When data transfer is completed after a matched slave address or a general call is received.

When an I2CINT0 interrupt request is generated, I2C0SR<PIN> is cleared to 0. While I2C0SR<PIN> is 0, I2C0CL is pulled low.

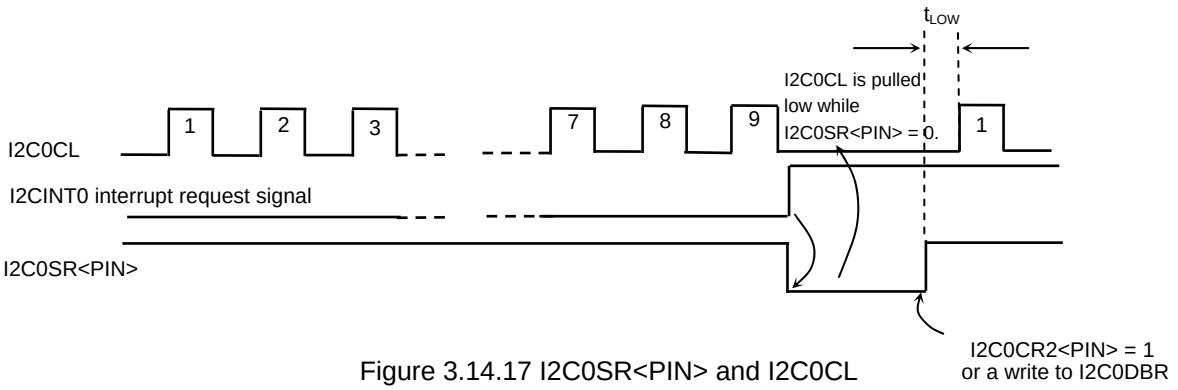


Figure 3.14.17 I2C0SR<PIN> and I2C0CL

Writing data into I2C0DBR sets I2C0SR<PIN> to 1.

It takes the t_{LOW} period for I2C0CL to be released after I2C0SR<PIN> is set to 1. I2C0CR2<PIN> can be set to 1 by writing 1 whereas it cannot be cleared to 0 by writing 0.

3.14.6.8 I²C Bus Mode

When I2C0CR2<I2CM> is set to 1, I²C bus mode is selected.

Before enabling I²C bus mode, make sure that the I2C0DA and I2C0CL pins are high and then set I2C0CR2<I2CM> to 1. Before initializing I²C, make sure that the bus is free and then clear I2C0CR2<I2CM> to 0.

Note: When I2C0CR2<I2CM> = 0, no value can be written to bits in the I2C0CR2 register other than I2C0CR2<I2CM>. Before setting I2C0CR2, write 1 to I2C0CR2<I2CM> to select I²C bus mode.

3.14.6.9 Software Reset

I²C has a software reset function. If I²C locks up due to noise, etc., it can be initialized by this function.

A software reset can be generated by writing 0y10 and then 0y01 to I2C0CR2<SWRES[1:0]>.

After a software reset, I²C is initialized except the I2C0CR2<I2CM> bit and the I2C0DBR register.

3.14.6.10 Arbitration Lost Detection Monitor

Since the I²C bus allows multiple masters to exist simultaneously, the bus arbitration feature must be implemented to ensure the integrity of transferred data.

The I²C bus uses data on the SDA line for bus arbitration.

The following shows an example of the bus arbitration procedure when two master devices exist on the bus simultaneously.

Master A and Master B output the same data until point “a”, where Master B outputs 1 and Master A outputs 0. This causes the SDA line to be pulled low by Master A since the SDA line is driven by the wired AND method.

When the SCL line rises at point “b”, the slave device captures the data on the SDA line, i.e., the data from Master A.

At this time, the data output from Master B becomes invalid. This is called “arbitration lost”. Master B that lost arbitration must release I2C0DA and I2C0CL so that Master A can use the bus without any hindrance. If more than one master outputs identical data on the first word, the arbitration procedure is continued on the second word.

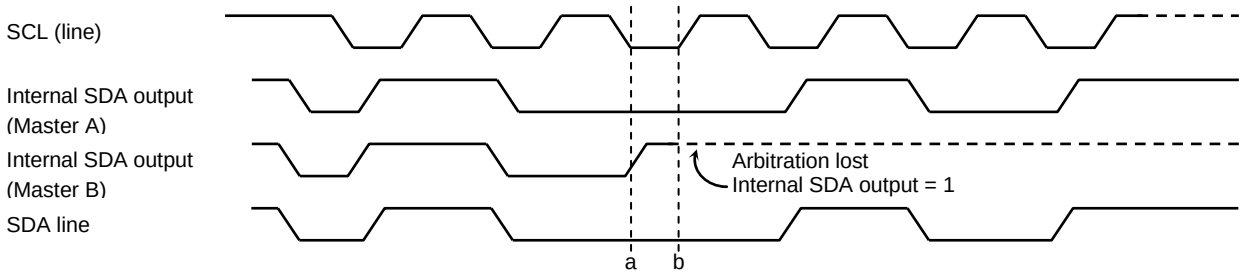


Figure 3.14.18 Arbitration lost

Master B compares the level of I2C0DA with the level of the SDA line on the bus on the rising edge of the SCL line. If the two levels do not match, arbitration lost is determined and I2C0SR<AL> is set to 1.

When I2C0SR<AL> is set to 1, I2C0SR<MST> and I2C0SR<TRX> are cleared to 0, thereby selecting slave receiver mode. Thus, after I2C0SR<AL> is set to 1, Master B stops clock output. After the data transfer on the bus is completed, I2C0SR<PIN> is cleared to 0 and I2C0CL is pulled low.

I2C0SR<AL> is cleared to 0 when data is written to or read from I2C0DBR or when data is written to I2C0CR2.

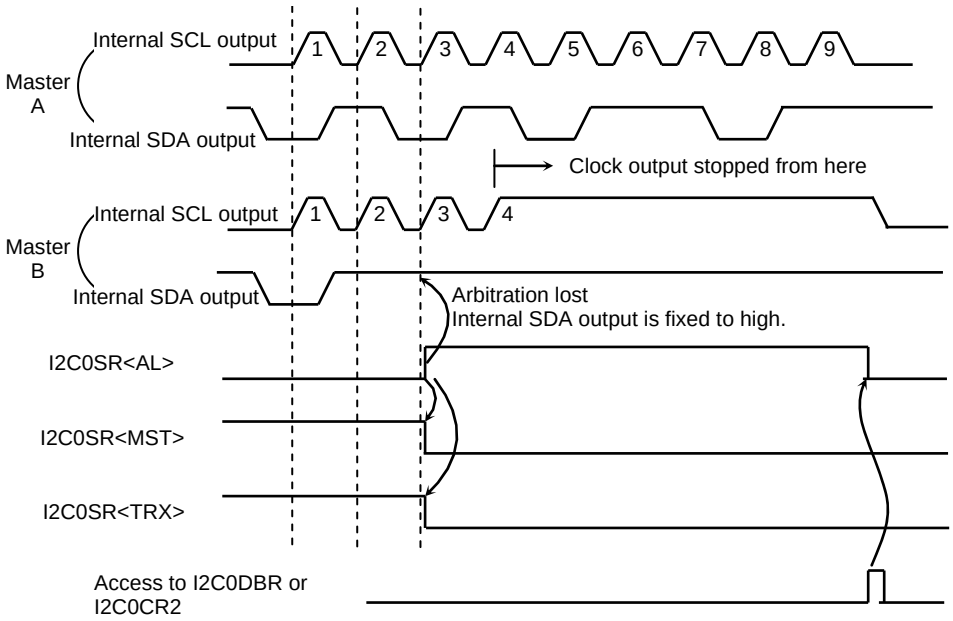


Figure 3.14.19 Arbitration lost operation (with internal flags associated with Master B)

3.14.6.11 Slave Address Match Detection Monitor

I²C bus mode (I2C0AR<ALS> = 0) allows slave address match detection when slave mode is selected.

Clearing I2C0CR1<NOACK> to 0 enables the slave address match detection. When a general call is received or the slave address sent from the master matches the slave address set in I2C0AR<SA>, I2C0SR<AAS> is set to 1.

Setting I2C0CR1<NOACK> to 1 disables the slave address match detection. Even if a general call is received or the salve address sent from the master matches the slave address set in I2C0AR<SA>, I2C0SR<AAS> remains 0.

When the free data format is used (I2C0AR<ALS> = 1), it is not used as address match detection, and I2C0SR<AAS> is set to 1 upon receipt of the first word of data. It is cleared to 0 when data is written to or read from I2C0DBR.

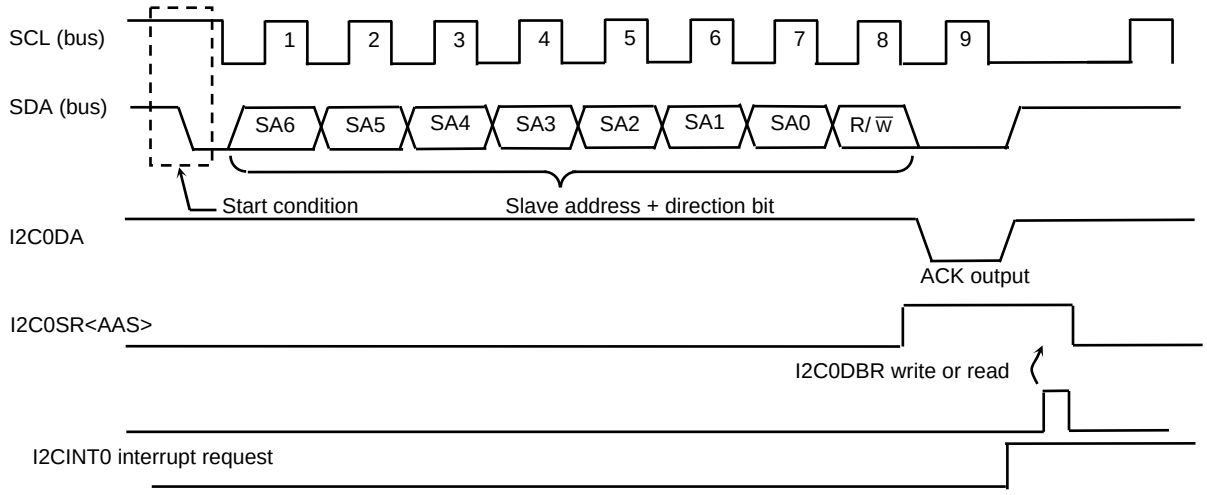


Figure 3.14.20 Changes in the slave address match monitor

3.14.6.12 General Call Detection Monitor

I²C bus mode (I2C0AR<ALS> = 0) also allows the detection of a general call as well as slave address match in slave mode.

When I2C0CR1<NOACK> = 0, I2C0SR<AD0> is set to 1 when a general call (8 bits received immediately after a start condition are all 0s) is received. (At this time, I2C0SR<AAS> is also set to 1.)

Setting I2C0CR1<NOACK> to 1 disables the slave address match detection and general call detection. I2C0SR<AD0> remains 0 even if a general call is received. (At this time, I2C0SR<AAS> also remains 0.)

I2C0SR<AD0> is cleared to 0 when a start or stop condition is detected on the bus.

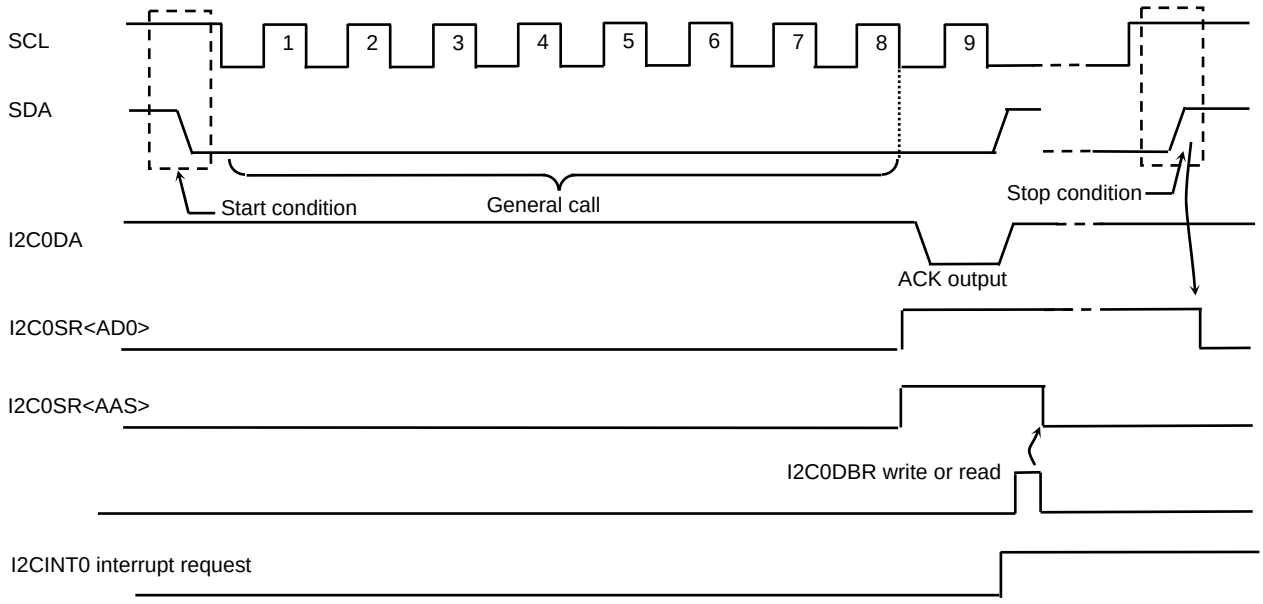


Figure 3.14.21 Changes in the general call detection monitor

3.14.6.13 Last Received Bit Monitor

I2C0SR<LRB> stores the SDA line value captured on every rising edge of the SCL line.

When acknowledge operation is enabled, the acknowledge signal is read from I2C0SR<LRB> immediately after generation of an I2CINT0 interrupt request.

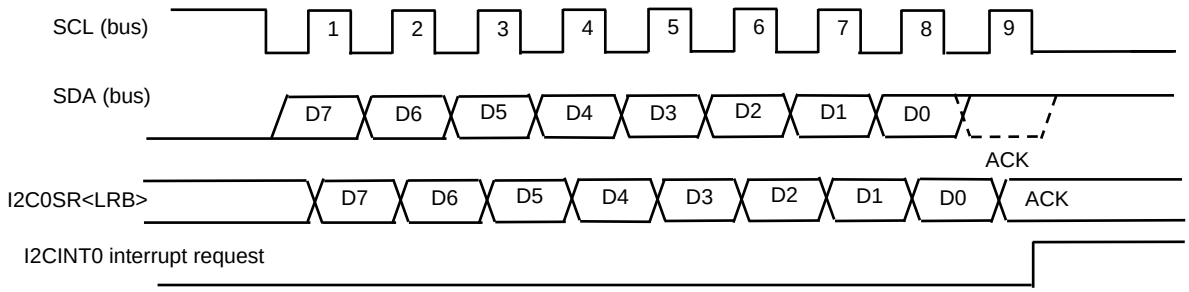


Figure 3.14.22 Changes in the last received bit monitor

3.14.6.14 Setting the Slave address and Address Recognition Mode

To use I²C in I²C bus mode, clear I2C0AR<ALS> to 0 and set a slave address in I2C0AR<SA>.

To use the free data format in which slave addresses are not recognized, set I2C0AR<ALS> to 1. When I²C is used with the free data format, the slave address and direction bit are not recognized and bits immediately following a start condition are handled as data.

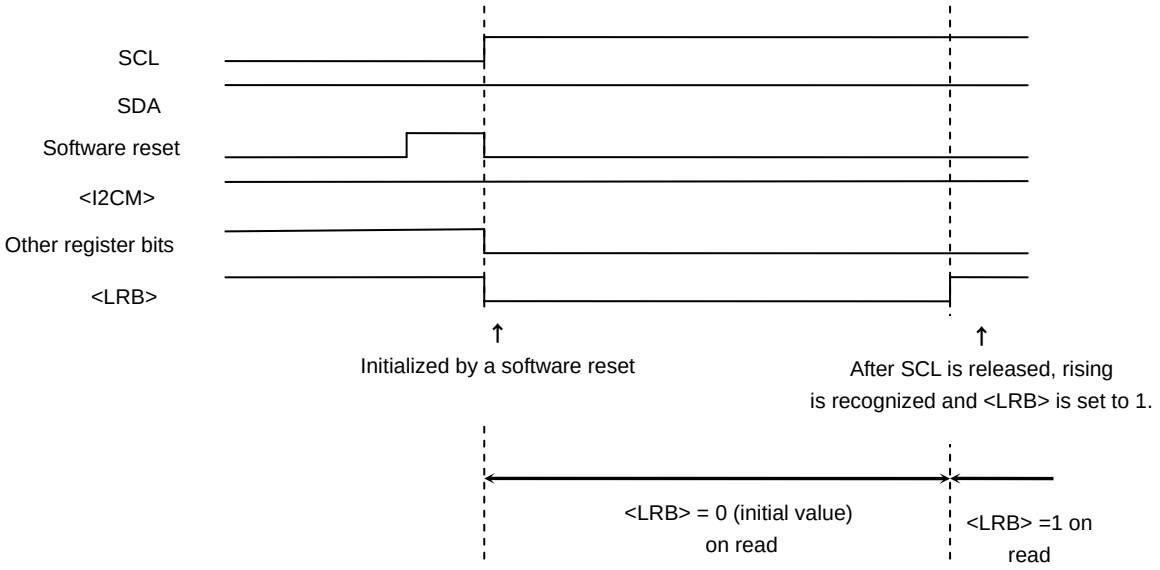
[Notes on Specifications]

3.14.6.15 Register Values after a Software Reset

A software reset initializes the I²C registers other than I2C0CR2<I2CM> and internal circuitry and releases the SCL and SDA lines. (Refer to Section 3.14.6.3 "(2) Clock synchronization".)

However, depending on read timing after a software reset, reading I2C0SR<LRB> may return a value other than the initial value (0).

<When a software reset releases SCL (0→1) while SDA = 1 >



3.15 SSP (Synchronous Serial Port)

This LSI contains the SSP (SSP: Synchronous Serial Port) comprised of one channel.

The SSP has the following features:

	Channel 0
Communication protocol	Synchronous serial communication that includes SPI : 3 types
Operation mode	Master/ Slave mode support
Transmit FIFOs	16-bit wide, 8 locations deep
Receive FIFOs	16-bit wide, 8 locations deep
Transmit/Receive data size	4 to 16 bits
Interrupt type	Transmit interrupt Receive interrupt Receive overrun interrupt Timeout interrupt
Baud rate	Master mode: $f_{PCLK}/2$ (Max 20 Mbps) Slave mode: $f_{PCLK}/12$ (Max 8.33 Mbps)
DMA	Support
Internal test function	Internal loopback test mode available
Control pins	SP0CLK SP0FSS SP0DO SP0DI

3.15.1 Block Diagrams

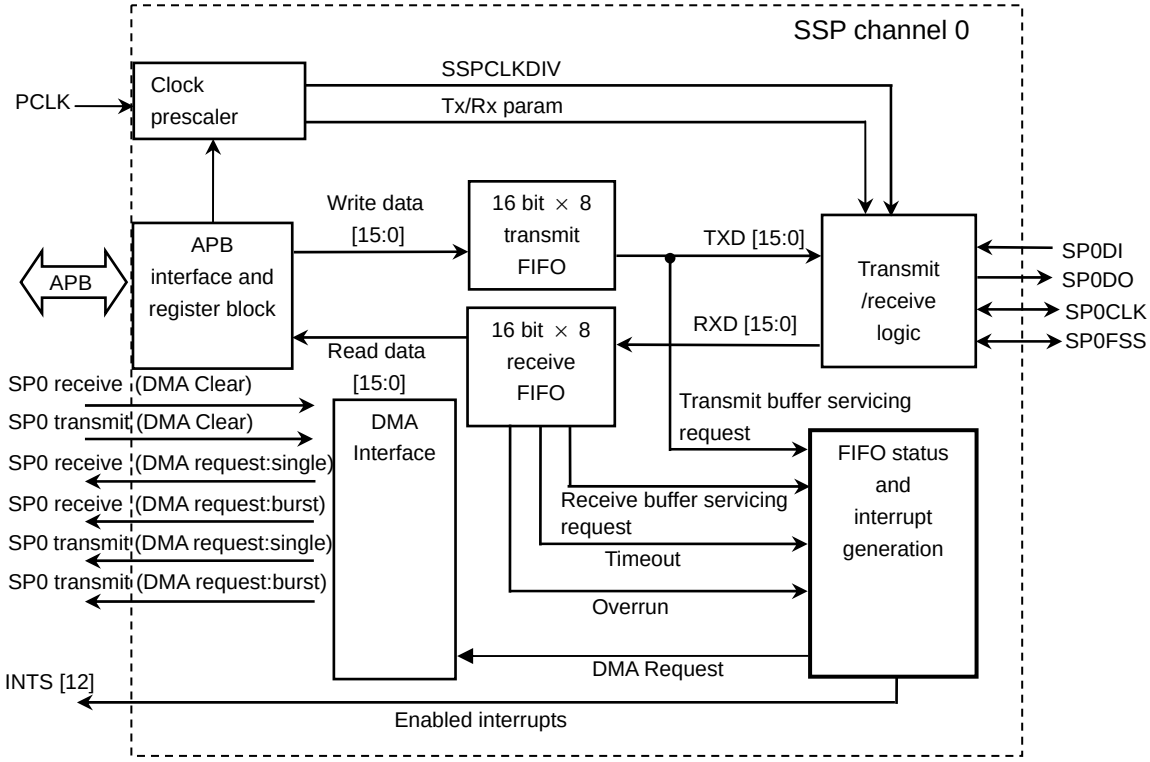


Figure 3.15.1 Block diagram of SSP

3.15.2 SSP Overview

The SSP is an interface for serial communication with peripheral devices that have three types of synchronous serial interfaces.

The SSP performs serial-to-parallel conversion on data received from a peripheral device. The transmit and receive paths are buffered with a 16-bit wide, 8 locations deep independent transmit FIFO and receive FIFO in transmit mode and receive mode, respectively. Serial data is transmitted on SP0DO and received on SP0DI.

The SSP contains a programmable prescaler to generate the serial output clock SP0CLK from the input clock PCLK. The SSP operating mode, frame format and size are programmed through the control registers SSP0CR0 and SSP0CR1.

(1) Clock prescaler

When configured as a master, a clock prescaler comprising two serially linked free-running counters is used to provide the serial output clock SP0CLK.

This clock prescaler can be programmed, through the SSP0CPSR register, to divide PCLK by a factor of 2 to 254 in steps of two. By not using the least significant bit of the SSP0CPSR register, division by an odd number cannot be programmed.

The output of the prescaler is further divided by a factor of 1 to 256, obtained by adding one to the value programmed in the SSP0CR0 control register, to give the master output clock SP0CLK.

Bit rate = $f_{PCLK} / (CPSDVSR \times (1 + SCR))$

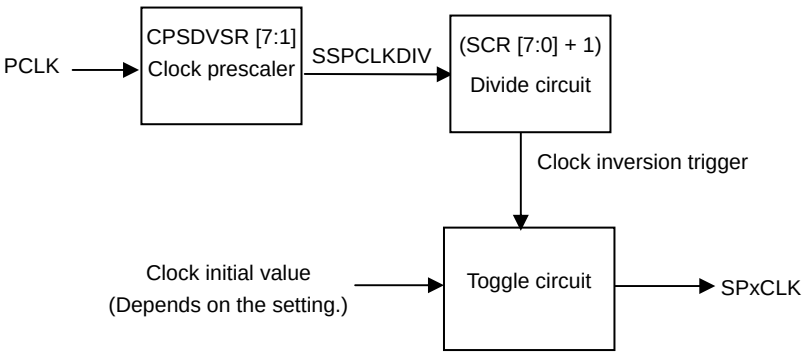


Figure 3.15.2 Block diagram of Clock prescaler

(2) Transmit FIFO

The transmit FIFO buffer of 16-bit wide, 8 locations deep are shared with Master mode and Slave mode.

(3) Receive FIFO

The receive FIFO buffer of 16-bit wide, 8 locations deep are shared with Master mode and Slave mode.

(4) Interrupt

Four individual maskable interrupts are supported by the SSP.

A combined interrupt output is also generated as an OR function of the individual interrupt requests.

- Transmit interrupt: Indicates that TxFIFO is more than half empty.
  (Number of valid entries in TxFIFO ≤ 4)
- Receive interrupt: Indicates that RxFIFO is more than half full.
  (Number of valid entries in RxFIFO ≥ 4)
- Timeout interrupt: Indicates that data is present in RxFIFO and has not been read before a timeout period expires.
- Receive overrun interrupt: Indicates that data is written to RxFIFO when it is full.

Each of the four individual maskable interrupts can be masked by setting the appropriate bit in the interrupt mask set and clear register. Setting the appropriate mask bit High enables the interrupt.

(a) Transmit interrupt

The transmit interrupt is asserted when there are four or less valid entries in the transmit FIFO. The transmit interrupt is generated even when SSP operation is disabled (SSPxCR1<SSE> = 0).

The initial transmit data can be written into the transmit FIFO by using this interrupt.

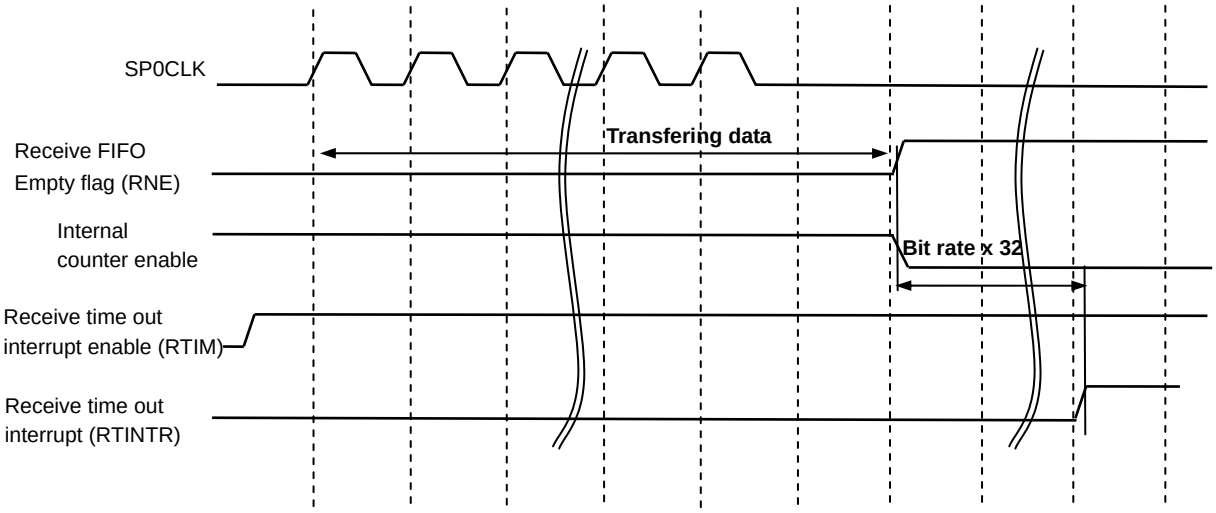
(b) Receive interrupt

The receive interrupt is asserted when there are four or more valid entries in the receive FIFO.

The transmit and receive interrupt requests are generated and cleared dynamically by monitoring the number of valid entries in the transmit and receive FIFOs. No interrupt request clear register is available.

(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 duration of 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. The timeout interrupt is generated both in master and slave modes. When the timeout interrupt is generated, read all the data in the receive FIFO. Data can be transmitted/received without reading all the data in the receive FIFO provided that the receive FIFO has empty space for receiving the data to be transmitted. The timeout interrupt is cleared when a transfer is started. If a transfer is performed when the receive FIFO is full, the timeout interrupt is cleared and the overrun interrupt is generated.



(d) Receive overrun interrupt

When the receive FIFO is already full and an additional (9th) data frame is received, the receive overrun interrupt is asserted immediately after the completion of the current transfer. Once the receive overrun error occurs, any subsequent data received (including the 9th data frame) is invalid and discarded. However, if the data in the receive FIFO is read while the 9th data frame is being received (before the receive overrun interrupt occurs), the 9th data frame is written into the receive FIFO as valid data. To perform proper transfer operation after the receive overrun error occurred, write 1 to the receive overrun interrupt clear register and then read all the data in the receive FIFO. Data can be transmitted/received without reading all the data in the receive FIFO provided that the receive FIFO has empty space for receiving the data to be transmitted. If the receive FIFO is not read (when it is not empty) for a fixed duration of 32-bit period (bit rate) after the receive overrun interrupt has been cleared, the timeout interrupt is generated.

(e) Combined interrupt

The individual masked sources of the above four interrupts are also combined into a single interrupt. The combined interrupt INTS [12] is asserted if any of the four interrupts is asserted.

(5) DMAC

The SSP provides an interface to connect to a DMA controller.

3.15.3 SPP Operation

(1) Configuring the SSP

The SSP communication protocol must be configured while the SSP is disabled.

Select master or slave mode by setting the control register SSPOCR0 and SSP0CR1under either of the following protocols.. The communication rate need also be set by programming the prescale register SSP0CPSR and SSP0CR0<SCR>.

This SSP supports the following frame formats:

- SPI
- SSI
- Microwire

(2) Enabling the SSP

Transmission of data begins when SSP operation is enabled after transmit data has been written into the transmit FIFO or when transmit data is written into the transmit FIFO after SSP operation has been enabled.

However, if the transmit FIFO has four entries or less when SSP operation is enabled, the transmit interrupt will be generated. It is possible to use this interrupt to write the initial transmit data.

Note: When using the SPI slave mode without using the FSS pin, be sure to write 1 byte or more of data into the transmit FIFO before enabling SSP operation. If SSP operation is enabled while the transmit FIFO is empty, transfer data cannot be output properly.

(3) Clock ratios

The PCLK frequency setting must satisfy the following conditions:

[Master mode]

f_{SP0CLK} (max): f_{PCLK} / 2

f_{SP0CLK} (min): f_{PCLK} / (254 × 256)

[Slave mode]

f_{SP0CLK} (max): f_{PCLK} / 12

f_{SP0CLK} (min): f_{PCLK} / (254 × 256)

(4) Frame format

Each frame format is between 4 to 16 bits long depending on the size of data programmed, and is transmitted starting with the MSB.

・Serial clock (SP0CLK)

For SSI and Microwire frame formats, the serial clock (SP0CLK) is held Low while the SSP is idle. For SPI frame format, the serial clock (SP0CLK) is held inactive while the SSP is idle. SP0CLK is output at the specified bit rate only while data is being transmitted.

・Serial frame (SP0FSS)

For SPI and Microwire frame formats, the serial frame (SP0FSS) pin is active Low, and is asserted during the entire transmission of the frame.

For SSI frame format, the SP0FSS pin is asserted for one bit rate period prior to the transmission of each frame. For this frame format, output data is transmitted on the rising edge of SP0CLK, and input data is received on the falling edge.

・Microwire frame format

The Microwire format uses a special master-slave messaging technique, which operates at half-duplex. In this mode, when a frame begins, an 8-bit control message is transmitted to the slave. During this transmission, no incoming data is received by the SSP. After the message has been sent, the slave decodes it and, after waiting one serial clock period 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.

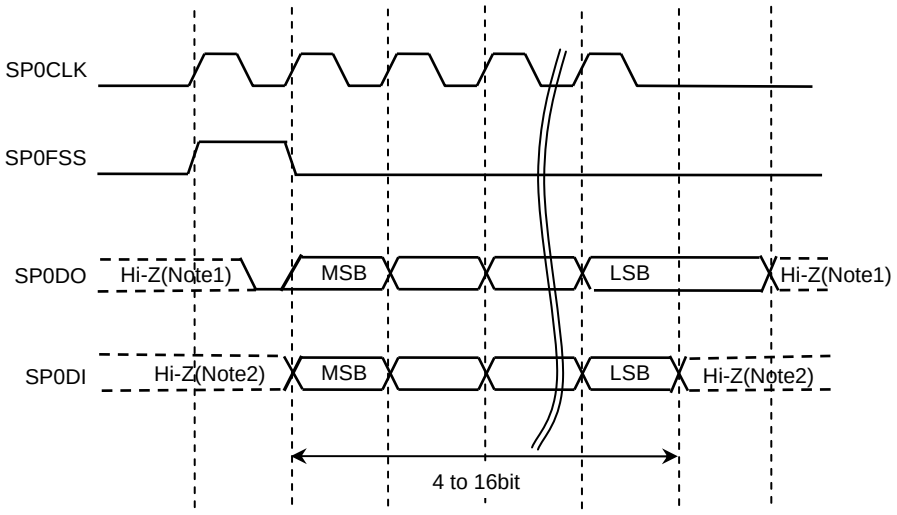
The details of each frame format are described below.

1) SSI frame format

In this mode, SP0CLK and SP0FSS are forced Low and the transmit data line SP0DO is put in the Hi-Z state whenever the SSP is idle. When data is written into the transmit FIFO, the master pulses the SP0FSS line High for one SP0CLK period. The transmit data is transferred from the transmit FIFO to the transmit serial shift register. On the next rising edge of SP0CLK, the MSB of the 4 to 16-bit data frame is shifted onto the SP0DO pin.

Likewise, the MSB of the received data is input to the SP0DI pin on the falling edge of SP0CLK. The received data is transferred from the serial shift register to the receive FIFO on the first rising edge of SP0CLK after the LSB has been latched.

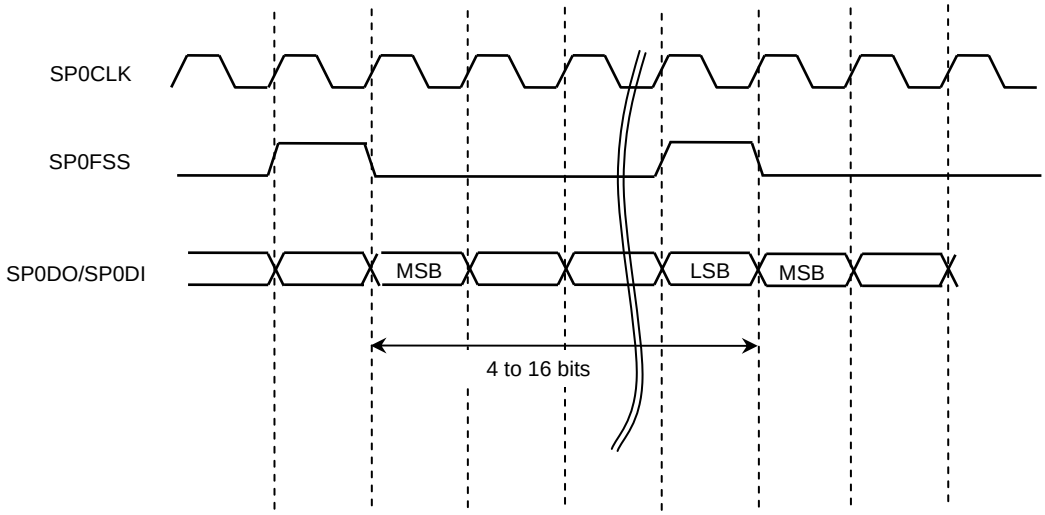
SSI frame format (single transfer)



Note1) When transmission is disable , SP0DO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note2) SP0DI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

SSI frame format (continuous transfer)



Note1) When transmission is disable , SP0DO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note2) SP0DI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

2) SPI

The SPI interface is a four-wire interface where the SP0FSS signal behaves as a slave select. The main feature of the SPI format is that the operation timing of SP0CLK is programmable through the <SPO> and <SPH> bits in the SSP0CR0 control register.

SSP0CR0<SPO>

SSP0CR0<SPO> specifies the SP0CLK level during idle periods.

<SPO> = 1: SP0CLK is held High.

<SPO> = 0: SP0CLK is held Low.

SSP0CR0<SPH>

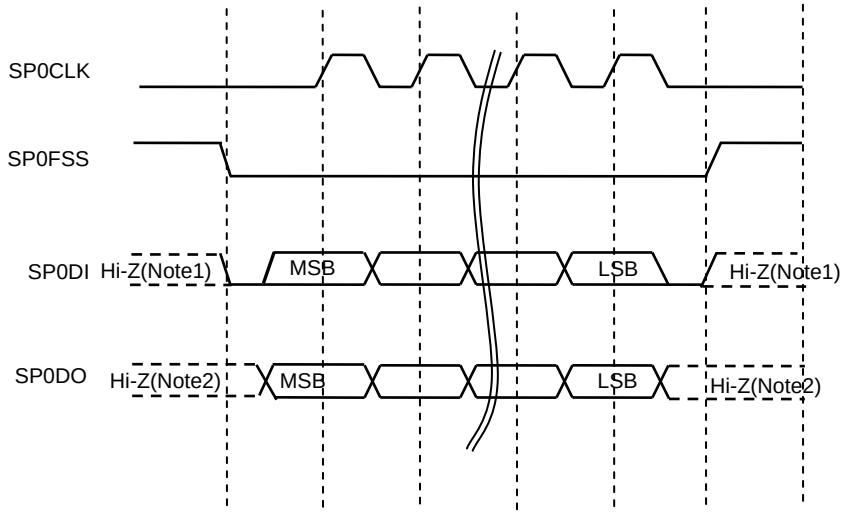
SSP0CR0<SPH> selects the clock edge for latching data.

SSP0CR0<SPH> = 0: Data is latched on the first clock edge.

SSP0CR0<SPH> = 1: Data is latched on the second clock edge.

SPI operation examples:

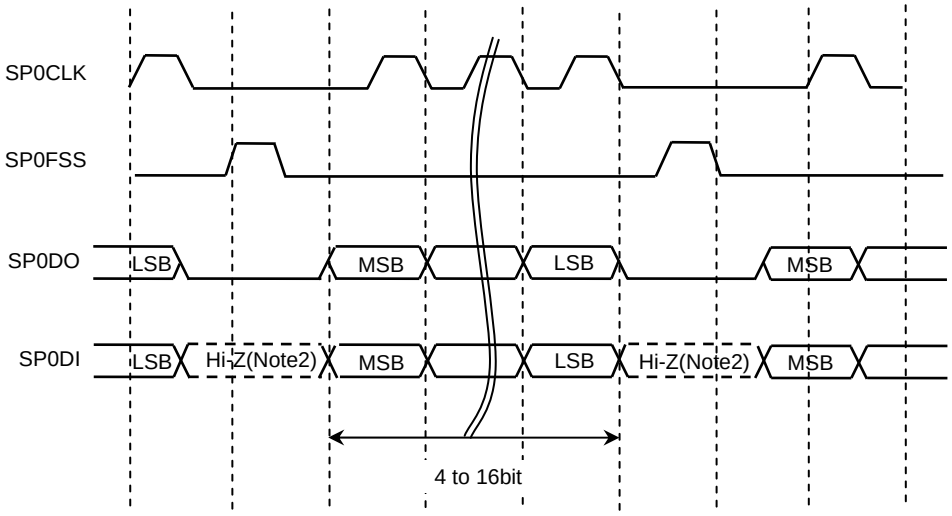
SPI (single transfer, <SPO> = 0 & <SPH> = 0)



Note1) When transmission is disable , SP0DO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note2) SP0DI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

SPI (continuous transfer, <SPO> = 0 & <SPH> = 0)



Note1) When transmission is disable , SP0DO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note2) SP0DI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

With this setting, during idle periods:

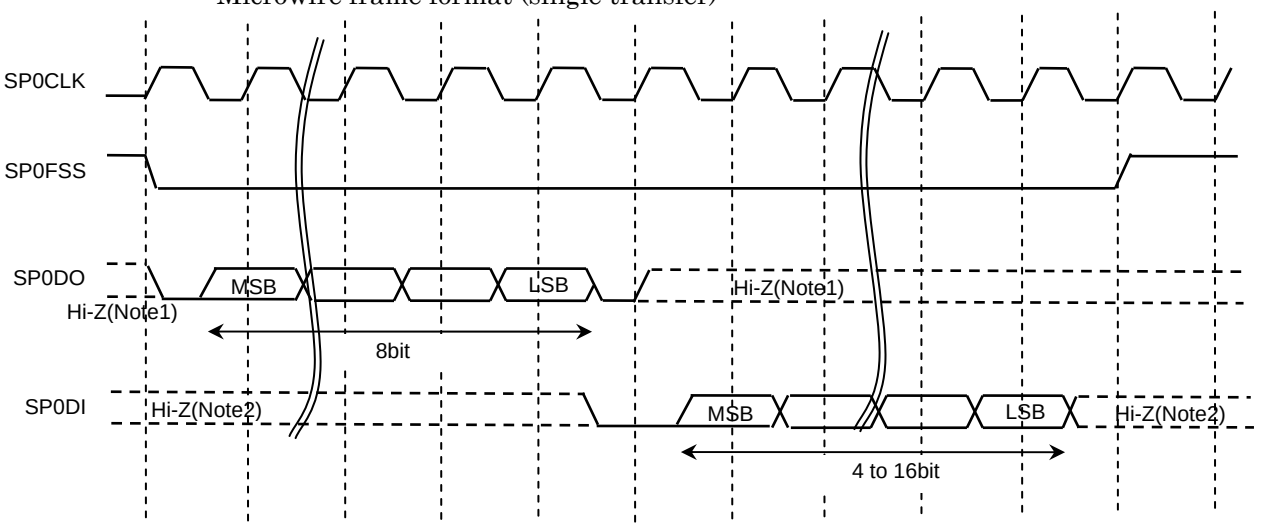
- The SP0CLK signal is forcedly set to Low
- SP0FSS is forcedly set to High
- The transmit data line SP0DO is arbitrarily set to LOW.

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 one 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.

3) Microwire frame format

Microwire frame format (single transfer)



Note1) When transmission is disable , SP0DO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note2) SP0DI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

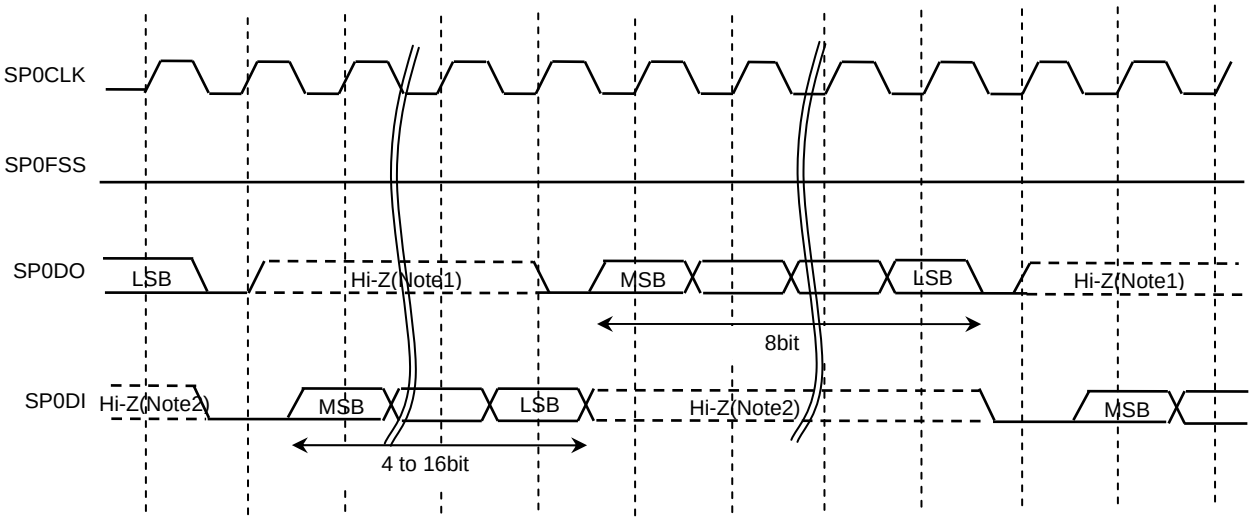
Though the Microwire format is similar to the SPI format, it uses the master/slave message transmission method for half-duplex communications. Each serial transmission is started by an 8-bit control word, which is sent to the off-chip slave device. During this transmission, the SSP does not receive input 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)



Note1) When transmission is disable , SP0DO terminal doesn't output and is high impedance status. This terminal needs to add suitable pull-up/down resistance to valid the voltage level.

Note2) SP0DI terminal is always input and internal gate is open. In case of transmission signal will be high impedance status, this terminal needs to add suitable pull-up/down resistance to valid the voltage level.

For continuous transfers, data transmission begins and ends in the same manner as a single transfers. 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, SSP0DMACR.

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

3.15.4 Description of Registers

The following lists the SFRs:

- SSP0

Base address = 0xF200_2000

Register Name	Address (base +)	Description
SSP0CR0	0x0000	SSP0 Control register 0
SSP0CR1	0x0004	SSP0 Control register 1
SSP0DR	0x0008	SSP0 Data register
SSP0SR	0x000C	SSP0 Status register
SSP0CPSR	0x0010	SSP0 Clock prescale register
SSP0IMSC	0x0014	SSP0 Interrupt mask set and clear register
SSP0RIS	0x0018	SSP0 Raw interrupt status register
SSP0MIS	0x001C	SSP0 Masked interrupt status register
SSP0ICR	0x0020	SSP0 Interrupt clear register
SSP0DMACR	0x0024	DMA Control register
–	0x0028 to 0xFFC	Reserved

1. SSP0CR0 (SSP0 Control register 0)

Address = (0xF200_2000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:8]	SCR	R/W	0y0	Parameter for setting the serial clock rate: 0x00 to 0xFF (See [Description] below.)
[7]	SPH	R/W	0y0	SPCLK phase 0y0: Data is latched on the first clock edge 0y1: Data is latched on the second clock edge (Applicable to SPI frame format only. See “2) SPI”.)
[6]	SPO	R/W	0y0	SPCLK polarity 0y0: SP0CLK is held Low 0y1: SP0CLK is held High (Applicable to SPI frame format only. See “2) SPI”.)
[5:4]	FRF	R/W	0y00	Frame format: 0y00: SPI frame format 0y01: SSI frame format 0y10: 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

[Description]

a. <SCR>

The <SCR> value is used to generate the transmit and receive bit rate of the SSP.

The bit rate is:

Bit rate = f_{PCLK} / (<CPSDVSR> × (1 + <SCR>))

Please refer to SSPxCPSR register about <CPSDVSR>.

2. SSP0CR1 (SSP0 Control register 1)

Address = (0xF200_2000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3]	SOD	R/W	0y0	Slave mode SP0DO output disable: 0y0: Enable 0y1: Disable
[2]	MS	R/W	0y0	Master/slave mode select: 0y0: The device is a master. 0y1: The device is a slave.
[1]	SSE	R/W	0y0	SSP enable: 0y0: Disable 0y1: Enable
[0]	Reserved	R/W	0y0	Write as zero.

[Description]

a. <SOD>

Slave mode output disable. This bit is relevant only in the slave mode (<MS> = 1).

b. <MS>

Master/slave mode select. When transmit mode with Slave mode, must be set it in the following order.

1. Set to Slave mode (<MS> = 1)
2. Set transmit data to FIFO
3. Set SSP to enable (<SSE> = 1)

3. SSP0DR (SSP0 Data register)

Address = (0xF200_2000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	DATA	R/W	0x0000	Transmit/receive FIFO data: 0x00 to 0xFF

[Description]

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 the transmit logic. The receive logic automatically right-justifies.

4. SSP0SR (SSP0 Status register)

Address = (0xF200_2000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined.
[4]	BSY	RO	0y0	Busy flag: 0y0: IDLE 0y1: Busy
[3]	RFF	RO	0y0	Receive FIFO full flag: 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.

[Description]

a. <BSY>

This bit indicates, when set to 1 (BSY = 1), that a frame is currently being transmitted or received or the transmit FIFO is not empty.

5. SSP0CPSR (SSP0 Clock prescale register)

Address = (0xF200_2000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	CPSDVSR	R/W	0x0000	Clock prescale divisor: Must be an even number from 2 to 254.

[Description]

a. <CPSDVSR>

Clock prescale divisor. Must be an even number from 2 to 254, depending on the frequency of PCLK. The least significant bit always returns 0y0 on reads.

6. SSP0IMSC (SSP0 Interrupt mask set and clear register)

Address = (0xF200_2000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as 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

[Description]

a. <TXIM>

Enables or disables interrupts that are generated when TxFIFO is half empty or less.

b. <RXIM>

Enables or disables interrupts that are generated when RxFIFO is half full or less.

c. <RTIM>

Enables or disables interrupts that are generated when the data in RxFIFO is not read out before the timeout period expires.

d. <RORIM>

Enables or disables interrupts that are generated when data is written to RxFIFO while it is full.

7. SSP0RIS (SSP0 Raw interrupt status register)

Address = (0xF200_2000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined.
[3]	TXRIS	RO	0y0	Transmit interrupt status prior to enable gate: 0y0: No interrupt 0y1: Interrupt requested
[2]	RXRIS	RO	0y0	Receive interrupt status prior to enable gate: 0y0: No interrupt 0y1: Interrupt requested
[1]	RTRIS	RO	0y0	Receive timeout interrupt status prior to enable gate: 0y0: No interrupt 0y1: Interrupt requested
[0]	RORRIS	RO	0y0	Receive overrun interrupt status prior to enable gate: 0y0: No interrupt 0y1: Interrupt requested

8. SSP0MIS (SSP0 Masked interrupt status register)

Address = (0xF200_2000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined.
[3]	TXMIS	RO	0y0	Transmit interrupt status after enable gate: 0y0: No interrupt 0y1: Interrupt requested
[2]	RXMIS	RO	0y0	Receive interrupt status after enable gate: 0y0: No interrupt 0y1: Interrupt requested
[1]	RTMIS	RO	0y0	Receive timeout interrupt status after enable gate: 0y0: No interrupt 0y1: Interrupt requested
[0]	RORMIS	RO	0y0	Receive overrun interrupt status after enable gate: 0y0: No interrupt 0y1: Interrupt requested

9. SSP0ICR (SSP0 Interrupt clear register)

Address = (0xF200_2000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	RTIC	WO	Undefined	Receive timeout interrupt flag clear: 0y0: Invalid 0y1: Clear
[0]	RORIC	WO	Undefined	Receive overrun interrupt flag clear: 0y0: Invalid 0y1: Clear

10. SSP0DMACR (SSP0DMA Control register)

Address = (0x4001_D000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	TXDMAE	R/W	0y0	DMA Enable for Transmit FIFO : 0y0: Disable 0y1: Enable
[0]	RXDMAE	R/W	0y0	DMA Enable for Receive FIFO : 0y0: Disable 0y1: Enable

3.16 USB Device Controller

3.16.1 System Overview

1) Conforming to Universal Serial Bus Specification Rev. 2.0
2) Supports both High-Speed and Full-Speed (Low-Speed is not supported).
3) Supports Chirp.
4) USB protocol processing
5) Detects SOF/USB_RESET/SUSPEND/RESUME.
6) Generates and checks packet IDs.
7) Generates and checks data synchronization bits (DATA0/DATA1/DATA2/MDATA).
8) Checks CRC5, generates and checks CRC16.
9) Supports PING.
10) Supports 4 transfer modes (Control/Interrupt/Bulk/Isochronous).
11) Supports 4 endpoints:

Endpoint 0:	Control	64 bytes × 1 FIFO
Endpoint 1:	Bulk (IN)	512 bytes × 2 FIFOs
Endpoint 2:	Bulk (OUT)	512 bytes × 2 FIFOs
Endpoint 3:	Interrupt (IN)	64 bytes × 1 FIFO

12) Supports Dual Packet Mode (except for Endpoint 0).
13) Interrupt source signal to Interrupt controller: INTS[21]

3.16.1.1 System Structure

The USB device controller consists of the core part called UDC2 and the bus bridge part called UDC2AB which enables connection with the AHB bus.

In this section, the circuit function is outlined first. Then, section 3.16. 2 describes the configuration of the UDC2AB bus bridge, and section 3.16.3 describes the configuration of UDC2.

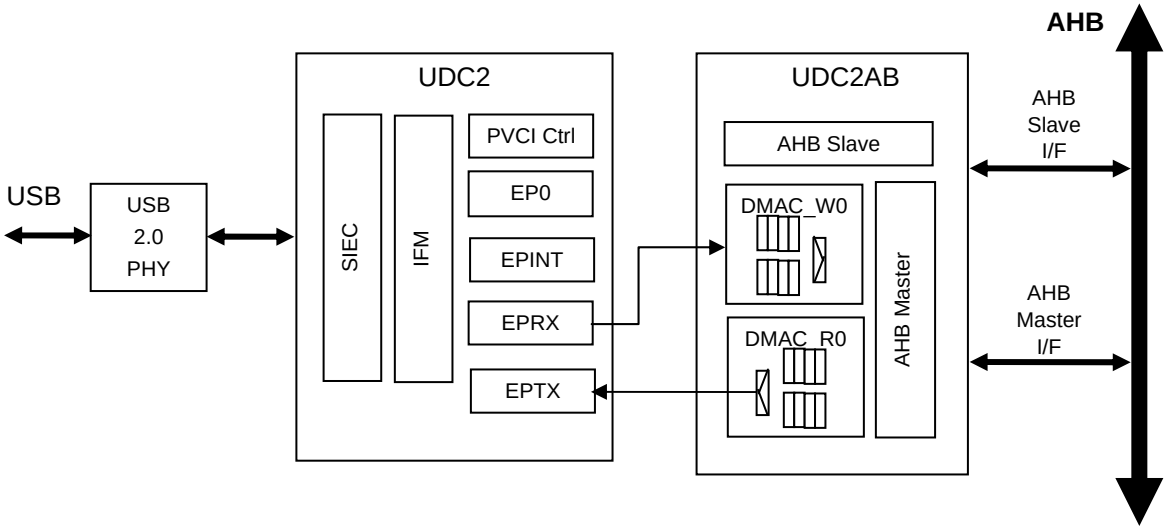
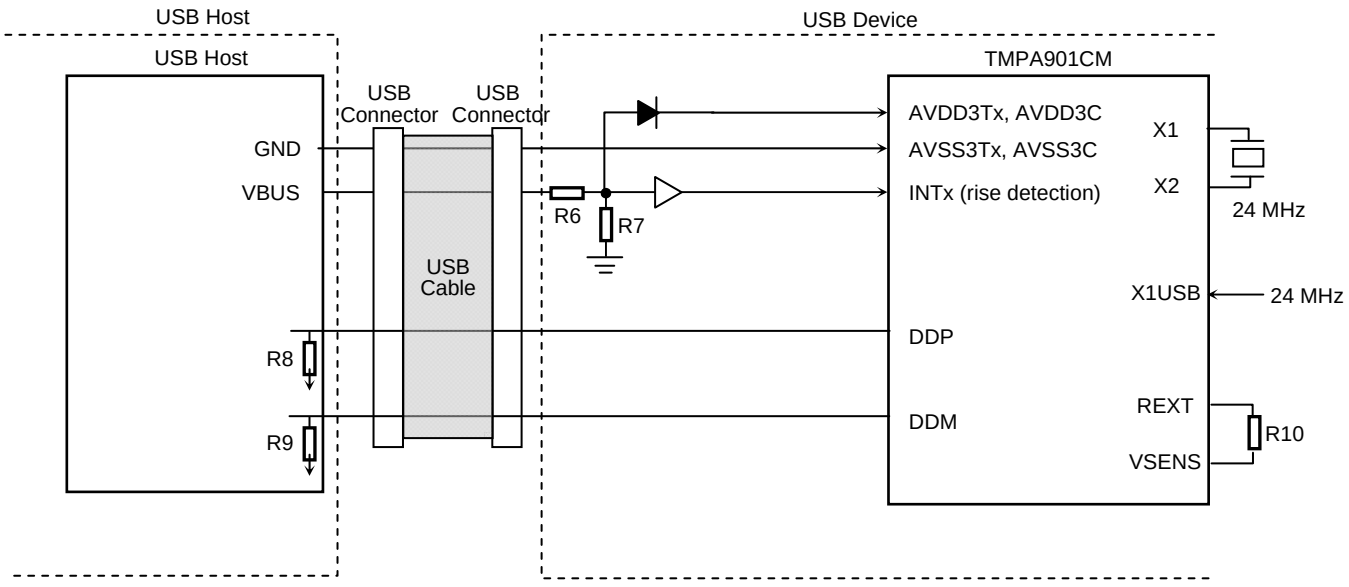


Figure 3.16.1 Block diagram of the USB device controller

3.16.1.2 Example of Connection



The above diagram shows the connections required for using the USB controller in the TMPA901CM.

(1) Pulling up of the DDP pin

The USB specification requires that the DDP pin be pulled up for Full-Speed communication. An internal pull-up resistor is provided, and no external circuit is required.

(2) Insertion of series resistance for the DDP and DDM pins

The USB specification requires that series resistance be inserted for each of the DDP and DDM pins. Internal series resistance is provided for each of these pins, and no external circuit is required.

(3) Detection of connector connection

How to detect connector connection with VBUS (5 V) is explained as an example.

As shown in the connection example above, R6 and R7 for dividing resistance should be connected to the VBUS pin in such a way as to assert the interrupt pin High (3.3 V) when power is connected. By detecting this interrupt by software, connector connection can be detected.

Note: If the waveform rises slowly, it is recommended to insert appropriate buffering for waveform shaping.

Recommended values: R6 = 60 kΩ, R7 = 100 kΩ

(VBUS consumption current in suspended state < 500 μA)

(4) 24-MHz clock input

The USB device controller requires a 24-MHz clock. This clock input can be implemented in two ways. One is to connect a 24-MHz resonator to the X1 and X2 pins and the other is to input a 24-MHz clock from the X1USB pin. SYSCR0<USBCLKSEL> is used to select either of these methods. In whichever case, the clock precision must be ±100 ppm or less.

(5) Pull-down resistors on the USB host

The USB specification requires that the DDP and DDM pins be pulled down at the USB host end.

Recommended values: R8 = 15 kΩ, R9 = 15 kΩ

(6) Resistor for USB_PHY

It is necessary to connect a resistor between the REXT pin and the VSENS pin. R10 should be 12 kΩ (with an error within ±1.0%).

Note: The above connections, resistor values and other information are provided as examples and their operations are not guaranteed. Please be sure to check the latest USB specification and to perform operation checks on the actual set.

3.16.2 UDC2AB AHB Bus Bridge

UDC2AB (UDC2 AHB Bridge) is the bridge circuit between Toshiba USB-Spec 2.0 Device Controller (hereinafter "UDC2") and AHB. UDC2AB has the DMA controller that supports the AHB master transfer and controls transfer between the specified address on AHB and the Endpoint-FIFO (Endpoint I/F) inside UDC2.

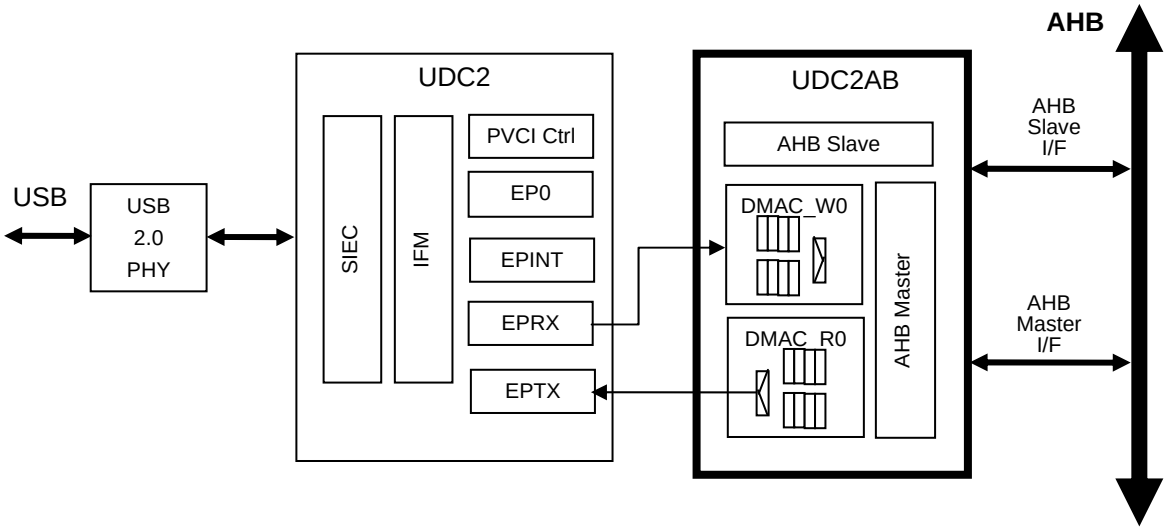


Figure 3.16.2 UDC2AB block diagram

3.16.2.1 Functions and Features

UDC2AB has the following functions and features:

(1) Connection with UDC2

There is no specific restriction on the endpoint configuration for the UDC2 to be connected. However, the DMA controller in UDC2AB (AHB master function) can be connected with only one Rx-EP and one Tx-EP. Accesses to other endpoints (including EP0) should be made through PVCI I/F of UDC2 using the AHB slave function. Please note the EPx_FIFO register of a UDC2 endpoint in master transfer with the DMA controller cannot be accessed through PVCI I/F.

If the maximum packet size of the endpoint to be connected with the AHB master read function will be an odd number, there will be some restrictions on the usage. See section 3.16.2.9 "(3)Setting the maximum packet size in Master Read transfers" for more information.

(2) AHB functions

AHB Master and AHB Slave functions are provided.

(a) AHB Master function

Specifications of the AHB Master function:

- Has two DMA channels; one each is allocated to the Rx-EP and the Tx-EP.
- Single and Burst (INCR/INCR8) transactions are supported.
- Split transaction is not supported.
- Little Endian is supported.
- Protection Control is not supported.
- Early Burst Termination is supported.
- Address width and data width are both 32 bits.
- Transaction sizes in bytes or words are supported.

The image of Endian conversion is as shown below.

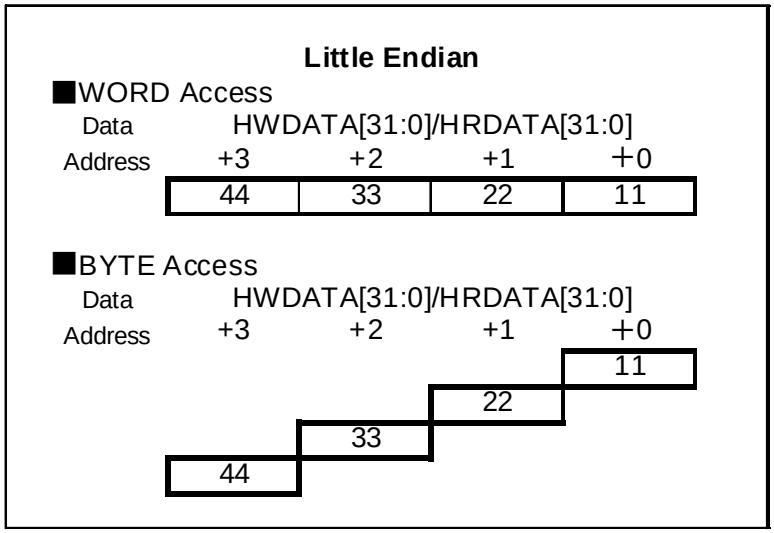


Figure 3.16.3 Image of Endian conversion in AHB Master function

(b) AHB Slave function

Specifications of the AHB Slave function:

- Used for accessing the internal register.
- Little Endian is supported.
- Only single transactions are supported.
- Address width and data width are both 32 bits.
- Transaction sizes in bytes or words are supported.

The image of Endian conversion is as shown below.

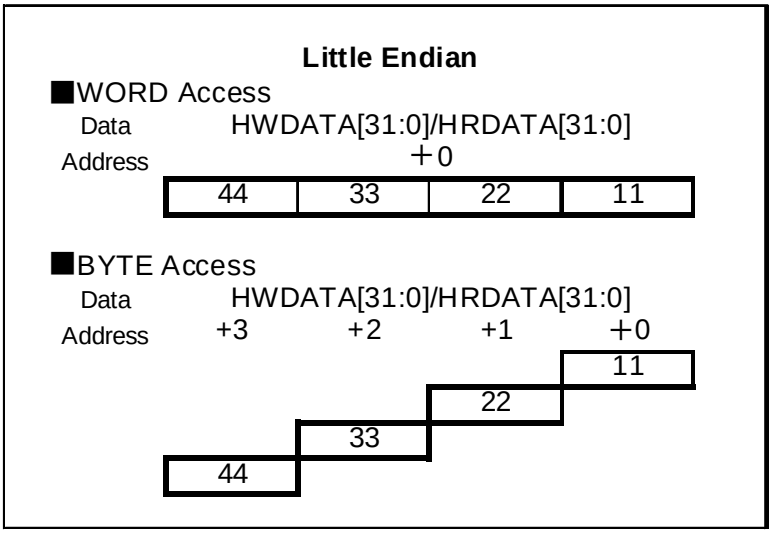


Figure 3.16.4 Image of Endian conversion in AHB Slave function

3.16.2.2 Overall Composition

UDC2AB mainly consists of the AHB Slave function that controls the access to the UDC2AB internal registers and UDC2 registers and the AHB Master function that controls the DMA access to the UDC2 Endpoint I/F.

The AHB Master function has two built-in channels; Master Read Channel (AHB to UDC2) and Master Write Channel (UDC2 to AHB), which enable DMA transfer between the Endpoint I/F of Rx-EP and Tx-EP of UDC2. Each channel has two built-in 8-word buffers (four in total).

3.16.2.3 Clock Domain

CLK_U: 30 MHz (to be supplied by USB 2.0 PHY)

CLK_H: HCLK

3.16.2.4 Terms and Presentation

Assert	: Indicates the signal is active.
Deassert	: Indicates the signal is inactive.
Word	: 32 bits
Byte	: 8 bits
UDC2	: Indicates the USB2.0 device controller to be connected to UDC2AB.
UDC2AB	: This IP: Abbreviation of UDC2-AHB-Bridge
Endpoint	: FIFO held by UDC2 for communication with the USB host. Abbreviated as “EP”.
Rx-EP	: Receive endpoint. For the OUT transfer of USB transfer (USB host to USB device).
Tx-EP	: Transmit endpoint. For the IN transfer of USB transfer (USB device to USB host).
Endpoint I/F	: DMA interface dedicated to the endpoints held by UDC2.
PVCI I/F	: Common interface held by UDC2. Used for accessing the internal registers of UDC2
Master transfer	: Indicates that UDC2AB acquires the bus right to make transfer.
Target device	: Indicates the device (such as memories) to be accessed by UDC2AB with master transfer.
Master Write transfer	: Indicates the transfer with Rx-EP made by UDC2AB.
Master Read transfer	: Indicates the master transfer with Tx-EP made by UDC2AB.
Slave transfer	: Indicates the transfer made by other devices than UDC2AB targeted at UDC2AB.
USB_RESET	: Bus reset sent from the USB host. "Reset Signaling" in the USB specification.
NULL packet	: 0-length data to be transferred on USB.
PHY	: USB 2.0 PHY
Interrupt	: Indicates the INTS [21] output signal. Descriptions like "Assert xx interruption" in this document are based on the assumption that the relevant bit of the Interrupt Enable resistor is enabled. See section 3.16.2.7 “Interrupt Signal (INTS[21])" for more information.

3.16.2.5 Registers

The register map of UDC2AB consists of registers for setting UDC2AB and those for setting UDC2. When the registers for setting UDC2 are accessed, UDC2AB automatically accesses **UDC2 via PVCI I/F.** Each register has the width of 32 bits.

(1) Register map

The register map of UDC2AB is shown below.

Table 3.16.1 UDC2AB/UDC2 register map (1/2)

Base address = 0xF440_0000

	Register Name	Address (base +)	Description
UDC2AB Bridge	UDINTSTS	0x0000	Interrupt Status Register
	UDINTENB	0x0004	Interrupt Enable Register
	UDMWTOUT	0x0008	Master Write Timeout Register
	UDC2STSET	0x000C	UDC2 Setting Register
	UDMSTSET	0x0010	DMAC Setting Register
	DMACRDREQ	0x0014	DMAC Read Request Register
	DMACRDVL	0x0018	DMAC Read Value Register
	UDC2RDREQ	0x001C	UDC2 Read Request Register
	UDC2RDVL	0x0020	UDC2 Read Value Register
	Reserved	0x0024 to 0x0038 *4)	
	ARBTSET	0x003C	Arbiter Setting Register
	UDMWSADR	0x0040	Master Write Start Address Register
	UDMWEADR	0x0044	Master Write End Address Register
	UDMWCADR	0x0048 *1)	Master Write Current Address Register
	UDMWAHBADR	0x004C	Master Write AHB Address Register
	UDMRSADR	0x0050	Master Read Start Address Register
	UDMREADR	0x0054	Master Read End Address Register
	UDMRCADR	0x0058 *1)	Master Read Current Address Register
	UDMRAHBADR	0x005C	Master Read AHB Address Register
	Reserved	0x0060 to 0x007C	
	UDPWCTL	0x0080	Power Detect Control Register
	UDMSTSTS	0x0084	Master Status Register
	UDTOUTCNT	0x0088 *1)	Timeout Count Register
	Reserved	0x008C to 0x1FC *4)	

UDC2AB/UDC2 register map (2/2)

Base address = 0xF440_0000

	Register Name	Address (base +)	Description
UDC2 *2), *3)	UD2ADR	0x0200	UDC2 Address-State Register
	UD2FRM	0x0204	UDC2 Frame Register
	UD2TMD	0x0208	UDC2 USB-Testmode Register
	UD2CMD	0x020C	UDC2 Command Register
	UD2BRQ	0x0210	UDC2 bRequest-bmRequestType Register
	UD2WVL	0x0214	UDC2 wValue Register
	UD2WIDX	0x0218	UDC2 wIndex Register
	UD2WLGTH	0x021C	UDC2 wLength Register
	UD2INT	0x0220	UDC2 INT Register
	UD2INTEP	0x0224	UDC2 INT_EP Register
	UD2INTEPMSK	0x0228	UDC2 INT_EP_MASK Register
	UD2INTRX0	0x022C	UDC2 INT_RX_DATA0 Register
	UD2EP0MSZ	0x0230	UDC2 EP0_MaxPacketSize Register
	UD2EP0STS	0x0234	UDC2 EP0_Status Register
	UD2EP0DSZ	0x0238	UDC2 EP0_Datasize Register
	UD2EP0FIFO	0x023C	UDC2 EP0_FIFO Register
	UD2EP1MSZ	0x0240	UDC2 EP1_MaxPacketSize Register
	UD2EP1STS	0x0244	UDC2 EP1_Status Register
	UD2EP1DSZ	0x0248	UDC2 EP1_Datasize Register
	UD2EP1FIFO	0x024C	UDC2 EP1_FIFO Register
	UD2EP2MSZ	0x0250	UDC2 EP2_MaxPacketSize Register
	UD2EP2STS	0x0254	UDC2 EP2_Status Register
	UD2EP2DSZ	0x0258	UDC2 EP2_Datasize Register
	UD2EP2FIFO	0x025C	UDC2 EP2_FIFO Register
	UD2EP3MSZ	0x0260	UDC2 EP3_MaxPacketSize Register
	UD2EP3STS	0x0264	UDC2 EP3_Status Register
	UD2EP3DSZ	0x0268	UDC2 EP3_Datasize Register
	UD2EP3FIFO	0x026C	UDC2 EP3_FIFO Register
	Reserved	0x0270 to 0x032C	
	UD2INTNAK	0x0330	UDC2 INT_NAK Register
	UD2INTNAKMSK	0x0334	UDC2 INT_NAK_MASK Register
	Reserved	0x0338 to 0x03FC	

*1) Be sure to make Read accesses via DMAC Read Request Register.

*2) Be sure to make Read accesses via UDC2 Read Request Register.

*3)Though the registers of UDC2 are assigned to +0x200 to +0x3FC, no access should be made to the registers of endpoints not supported in the UDC2 to be connected or to any “Reserved” registers.

*4) Those shown as ”Reserved” and in addresses of 0x400 to 0xFFF above are prohibited to aceess. Read/Write is prohibited to those “Reserved” areas.

(2) Register descriptions

The following subsections describe the registers in UDC2AB in detail.
The descriptions of each bit have the following meanings:

(Example)

Address = (0xF440_0000) + (0xxxxx)

Bit	Bit Symbol (Note 1)	Type (Note 2)	Reset Value (Note 3)	Description
[31:30]	–	–	Undefined	Read as undefined. Write as zero.
[29]	mw_rerror_en	R/W	0y0, (-)	
[28]	power_detect_en	R/W	0y0, (-)	
[27:26]	–	–	Undefined	Read as undefined. Write as zero.
[25]	dmac_reg_rd_en	R/W	0y0, (-)	
[24]	udc2_reg_rd_en	R/W	0y0, (-)	

Note 1: Bit symbol

Name of each bit.

Those shown as "–" are reserved bits which cannot be written. 0 will be returned when read.

Note 2: Register properties

RO : Read only. Write is ignored.

WO : Write only. 0 will be returned when read.

R/W : Read/Write

R/W1C : Read/Write 1 Clear. These bits can be both read and written.
When 1 is written, the corresponding bit is cleared. Writing 0 is invalid.

R/W1S : Read/Write 1 Set. These bits can be both read and written.
When 1 is written, the corresponding bit is set. Writing 0 is invalid.

Note 3: Reset value

Initial values for the bit after resetting (1 or 0). Initial values for Hardware Reset and Software Reset (Power Detect Control <pw_resetb>) are identical.

Those bits which will not be reset by Software Reset is shown with (-)

1. UDINTSTS (Interrupt Status register)

This register sets 1 to each corresponding bit when an interrupt source arises. The status can be cleared by writing 1 into bits [29:8]. Bits [7:0] corresponds to the output pins of UDC2 and read-only. It can be cleared by writing 1 into the appropriate bit of INT register in UDC2.

Note: For the operation of interrupt signals, refer to “3.16.2.7 Interrupt Signal (INTS[21])”.

Address = (0xF440_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:30]	–	–	Undefined	Read as undefined. Write as zero.
[29]	int_mw_rerror	R/W1C	0y0	Master Write Endpoint Read error 0y0: Not detected 0y1: Endpoint read error occurred in Master Write
[28:26]	–	–	Undefined	Read as undefined. Write as zero.
[25]	int_dmac_reg_rd	R/W1C	0y0	DMAC register access complete 0y0: Not detected 0y1: Register read completed
[24]	int_udc2_reg_rd	R/W1C	0y0	UDC2 register access complete 0y0: Not detected 0y1: Register read/write completed
[23]	int_mr_ahberr	R/W1C	0y0	Master Read transfer error status 0y0: Not detected 0y1: AHB error occurred
[22]	int_mr_ep_dset	R/W1C	0y0	Master Read endpoint data set status 0y0: FIFO is not writable 0y1: FIFO is writable
[21]	int_mr_end_add	R/W1C	0y0	Master Read transfer end status 0y0: Not detected 0y1: Master Read transfer finished
[20]	int_mw_ahberr	R/W1C	0y0	Master Write transfer error status 0y0: Not detected 0y1: AHB error occurred
[19]	int_mw_timeout	R/W1C	0y0	Master Write transfer time-out status 0y0: Not detected 0y1: Master Write transfer timed out
[18]	int_mw_end_add	R/W1C	0y0	Master Write transfer end status 0y0: Not detected 0y1: Master Write transfer finished
[17]	int_mw_set_add	R/W1C	0y0	Master Write transfer address request status 0y0: Not detected 0y1: Master Write transfer address request
[16:11]	–	–	Undefined	Read as undefined. Write as zero.
[10]	int_usb_reset_end	R/W1C	0y0	USB_RESET END 0y0: UDC2 has not deasserted the usb_reset signal after this bit was cleared. 0y1: Indicates UDC2 has deasserted the usb_reset signal.
[9]	int_usb_reset	R/W1C	0y0	USB_RESET 0y0: UDC2 has not asserted the usb_reset signal after this bit was cleared. 0y1: Indicates UDC2 has asserted the usb_reset signal.
[8]	int_suspend_resume	R/W1C	0y0	Suspend/resume interrupt status 0y0: Status has not changed 0y1: Status has changed

Bit	Bit Symbol	Type	Reset Value	Description
[7]	int_nak	RO	0y0	UDC2_INT_NAK register
[6]	int_ep	RO	0y0	UDC2 INT_EP register
[5]	int_ep0	RO	0y0	UDC2 INT_EP0 register
[4]	int_sof	RO	0y0	UDC2 INT_SOF register
[3]	int_rx_zero	RO	0y0	UDC2 INT_RXDATA0 register
[2]	int_status	RO	0y0	UDC2 INT_STATUS register
[1]	int_status_nak	RO	0y0	UDC2 INT_STATUS_NAK register
[0]	int_setup	RO	0y0	UDC2 INT_STATUS register

[Description]

a. <int_mw_rerror>

Will be set to 1 when the access to the endpoint has started Master Write transfer during the setting of common bus access (bus_sel bit of EPx_Status register is 0).

0y0: Not detected

0y1: Endpoint read error occurred in Master Write

b. <int_dmac_reg_rd>

Will be set to 1 when the register access executed by the setting of DMAC Read Request register is completed and the value read to DMAC Read Value register is set.

0y0: Not detected

0y1: Register read completed

c. <int_udc2_reg_rd>

Will be set to 1 when the UDC2 access executed by the setting of UDC2 Read Request register is completed and the value read to UDC2 Read Value register is set.

Also set to 1 when Write access to the internal register of UDC2 is completed.

0y0: Not detected

0y1: Register read/write completed

d. <int_mr_ahberr>

This status will be set to 1 when the AHB error has occurred during the operation of Master Read transfer.

After this interrupt has occurred, the Master Read transfer block needs to be reset by the mr_reset bit of DMAC Setting register.

0y0: Not detected

0y1: AHB error occurred

e. <int_mr_ep_dset >

Will be set to 1 when the FIFO of EP for UDC2 Tx to be used for Master Read transfer becomes writable (not full).

0y0: FIFO is not writable

0y1: FIFO is writable

f. <int_mr_end_add>

Will be set to 1 when the Master Read transfer has finished.

0y0: Not detected

0y1: Master Read transfer finished

g. <int_mw_ahberr>

This status will be set to 1 when the AHB error has occurred during the operation of Master Write transfer.

After this interrupt has occurred, the Master Write transfer block needs to be reset by the mw_reset bit of DMAC Setting register.

0y0: Not detected

0y1: AHB error occurred

h. <int_mw_timeout>

This status will be set to 1 when time-out has occurred during the operation of Master Write transfer.

0y0: Not detected

0y1: Master Write transfer timed out

i. <int_mw_end_add>

Will be set to 1 when the Master Write transfer has finished.

0y0: Not detected

0y1: Master Write transfer finished

j. <int_mw_set_add>

Will be set to 1 when the data to be sent by Master Write transfer is set to the corresponding EP of Rx while the Master Write transfer is disabled.

0y0: Not detected

0y1: Master Write transfer address request

k. <int_usb_reset_end>

Indicates whether UDC2 has deasserted the usb_reset signal.

The timing in which UDC2 sets the UDC2 register to the initial value after USB_RESET is after the usb_reset signal is deasserted. To detect this timing, use this bit.

The status of the usb_reset signal can be checked using the usb_reset bit of Power Detect Control register.

0y0: UDC2 has not deasserted the usb_reset signal after this bit was cleared.

0y1: Indicates UDC2 has deasserted the usb_reset signal.

l. <int_usb_reset>

Indicates whether UDC2 has asserted the usb_reset signal. The status of the usb_reset signal can be checked using the usb_reset bit of Power Detect Control register.

0y0: UDC2 has not asserted the usb_reset signal after this bit was cleared.

0y1: Indicates UDC2 has asserted the usb_reset signal.

m. <int_suspend_resume>

Asserts 1 each time the suspend_x signal of UDC2 changes. The status can be checked using the suspend_x bit of Power Detect Control register.

0y0: Status has not changed

0y1: Status has changed

n. <int_nak>

The int_nak signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT or INT_NAK register of UDC2.

o. <int_ep>

The int_ep signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT or INT_EP register of UDC2.

p. <int_ep0>

The int_ep0 signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT register of UDC2.

q. <int_sof>

The int_sof signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT register of UDC2.

r. <int_rx_zero>

The int_rx_zero signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT or INT_RX_ZERO register of UDC2.

s. <int_status>

The int_status signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT register of UDC2.

t. <int_status_nak>

The int_status_nak signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT register of UDC2.

u. <int_setup>

The int_setup signal of UDC2 can be directly read. To clear it, clear the corresponding bit of INT register of UDC2.

The connection between the output signals of UDC2 and bits [9] and [7:0] of this register is shown below.

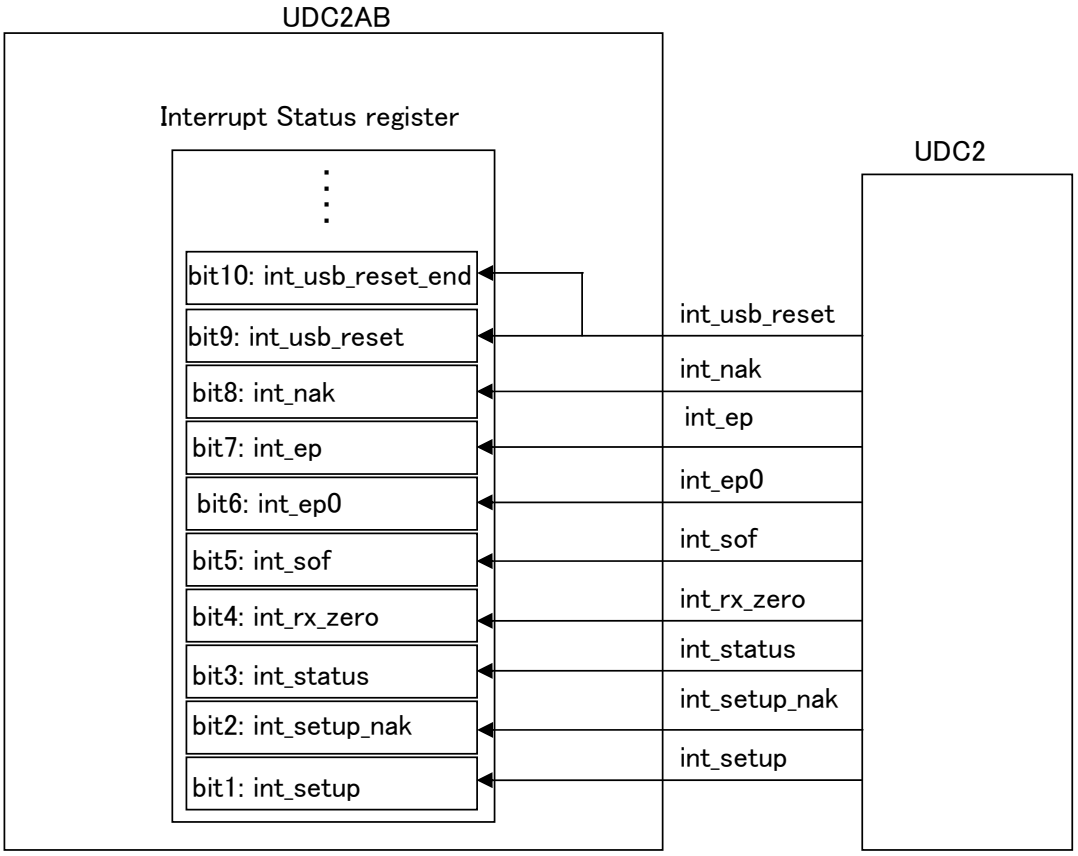


Figure 3.16.5 Connection between the flag output signals and interrupt bits

2. UDINTENB (Interrupt Enable register)

By writing 0 into the corresponding bit of this register, the corresponding interrupt source of the interrupt signal (INTS[21] output signal) can be disabled. Writing 1 will enable the corresponding interrupt source.

Since the corresponding bit of Interrupt Status register will be set regardless of the enabled or disabled status of each bit, an interrupt may occur at the same time as this register was enabled. If such behavior should be avoided, the corresponding bit of Interrupt Status register should be cleared in advance.

The interrupt control register corresponding to bits [7:0] of the Interrupt Status register is bits [15:8] of the INT register of UDC2, not this register. See the section of UDC2.

Note: For the operation of interrupt signals, refer to “3.16.2.7 Interrupt Signal (INTS[21])”.

Address = (0xF440_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:30]	–	–	Undefined	Read as undefined. Write as zero.
[29]	mw_rerror_en	R/W	0y0, (-)	Master Write endpoint read error 0y0: Disable 0y1: Enable
[28:26]	–	–	Undefined	Read as undefined. Write as zero.
[25]	dmac_reg_rd_en	R/W	0y0, (-)	DMAC register read complete 0y0: Disable 0y1: Enable
[24]	udc2_reg_rd_en	R/W	0y0, (-)	UDC2 register read access complete 0y0: Disable 0y1: Enable
[23]	mr_ahberr_en	R/W	0y0, (-)	Master Read transfer error status interrupt enable 0y0: Disable 0y1: Enable
[22]	mr_ep_dset_en	R/W	0y0, (-)	Master Read endpoint data set status interrupt enable 0y0: Disable 0y1: Enable
[21]	mr_end_add_en	R/W	0y0, (-)	Master Read transfer end status interrupt enable 0y0: Disable 0y1: Enable
[20]	mw_ahberr_en	R/W	0y0, (-)	Master Write transfer error status interrupt enable 0y0: Disable 0y1: Enable
[19]	mw_timeout_en	R/W	0y0, (-)	Master Write transfer timeout status interrupt enable 0y0: Disable 0y1: Enable
[18]	mw_end_add_en	R/W	0y0, (-)	Master Write transfer end status interrupt enable 0y0: Disable 0y1: Enable
[17]	mw_set_add_en	R/W	0y0, (-)	Master Write transfer address request status interrupt enable 0y0: Disable 0y1: Enable

Bit	Bit Symbol	Type	Reset Value	Description
[16:11]	–	–	Undefined	Read as undefined. Write as zero.
[10]	usb_reset_end_en	R/W	0y0, (-)	USB_RESET end interrupt enable 0y0: Disable 0y1: Enable
[9]	usb_reset_en	R/W	0y0, (-)	USB_RESET interrupt enable 0y0: Disable 0y1: Enable
[8]	suspend_resume_en	R/W	0y0, (-)	Suspend/resume interrupt enable 0y0: Disable 0y1: Enable
[7:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <mw_rerror_en>
Controls the int_mw_rerror interrupt.
0y0: Disable
0y1: Enable

b. <dmac_reg_rd_en >
Controls the int_dmac_reg_rd interrupt.
0y0: Disable
0y1: Enable

c. <udc2_reg_rd_en >
Controls the int_udc2_reg_rd interrupt.
0y0: Disable
0y1: Enable

d. <mr_ahberr_en >
Controls the int_mr_ahberr interrupt.
0y0: Disable
0y1: Enable

e. <mr_ep_dset_en >
Controls the int_mr_ep_dset interrupt.
0y0: Disable
0y1: Enable

f. <mr_end_add_en>
Controls the int_mr_end_add interrupt.
0y0: Disable
0y1: Enable

g. <mw_ahberr_en>
Controls the int_mw_ahberr interrupt.
0y0: Disable
0y1: Enable

h. <mw_timeout_en>
Controls the int_mw_timeout interrupt.
0y0: Disable
0y1: Enable

i. <mw_end_add_en>
Controls the int_mw_end_add interrupt.
0y0: Disable
0y1: Enable

j. <mw_set_add_en>
Controls the int_mw_set_add interrupt.
0y0: Disable
0y1: Enable

k. <usb_reset_end_en>
Controls the int_usb_reset_end interrupt.
0y0: Disable
0y1: Enable

l. <usb_reset_en>
Controls the int_usb_reset interrupt.
0y0: Disable
0y1: Enable

m. <suspend_resume_en>
Controls the int_suspend_resume interrupt.
0y0: Disable
0y1: Enable

3. UDMWTOUT (Master Write Timeout register)

This register is provided for controlling timeout during the Master Write operation.

Address = (0xF440_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	timeoutset	R/W	0x7FFFFFFF	Master Write timeout timer setting register
[0]	timeout_en	R/W	0y1	Master Write timeout enable register 0y0: Disable 0y1: Enable

[Description]

a. <timeoutset>

The setting should not be changed during the Master Write transfer. Timeout occurs when the number of times CLK_U was set is counted after the data of Master Write (Rx) endpoint is exhausted.

The timeout counter comprises 32 bits of which upper 31 bits can be set by timeoutset [31:1] of this register, while the lowest bit of the counter is set to 1.

As CLK_U is 30 MHz, approximately 33 [ns] to 143 [s] can be set as a timeout value.

While CLK_U stopped (PHY is being suspended and so on), no timeout interrupt will occur as the counter does not work.

b. <timeouten>

Used to enable Master Write timeout. It is set to Enable by default.

The setting should not be changed during the Master Write transfer.

0y0: Disable

0y1: Enable

4. UDC2STSET (UDC2 Setting register)

This register controls transfer operations of UDC2.

Address = (0xF440_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined. Write as zero.
[4]	eopb_enable	R/W	0y1	Master Read EOP enable 0y0: Disable 0y1: Enable
[3:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	tx0	R/W1S	0y0	NULL packet transmission 0y0: No operation 0y1: Transmits NULL packets

[Description]

a. <eopb_enable>

Used to enable Master Read EOP. It is set to Enable by default. The setting should not be changed during the Master Read transfer.

If this bit is 0, the final data transfer to UDC2 will not take place when the last word is 1 byte. If the last word is 2 bytes, the final data transfer to UDC2 will take place when epx_w_eop = 0.

If this bit is 1, the final data transfer to UDC2 will take place when epx_w_eop = 1 regardless of byte number of the last word.

Note: See section 3.16.2.9 “(1) Master Read transfer" for more information.

0y0: Master Read EOP disabled

0y1: Master Read EOP enabled

b. <tx0>

Used to transmit NULL packets at an endpoint connected to the Master Read operation side. Only valid when the mrepempty bit of Master Status register is 1, otherwise this bit is ignored. It will be automatically cleared to 0 after writing. Setting 1 to this bit will assert the epx_tx0data signal of the UDC2 Endpoint-I/F and the value of 1 is retained during the transmission of NULL packets. After this bit is set, next data setting for Tx-EP should not be made until it is cleared.

0y0: No operation

0y1: Transmits NULL packets

5. UDMSTSET (DMAC Setting register)

This register controls transfers of the built-in DMAC.

Address = (0xF440_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:9]	–	–	Undefined	Read as undefined. Write as zero.
[8]	m_burst_type	R/W	0y0, (-)	Master burst type 0y0: INCR8 (HBURST=5h) 0y1: INCR (HBURST=1h)
[7]	–	–	Undefined	Read as undefined. Write as zero.
[6]	mr_reset	R/W1S	0y0	Master Read reset 0y0: No operation 0y1: Reset
[5]	mr_abort	WO	0y0	Master Read abort 0y0: No operation 0y1: Abort
[4]	mr_enable	R/W1S	0y0	Master Read enable 0y0: Disable 0y1: Enable
[3]	–	–	Undefined	Read as undefined. Write as zero.
[2]	mw_reset	R/W1S	0y0	Master Write reset 0y0: No operation 0y1: Reset
[1]	mw_abort	WO	0y0	Master Write abort 0y0: No operation 0y1: Abort
[0]	mw_enable	R/W1S	0y0	Master Write enable 0y0: Disable 0y1: Enable

[Description]

a. <m_burst_type>

Selects the type of HBURST[2:0] when making a burst transfer in Master Write/Read transfers. The type of burst transfer made by UDC2AB is INCR8 (burst of 8 beat increment type). Accordingly, 0 (initial value) should be set in normal situation. However, in case INCR can only be used as the type of burst transfer based on the AHB specification of the system, set 1 to this bit. In that case, UDC2AB will make INCR transfer of 8 beat. Please note the number of beat in burst transfers cannot be changed.

Setting of this bit should be made in the initial setting of UDC2AB. The setting should not be changed after the Master Write/Read transfers started.

Note: UDC2AB does not make burst transfers only in Master Write/Read transfers. It combines burst transfers and single transfers. This bit affects the execution of burst transfers only.

0y0: INCR8

0y1: INCR

b. <mr_reset>

Initializes the Master Read transfer block of UDC2AB. However, as the FIFOs of endpoints are not initialized, you need to access the Command register of UDC2 to initialize the corresponding endpoint separately from this reset.

This reset should be used after stopping the Master operation.

This bit will be automatically cleared to 0 after being set to 1. Subsequent Master Read transfers should not be made until it is cleared.

0y0: No operation

0y1: Reset

c. <mr_abort>

Controls Master Read transfers. Master Read operations can be stopped by setting 1 to this bit.

When aborted during transfers, transfer of buffers for Master Read to UDC2 is interrupted and the mr_enable bit is cleared, stopping the Master Read transfer.

Aborting completes when the mr_enable bit is disabled to 0 after setting this bit to 1.

0y0: No operation

0y1: Abort

d. <mr_enable>

Controls Master Read transfers. Enabling should be made when setting the transfer address is completed. It will be automatically disabled as the master transfer finishes. Since Master Read operations cannot be disabled with this register, use the mr_abort bit if the Master Read transfer should be stopped.

0y0: Disable

0y1: Enable

e. <mw_reset>

Initializes the Master Write transfer block. However, as the FIFOs of endpoints are not initialized, you need to access the Command register of UDC2 to initialize the corresponding endpoint separately from this reset.

This reset should be used after stopping the Master operation.

This bit will be automatically cleared to 0 after being set to 1. Subsequent Master Write transfers should not be made until it is cleared.

0y0: No operation

0y1: Reset

f. <mw_abort>

Controls Master Write transfers. Master Write operations can be stopped by setting 1 to this bit.

When aborted during transfers, transfer of buffers for Master Write from UDC2 is interrupted and the mw_enable bit is cleared, stopping the Master Write transfer. Aborting completes when the mw_enable bit is disabled to 0 after setting this bit to 1.

0y0: No operation

0y1: Abort

g. <mw_enable>

Controls Master Write transfers. Enabling should be made when setting the transfer address is completed. It will be automatically disabled as the master transfer finishes. Since Master Write operations cannot be disabled with this register, use the mw_abort bit if the Master Write transfer should be stopped.

0y0: Disable

0y1: Enable

6. DMACRDREQ (DMAC Read Request register)

This register is used to issue read requests for reading the following registers:

- Master Read Current Address register
- Timeout Count register

The read value will be saved in the DMAC Read Value register.

Note: As accesses to this register become unavailable when the clock (= CLK_U) supply from PHY is stopped with UDC2 suspended, no access should be made. If this register is accessed when the phy_suspend bit of Power Detect Control register is set to 1, an AHB error will be returned.

Address = (0xF440_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31]	dmardreq	R/W1S	0y0	Register read request & busy 0y0: No operation 0y1: Issue read request
[30]	dmardclr	R/W1S	0y0	Read request clear 0y0: No operation 0y1: Issue forced clearing
[29:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:2]	dmardadr	R/W	0y000000	Read request register address (upper 6 bits) select 0x48: Read the Master Write Current Address register 0x58: Read the Master Read Current Address register 0x88: Read the Timeout Count register
[1:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <dmardreq>

The bit for requesting read access to the DMAC registers. Setting this bit to 1 will make a read access to the address specified by dmardadr. When the read access is complete and the read value is stored in the DMAC Read Value register, this bit will be automatically cleared and the int_dmac_reg_rd bit of Interrupt Status register will be set to 1.

0y0: No operation

0y1: Issue read request

b. <dmardclr>

The bit for forcibly clearing the register read access request associated with DMAC. Setting this bit to 1 will forcibly stop the register read access request by dmardreq and the value of dmardreq will be cleared to 0. After the forced clearing completes, this bit will be automatically cleared.

0y0: No operation

0y1: Issue forced clearing

c. <dmardadr>

Sets the address of the register (upper 6 bits) to be read. It should be set in combination with the dmardreq bit mentioned above.

Any one of the following addresses should be set:

0x48: Read the Master Write Current Address register

0x58: Read the Master Read Current Address register

0x88: Read the Timeout Count register

7. DMACRDVL (DMAC Read Value register)

The register in which the values read via DMAC Read Request register are stored.

(Relevant registers)

- Master Write Current Address register
- Master Read Current Address register
- Timeout Count register

Address = (0xF440_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	dmardata	RO	0x00000000	Register read data

[Description]

a. <dmardata>

This register stores the data requested by DMAC Read Request register. This register should not be accessed when the dmardreq bit of DMAC Read Request register is set to 1.

8. UDC2RDREQ (UDC2 Read Request register)

The register for issuing read requests when reading UDC2 registers. The read value will be saved in the UDC2 Read Value register.

Address = (0xF440_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31]	udc2rdreq	R/W1S	0y0	Register read request & busy 0y0: No operation 0y1: Issue read request
[30]	udc2rdclr	R/W1S	0y0	Read request clear 0y0: No operation 0y1: Issue forced clearing
[29:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:2]	udc2rdadr	R/W	0x00	The address of the UDC2 register that issues the read request
[1:0]	–	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <udc2rdreq>

The bit for requesting read access to the UDC2 registers. Setting this bit to 1 will make a read access to the address set in the udc2rdadr bit. When the read access is complete and the read value is set to UDC2 Read Value register, this bit will be automatically cleared and the UDINTSTS<int_udc2_reg_rd> bit of Interrupt Status register will be set to 1.

During a write access to UDC2 registers, it works as a status bit which indicates the access being made to display the value of 1. Subsequent accesses to UDC2 registers should not be made while this bit is set to 1.

0y0: No operation

0y1: Issue read request

b. <udc2rdclr>

The bit for forcibly clearing the read/write access request of UDC2 registers. Setting this bit to 1 will forcibly stop the register read request/UDC2 write access by udc2rdreq and the value of udc2rdreq will be 0. After the forced clearing completes, this bit will be automatically cleared to 0. When interrupted, the read and write values during the access will not be secured.

0y0: No operation

0y1: Issue forced clearing

c. <udc2rdadr>

Sets the address of the UDC2 register (upper 8 bits) to be read. Regarding registor address, please refer to “Table 3.16.2 Register map”.Between 0x0200 to 0x0334 that is the offset address of this register map corresponds.It should be set in combination with the udc2rdreq bit mentioned above.

9. UDC2RDVL (UDC2 Read Value register)

The register in which the values read via UDC2 Read Request register are stored.

Address = (0xF440_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:0]	udc2rdata	RO	0x0000	Register read data

[Description]

a. <udc2rdata>

This register stores the data requested by UDC2 Read Request register. This register should not be accessed when the udc2rdreq bit of UDC2 Read Request register is set to 1.

10. ARBTSET (Arbiter Setting register)

The register for setting the priority when the internal arbiter accesses AHB.

Setting of this register should be changed after stopping the Master operation.

Please be sure to set the arbitration method with the following procedures (You need to make an access three times in total.):

(1) Write 0 into the abt_en bit to disable the arbitration circuit.

(2) Make settings for the abtmod and abtpri_* bits.

The abtmod and abtpri_* bits cannot be set unless 0 is written into the abt_en bit in (1). Values of the register for setting the priority should not be overlapped regardless of the value of the abtmod bit.

(3) Write 1 into the abt_en bit with the abtmod and abtpri_* bits set in (2) retained to enable the arbitration circuit.

Address = (0xF440_0000) + (0x003C)

Bit	Bit Symbol	Type	Reset Value	Description
[31]	abt_en	R/W	0y1	Arbiter enable 0y0: Disable (DMA access not allowed) 0y1: Enable
[30:29]	–	–	Undefined	Read as undefined. Write as zero.
[28]	abtmod	R/W	0y0	Arbiter mode 0y0: Round-robin 0y1: Fixed priority
[27:14]	–	–	Undefined	Read as undefined. Write as zero.
[13:12]	abtpri_w1	R/W	0y11	Master Write 1 priority 0y00 to 0y11
[11:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:8]	abtpri_w0	R/W	0y10	Master Write 0 priority 0y00 to 0y11
[7:6]	–	–	Undefined	Read as undefined. Write as zero.
[5:4]	abtpri_r1	R/W	0y01	Master Read 1 priority 0y00 to 0y11
[3:2]	–	–	Undefined	Read as undefined. Write as zero.
[1:0]	abtpri_r0	R/W	0y00	Master Read 0 priority 0y00 to 0y11

[Description]

a. <abt_en>

Enables the arbiter operation when making an access between DMAC and AHB.

0 should be set to this bit when setting the abtmod and abtpri_* bits of this register. Please note that 1 cannot be set to this bit in case values set for abtpri_* overlap.

Be sure to set this bit to 1 before starting a DMA access.

0y0: Disable (DMA access not allowed)

0y1: Enable

b. <abtmod>

Sets the mode of arbiter. Write access is only available when the abt_en bit is set to 0.

If 0 is set to this bit, access rights to the AHB bus will be given in a round-robin fashion regardless of the values set to each abtpri_* bit. If 1 is set to this bit, access rights to the AHB bus will be given in accordance with the access priority based on the values set to each abtpri_* bit.

0y0: Round-robin

0y1: Fixed priority

c. <abtpri_w1>

Set the priority of DMA accesses for Master Write 1 when the fixed priority mode is selected. Write access is only available when the abt_en bit is set to 0.

Priority ranges from [0] (highest) to [3] (lowest).

d. <abtpri_w0>

Set the priority of DMA accesses for Master Write 0 when the fixed priority mode is selected. Write access is only available when the abt_en bit is set to 0.

Priority ranges from [0] (highest) to [3] (lowest).

e. <abtpri_r1>

Set the priority of DMA accesses for Master Read 1 when the fixed priority mode is selected. Write access is only available when the abt_en bit is set to 0.

Priority ranges from [0] (highest) to [3] (lowest).

f. <abtpri_r0>

Set the priority of DMA accesses for Master Read 0 when the fixed priority mode is selected. Write access is only available when the abt_en bit is set to 0.

Priority ranges from [0] (highest) to [3] (lowest).

- Note:

Be sure to set different priority values for the abtpri_w1, abtpri_w0, abtpri_r1, and abtpri_r0 bits. If the same priority values are set, you will not be able to set 1 to abt_en.

<Relationship of DMAC and the priority area of the Arbiter Setting register>

Current UDC2AB specification supports one DMAC for Master Write (DMAC_W0) and one DMAC for Master Read (DMAC_R0). The second DMAC for Master Write (DMAC_W1) and the second DMAC for Master Read (DMAC_R1) are not supported. Accordingly, setting priority for DMAC_W1 and DMAC_R1 has virtually no meaning, but you should be sure to set different priority values for the abtpri_w1, abtpri_w0, abtpri_r1, and abtpri_r0 bits as mentioned above. There will be no problem to set values for the corresponding register areas of an unpackaged DMAC. The priority areas of Arbiter Setting register correspond with DMAC as shown below.

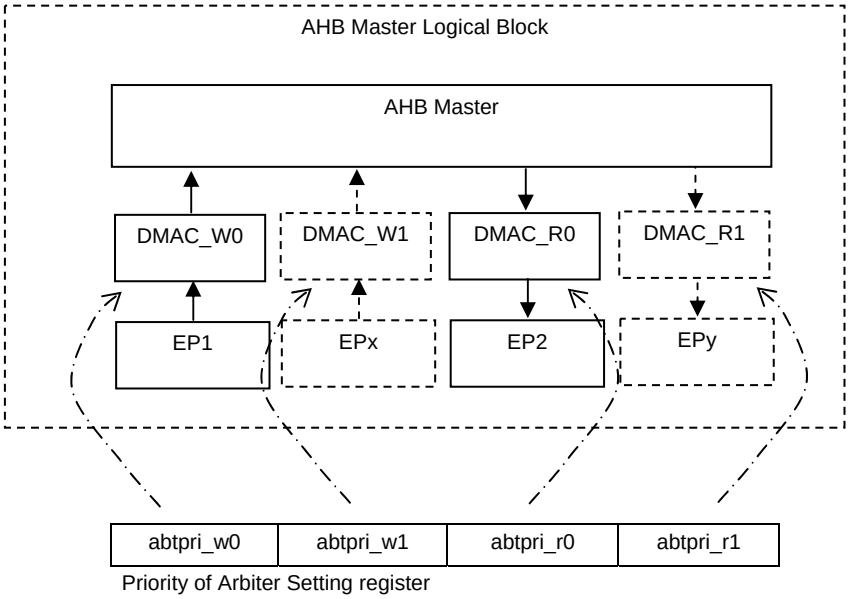


Figure 3.16.6 Relationship between DMAC and priority areas

11. UDMWSADR (Master Write Start Address register)

Sets the start address of Master Write transfer (UDC2 to AHB).

Address = (0xF440_0000) + (0x0040)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mwsadr	R/W	0xFFFFFFFF	Master Write start address

[Description]

a. <mwsadr>

Set the start address of Master Write transfer. However, as this master operation only supports address increments, values lower than the Master Write End Address register should be set.

12. UDMWEADR (Master Write End Address register)

Sets the end address of Master Write transfer (UDC2 to AHB).

Address = (0xF440_0000) + (0x0044)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mweadr	R/W	0xFFFFFFFF	Master Write end address

[Description]

a. <mweadr>

Set the end address of Master Write transfer. However, as this master only supports address increments, values above the Master Write Start Address register should be set.

13. UDMWCADR (Master Write Current Address register)

Displays the address to which transfers from endpoints to the Master Write buffers have been currently completed in Master Write transfers (UDC2 to AHB).

This register cannot be read by directly specifying the address. In order to read it, set a value to the DMAC Read Request register and then read the value from the DMAC Read Value register.

Address =(0xF440_0000)+ (0x0048)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mwcadr	RO	0x00000000	Master Write current address

[Description]

a. <mwcadr>

Displays the addresses to which transfers from endpoints to the Master Write buffers have been currently completed in Master Write transfers. This can be used in case a timeout interrupt has occurred or an error occurred during the transfer process.

This address is incremented at the point when the data is set from the endpoint to the Master Write buffer, while the data will reside inside the target device or the Master Write buffer during the Master Write transfer process until the displayed address.

14. UDMWAHBADR (Master Write AHB Address register)

Displays the address where the transfer to the target device has completed in Master Write transfer (UDC2 to AHB).

In some DMA transfers, accesses are made on a byte basis depending on the conditions. Please note that the address to be saved is the word border even when accessing by byte.

Address = (0xF440_0000) + (0x004C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mwahbadr	RO	0xFFFFFFFF	Master Write AHB address

[Description]

a. <mwahbadr>

Displays the address where the transfer to the target device has completed in Master Write transfer. This can be used in case a timeout interrupt has occurred or an error occurred during the transfer process. This address is incremented at the point when the data is set to the target device, while the data will reside inside the target device or during the Master Write transfer process until the displayed address.

15. UDMRSADR (Master Read Start Address register)

Sets the start address of Master Read transfer (AHB to UDC2).

Address = (0xF440_0000) + (0x0050)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mrsadr	R/W	0xFFFFFFFF	Master Read start address

[Description]

a. <mrsadr>

Set the start address of Master Read transfer. However, as this master only supports address increments, values lower than the Master Read End Address register should be set.

16. UDMREADR (Master Read End Address register)

Sets the end address of Master Read transfer (AHB to UDC2).

Address = (0xF440_0000) + (0x0054)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	Mreadr	R/W	0xFFFFFFFF	Master Read end address

[Description]

a. <Mreadr>

Set the end address of Master Read transfer. However, as this master only supports address increments, values above the Master Read Start Address register should be set.

17. UDMRCADR (Master Read Current Address register)

Displays the address where the transfer from the target device to the endpoint has completed in Master Read transfer (AHB to UDC2).

This register cannot be read by directly specifying the address. In order to read it, set a value to the DMAC Read Request register and then read the value from the DMAC Read Value register.

Address =(0xF440_0000)+ (0x0058)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mrcadr	RO	0x00000000	Master Read current address

[Description]

a. <mrcadr>

Displays the address to which transfers from the target device to the endpoint have been currently completed in Master Read transfers.

This address is incremented at the point when the data is set from the Master Read buffer to the endpoint, while the data will reside inside the FIFO for the endpoint during the Master Read transfer process until the displayed address.

18. UDMRAHBADR (Master Read AHB Address register)

Displays the address where the transfer from the target device to UDC2AB has completed in Master Read transfer (AHB to UDC2).

In some DMA transfers, accesses are made on a byte basis depending on the conditions. The address to be saved is the word border when accessing by byte.

Address = (0xF440_0000) + (0x005C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	mrahbadr	RO	0xFFFFFFFF	Master read AHB address

[Description]

a. <mrahbadr>

Displays the address where the transfer from the target device to UDC2AB has completed in Master Read transfer. This address is incremented at the point when the data is set from the target device, while the data will reside inside the buffer or the FIFO for the endpoint during the Master Read transfer process until the displayed address.

19. UDPWCTL (Power Detect Control register)

Controls UDC2AB when reset/suspended.

Address = (0xF440_0000) + (0x0080)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	Undefined	–	Read as undefined. Write as zero.
[7]	wakeup_en	R/W	0y0, (-)	Wakeup enable 0y0: Do not assert the WAKEUP_X signal 0y1: Assert the WAKEUP_X signal (Note 1)
[6]	phy_remote_wkup	R/W1S	0y0, (-)	Remote wakeup 0y0: No operation 0y1: Wakeup
[5]	phy_resetb	R/W	0y1, (-)	PHY reset 0y0: Reset asserted 0y1: Reset deasserted
[4]	suspend_x	RO	0y1	Suspend detection 0y0: Suspended (suspend_x = 0) 0y1: Not suspended (suspend_x = 1)
[3]	phy_suspend	R/W	0y0, (-)	PHY suspend 0y0: Not suspended 0y1: Suspended
[2]	pw_detect	RO	0y0, (-)	USB bus power detect 0y0: USB bus disconnected (VBUSPOWER = 0) 0y1: USB bus connected (VBUSPOWER = 1) (Note 2)
[1]	pw_resetb	R/W	0y1, (-)	Power reset 0y0: Reset asserted 0y1: Reset deasserted
[0]	usb_reset	RO	0y0	USB_RESET 0y0: usb_reset = 0 0y1: usb_reset = 1

Note 1: While UDC2AB originally has the function to assert the Wakeup signal, it is not supported for this LSI.

Note 2: While UDC2AB originally has the function to assert the int_powerdetect interrupt when VBUS is detected, it is not supported for this LSI. Power Detect Control<pw_detect> always indicates 0.

[Description]

a. <wakeup_en>

Set this bit to '1' if you want the system (AHB end) to sleep to stop CLK_H when the USB is suspended. If this bit is set to 1, the WAKEUP_X signal will be asserted to 0 asynchronously when the suspended status is cancelled (suspend_x = 1) or the system is disconnected (VBUSPOWER = 0), making it available for resuming the system.

See also section 3.16.2.13 "(4) Signal operations when suspended and resumed (disconnected)" for more information on using this bit.

0y0: Do not assert the WAKEUP_X signal

0y1: Assert the WAKEUP_X signal

Note: While UDC2AB originally has the function to assert the Wakeup signal, it is not supported for this LSI.

b. <phy_remote_wkup>

This bit is used to perform the remote wakeup function of USB. Setting this bit to 1 makes it possible to assert the udc2_wakeup output signal (wakeup input pin of UDC2) to 1. However, since setting this bit to 1 while no suspension is detected by UDC2 (when suspend_x = 1) will be ignored (not to be set to 1), be sure to set it only when suspension is detected. It will be automatically cleared to 0 when resuming the USB is completed (when suspend_x is deasserted).

See also section 3.16.2.13 "(4) Signal operations when suspended and resumed (disconnected)" for more information on using this bit.

0y0: No operation

0y1: Wakeup

c. <phy_resetb>

Setting this bit to 0 will make the PHYRESET output signal asserted to 1. The PHYRESET signal can be used to reset PHY. Since this bit will not be automatically released, be sure to clear it to 1 after the specified reset time of PHY.

0y0: Reset asserted

0y1: Reset deasserted

d. <suspend_x>

Detects the suspend signal (a value of the suspend_x signal from UDC2 synchronized).

0y0: Suspended (suspend_x = 0)

0y1: Unsuspended (suspend_x = 1)

e. <phy_suspend>

Setting this bit to 1 will make the PHYSUSPEND output signal asserted to 0 (CLK_H synchronization). It can be used as a pin for suspending PHY.

Setting this bit to 1 makes the UDC2 register and DMAC Read Request register not accessible.

It will be automatically cleared to 0 when resumed (when suspend_x of UDC2 is deasserted).

See also section 3.16.2.13 "(4) Signal operations when suspended and resumed (disconnected)" for more information on using this bit.

0y0: Not suspended

0y1: Suspended

f. <pw_detect>

Indicates the status of the VBUSPOWER input pin.

0y0: USB bus disconnected (VBUSPOWER = 0)

0y1: USB bus connected (VBUSPOWER = 1)

Note: While UDC2AB originally has the function to assert the int_powerdetect interrupt when VBUS is detected, it is not supported for this LSI. Power Detect Control<pw_detect> always indicates 0.

g. <pw_resetb>

Software reset for UDC2AB. (See section 3.16.2.6 “Reset” for details.).

Setting this bit to 0 will make the PW_RESETB output pin asserted to 0.

Resetting should be made while the master operation is stopped.

Since this bit will not be automatically released, be sure to clear it.

0y0: Reset asserted

0y1: Reset deasserted

h. <usb_reset>

The value of the usb_reset signal from UDC2 synchronized.

0y0: usb_reset = 0

0y1: usb_reset = 1

20. UDMSTSTS (Master Status register)

This is a status register of UDC2AB.

Address = (0xF440_0000) + (0x0084)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined.
[4]	mrepempty	RO	0y0, (-)	Master Read endpoint empty 0y0: Indicates the endpoint contains some data. 0y1: Indicates the endpoint is empty.
[3]	mrbfemp	RO	0y1	Master Read buffer empty 0y0: Indicates the buffer for the Master Read DMA contains some data. 0y1: Indicates the buffer for the Master Read DMA is empty.
[2]	mwbfemp	RO	0y1	Master Write buffer empty 0y0: Indicates the buffer for the Master Write DMA contains some data. 0y1: Indicates the buffer for the Master Write DMA is empty.
[1]	mrepdset	RO	0y0, (-)	Master Read endpoint DATASET 0y0: Data can be transferred into the endpoint. 0y1: There is no space to transfer data in the endpoint.
[0]	mwepdset	RO	0y0, (-)	Master Write endpoint DATASET 0y0: No data exists in the endpoint. 0y1: There is some data to be read in the endpoint.

[Description]

a. <mrepempty>

This is a register that indicates the endpoint for UDC2Rx is empty. Ensure that this bit is set to 1 when sending a NULL packet using the tx0 bit of UDC2 Setting register. (This bit is the eptx_empty input signal with CLK_H synchronization.)

0y0: Indicates the endpoint contains some data.

0y1: Indicates the endpoint is empty.

b. <mrbfemp>

Indicates whether or not the buffer for the Master Read DMA in UDC2AB is empty.

0y0: Indicates the buffer for the Master Read DMA contains some data.

0y1: Indicates the buffer for the Master Read DMA is empty.

c. <mwbfemp>

Indicates whether or not the buffer for the Master Write DMA in UDC2AB is empty.

0y0: Indicates the buffer for the Master Write DMA contains some data.

0y1: Indicates the buffer for the Master Write DMA is empty.

d. <mrepdset>

This bit will be set to 1 when the data to be transmitted is set to the Tx-EP of UDC2 by Master Read DMA transfer, making no room to write in the endpoint. It will turn to 0 when the data is transferred from UDC2 by the IN-Token from the host. While this bit is set to 0, DMA transfers to the endpoint can be made. (This bit is the eptx_dataset input signal with CLK_H synchronization.)

0y0: Data can be transferred into the endpoint.

0y1: There is no space to transfer data in the endpoint.

e. <mwepdset>

This bit will be set to 1 when the data received is set to the Rx-EP of UDC2. It will turn to 0 when the entire data was read by the DMA for Master Write. (This bit is the eprx_dataset input signal with CLK_H synchronization.)

0y0: No data exists in the endpoint.

0y1: There is some data to be read in the endpoint.

21. UDTOUTCNT (Timerout Count register)

This is a register to read the timeout count value. (for debugging)

This register cannot be read by directly specifying the address. In order to read it, set a value to the DMAC Read Request register and then read the value from the DMAC Read Value register.

Address = (0xF440_0000) + (0x0088)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	tmoutcnt	RO	0x00000000	Timeout count

[Description]

a. <tmoutcnt>

This is used for debugging. Values of the timer can be read when the timeout_en bit of Master Write Timeout register is enabled. It will be decremented each time CLK_U is counted after the endpoint for Master Write (Rx-EP) becomes empty.

22. UDC2 (UDC2 register) (0x0200 to 0x03FC)

The internal register of UDC2 (16 bits) can be accessed by making an access to the (0xF440_0000) + 0x200-0x3FC. AHB data bus of UDC2AB has 32 bits, of which bits 15-0 correspond with the UDC2 data bus. Bits 31-16 are reserved bits and read-only (read value: 0). Make a WORD (32-bit) access for both write and read. (However, a BYTE (8-bit) access may be made for Write accesses to the EPx_FIFO register. Details will be discussed later.)

It will take some time to complete an access for both write and read (accessing period to UDC2). Be sure to begin subsequent accesses after the previous UDC2 register access is completed, using the int_udc2_reg_rd interrupt. (You can also use the udc2rdreq bit of UDC2 Read Request register to confirm the access status when reading.)

- Write access

  When making a write access to the UDC2 register, write it directly in the relevant address.

- Read access

  When making a read access to the UDC2 register, use UDC2 Read Request and UDC2 Read Value registers.

  First, you set the address to access to the UDC2 Read Request register and then read the data from the UDC2 Read Value register for reading. You cannot read the data directly from the address shown in the address map.

- EPx_FIFO register

  When making a write access to the EPx_FIFO register, a lower 1-byte access may be required in UDC2 PVCI I/F. In such a case, make a BYTE access to the lower 1 byte for UDC2AB.

  If a lower 1-byte access is required when making a read access, make an access via UDC2 Read Request register as usual and read the data from UDC2 Read Value register. In that case, the access to UDC2 Read Value register can be either by WORD or BYTE.

- Reserved registers in UDC2

  Do not make any access to registers of endpoints not supported by UDC2 to be connected and to "Reserved" registers. (In case those registers are accessed, the access from UDC2AB to UDC2 itself will take place. It will be a Dummy write to UDC2 in case of write accesses. In case of read accesses, the read data from UDC2 (udc2_rdata) will be an indefinite value and the indefinite value will be set to the UDC2 Read Value register.)

- Accesses when UDC2 is suspended

  When UDC2 is in the suspended status, register accesses to UDC2 become unavailable if the clock (= CLK_U) supply from PHY is stopped. Make no register accesses to UDC2 in such cases. If the UDC2 register is accessed when the phy_suspend bit of Power Detect Control register is set to 1, an AHB error will be returned.

Address = (0xF440_0000) + (0x0200-0x03FC)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	udc2_data	–	–	UDC2 data register Refer to the section of UDC2 on the data values.

Access flow diagram for UDC2 register is shown below.

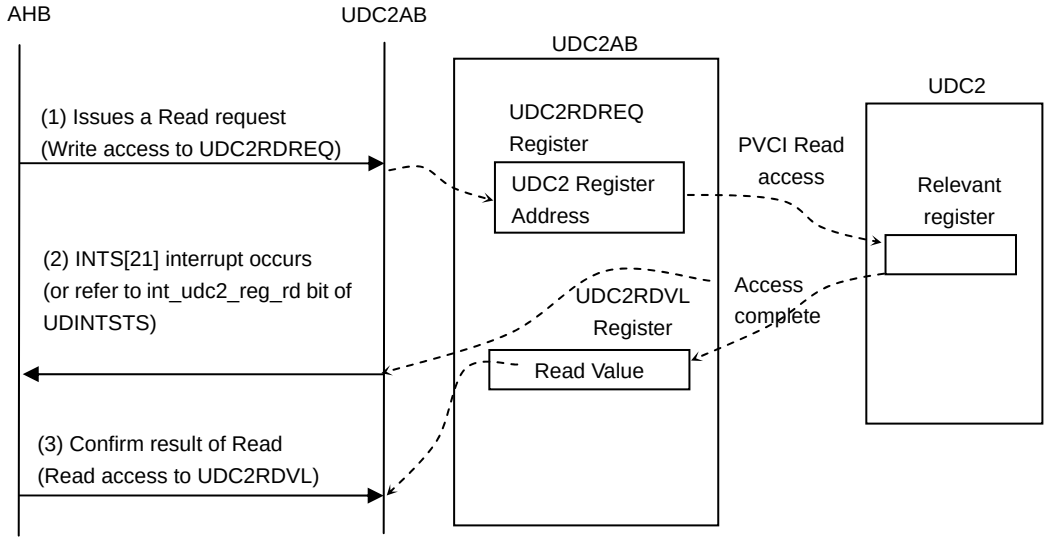


Figure 3.16.7 Read access flow for UDC2 register

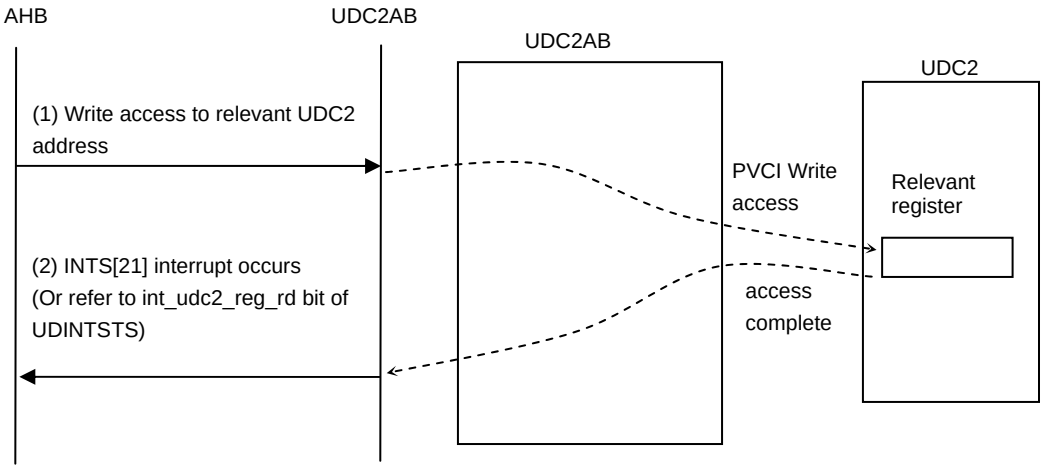


Figure 3.16.8 Write access flow for UDC2 register

3.16.2.6 Reset

UDC2AB supports software reset by the Power Detect Control<pw_resetb>.

It also supports master channel reset (mr_reset/mw_reset bit of DMAC Setting register) for DMAC master transfers.

- Software reset (Power Detect Control<pw_resetb>)

  Some bits of each register are initialized by hardware reset but not initialized by software reset with the values retained. As details are provided in the descriptions of each register, refer to section 3.16.2.5 “Registers”.

  When the USB bus power is detected, make software reset as initialization is needed.

- Master channel reset (mr_reset/mw_reset bit of DMAC Setting register)

  While the mw_reset bit is provided for the Master Write transfer block and the mr_reset bit for the Master Read transfer block, only the relevant master blocks are initialized and the UDC2AB register will not be initialized. For more information on using each reset, see section 3.16.2.5 “(2) 5. UDMSTSET (DMAC Setting register).”

3.16.2.7 Interrupt Signal (INTS[21])

The interrupt output signal of UDC2AB (INTS[21]) consists of interrupts generated by UDC2 and interrupts generated from other sources. Once the interrupt condition is met, UDC2AB sets the corresponding bit of its internal Interrupt Status register. When that bit is set, INTS[21] will be asserted if the relevant bit of Interrupt Enable register has been set to "Enable."

When the relevant bit of Interrupt Enable register has been set to "Disable," 1 will be set to the corresponding bit of Interrupt Status register while INTS[21] will not be asserted. When the relevant bit of Interrupt Enable register is set to "Enable" with Interrupt Status register set, INTS[21] will be asserted immediately after the setting is made.

Initial values for Interrupt Enable register are all 0 (Disable).

The image of the aforementioned description is shown in the figure below.

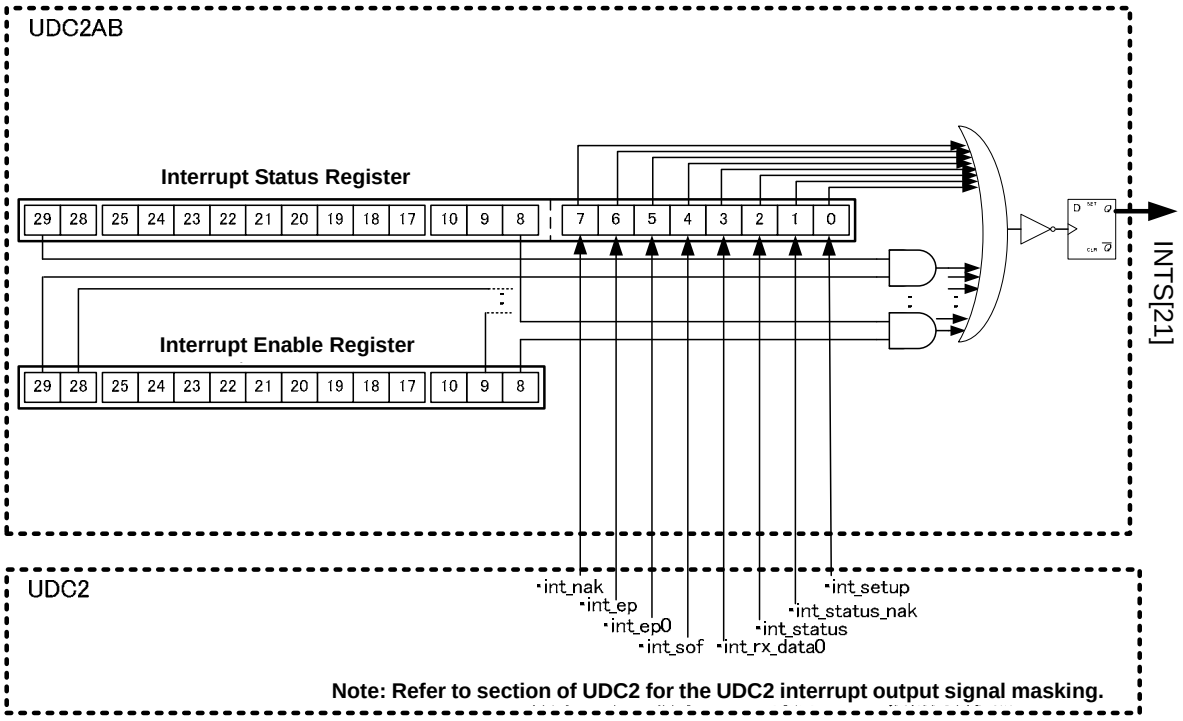


Figure 3.16.9 Relationship of INTS[21] and registers

3.16.2.8 Overall Operation Sequence

The overall operation sequence of UDC2AB is as follows:

1. Hardware reset

2. Set the interrupt signal

   In the Interrupt Enable register, set the required bit of the interrupt source to "Enable." See section 3.16.2.7 "Interrupt Signal (INTS[21])" for more information.

3. Detect the USB bus power supply (connect) and initialize

   See section 3.16.2.11 "USB Bus Power Detecting Sequence" for more information.

Note: While UDC2AB originally has the function to assert the int_powerdetect interrupt when VBUS is detected, it is not supported for this LSI. Power Detect Control<pw_detect> always indicates 0.

4. USB enumeration response

   See section 3.16.3.4 "USB Device Response" in the section of UDC2.

5a. Master Read transfer

   Make a Master Read transfer corresponding to the receiving request from the USB host. See section 3.16.2.9 "(1) Master Read transfer" for more information.

5b. Master Write transfer

   Make a Master Write transfer corresponding to the sending request from the USB host. See section 3.16.2.9 "(4) Master Write transfer" for more information.

6. USB bus power supply disconnection

   It may be possible that USB bus power supply is disconnected at any timing.

   See section 3.16.2.11 "USB Bus Power Detecting Sequence" for more information.

Note: While UDC2AB originally has the function to assert the int_powerdetect interrupt when VBUS is detected, it is not supported for this LSI. Power Detect Control<pw_detect> always indicates 0.

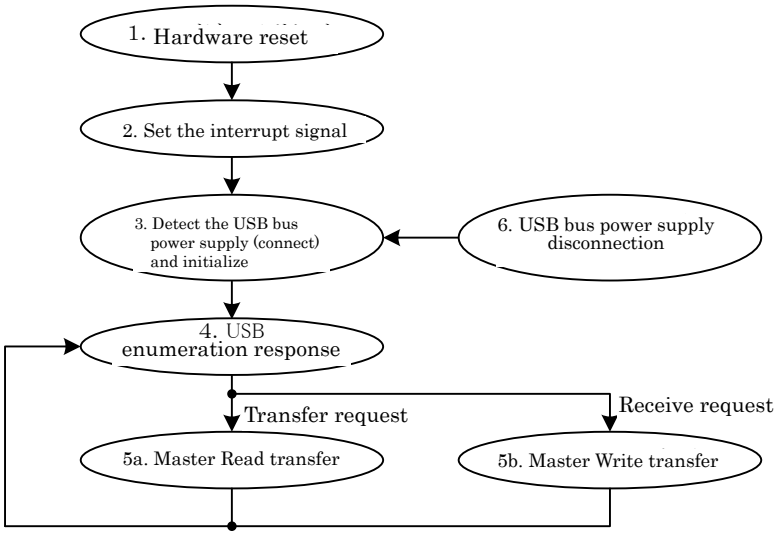


Figure 3.16.10 Overall operation sequence

3.16.2.9 Master Transfer Operation

This section describes the master transfer operation of UDC2AB.

When you start a master transfer, be sure to set the transfer setting of the relevant endpoint of UDC2 (bus_sel of UDC2 EPx_Status register (bit14)) to the direct access mode. It is prohibited to start DMAC when it is set to "Common bus access."

(1) Master Read transfer

- EOP enable mode

Master Read transfers when UDC2STSET<eopb_enable> is set to 1 (Master Read EOP enable) are described here. Master Read operations will be as follows:

1. Set Master Read Start Address and Master Read End Address registers.
2. Set the bits associated to the Master Read operation of DMAC Setting register and set 1 to the mr_enable.
3. UDC2AB starts the data transfer to the endpoint of UDC2. UDC2 transfers the data to the IN token from the USB host.
4. When the Master Read transfer reaches the Master Read end address, UDC2AB asserts the int_mr_end_add interrupt.
5. After the handling by the software ended, return to 1.

Note 1: About short packets

If the transfer size (Master Read End Address - Master Read Start Address + 1) is not the same size as the Max packet size, the last IN transfer will be the transfer of short packets.

Example: In case Master Read transfer size: 1035 bytes, and Max packet size: 512 bytes,

Transfers will take place in:

1st time: 512 bytes → 2nd time: 512 bytes → 3rd time: 11 bytes

Note 2: About int_mr_end_add interrupt

The int_mr_end_add interrupt occurs when the data transfer to the UDC2 endpoint is finished. In order to confirm whether the entire data has been transferred from UDC2 to the USB host, check the mrepempty bit of Master Status register.

- EOP Disable mode

Master Read transfers when UDC2STSET<eopb_enable > is set to 0 (Master Read EOP disable) are described here. Master Read operations will be as follows:

1. Set Master Read Start Address and Master Read End Address registers.
2. Set the register associated to the Master Read operation of DMAC Setting register and set 1 to the mr_enable bit.
3. UDC2AB starts the data transfer to the endpoint of UDC2. UDC2 transfers the data to the IN token from the USB host.
4. When reached the Master Read end address, UDC2AB asserts the int_mr_end_add interrupt. If the FIFO of the endpoint is as full as the maximum packet size in a Master Read transfer, the data will be transferred to the IN token from the USB host. If not, the data will remain in the FIFO and will be carried over to the next transfer.
5. After the handling by the software ended, return to 1.

Note: When UDC2AB is used in the EOP Disable mode, short packets will not be sent out even if the data string to be sent has been transferred. EOP Disable mode should be used only in case the size of the data string is a multiple of the maximum packet size.
The mode can be used if the total size of data string is a multiple of the maximum packet size. For example, the following transfer may be allowed:

Example:
- Size of the first Master Read transfer: 1000 bytes
- Size of the second Master Read transfer: 24 bytes (Total of first and second transfers = 1024 bytes)
- Maximum packet size: 512 bytes

A transfer of 512 bytes will be made twice for the IN transfer.

(2) Aborting of Master Read transfers

You can abort Master Read transfers with the following operation:

1. Use UDC2 Command register to set the status of the relevant endpoint to Disabled (EP_Disable). (If aborted without making the endpoint disabled, unintended data may be sent to the USB host.)
2. In order to stop the Master Read transfer, set 1 (Abort) to UDMSTSET <mr_abort>.
3. In order to confirm that the transfer is aborted, check that the mr_enable bit of DMAC Setting register was disabled to 0. Subsequent operations should not be made while the mr_enable bit is 1.

   (Information on the address where the transfer ended when aborted can be confirmed with Master Read Current Address and Master Read AHB Address registers.)
4. In order to initialize the Master Read transfer block, set 1 (Reset) to UDMSTSET<mr_reset>.
5. Use the Command register (EP_FIFO_Clear) to initialize the FIFO for the relevant endpoint.
6. Use the Command register (EP_Enable) to enable the relevant endpoint.

(3) Setting the maximum packet size in Master Read transfers

If the maximum packet size of the endpoint to be connected with the Master Read function of UDC2AB will be an odd number, there will be following restrictions to which you should pay attention:

- Even if the maximum packet size of the endpoint should be handled as an odd number, the setting of the max_pkt bit of UDC2 EPx_MaxPacketSize register should be an even number.

Note: Refer to the "section 3.16.4.2 “Appendix B About Setting an Odd Number of Bytes as MaxPacketSize" for more information on this setting.

- Set the eopb_enable bit of UDC2 Setting register to 1 (Master Read EOP enable).
- Make the transfer size to be specified for one Master Read transfer (Master Read End Address - Master Read Start Address + 1) not exceed the maximum packet size of an odd number.

(Example) A setting satisfying the above conditions:

- Set the maximum packet size of the endpoint (value to pass to the USB host) to be 63 bytes.
- Make the setting of the max_pkt bit of UDC2 EPx_MaxPacketSize register to be 64 bytes.
- Keep the transfer size to be specified for one Master Read transfer to 63 bytes or less.

(4) Master Write transfer

- Master Write transfer sequence

The operation of Master Write transfers are discussed here. Master Write operations will be as follows:

1. Set Master Write Start Address and Master Write End Address registers.
2. Set the bits associated to the Master Write operation of DMAC Setting register and set 1 to the mw_enable bit.
3. UDC2AB makes a Master Write transfer to the data in the endpoint received from the USB host.
4. Since the int_mw_end_add interrupt will be asserted when the writing ended to reach the Master Write End Address (with no timeout processed), you should make necessary arrangement with the software. UDC2 will return to 1 after receiving the correct packet.

Note: UDC2AB will assert the int_mw_set_add interrupt when the packet is received normally from the USB host with the mw_enable bit of DMAC Setting register disabled.

(5) Timeout

Master Write transfers would not finish if the OUT transfer from the USB host should stagnate before reaching the Master Write End Address during the transfer. In order to cope with such circumstances, you can set the timeout function.

When this timeout function is used, all data stored in the buffer in UDC2AB at the point of timeout will be transferred to AHB.

Timeout can be processed with the following operation:

1. Make an access to the Master Write Timeout register before starting a Master Write transfer and set timeoutset (timeout time) to make timeout_en enabled 1.
2. Start the Master Write transfer in accordance with the instruction in the preceding section.
3. When the timeout has occurred, the int_mw_timeout interrupt will be asserted. (The int_mw_end_add interrupt will not be asserted.) In that case, the Master Write transfer is not completed to reach the Master Write End Address. UDC2AB clears the mw_enable bit of DMAC Setting register to 0.
4. In Master Write Current Address register, the address to which the transfer has completed to the AHB end can be confirmed.

Please note that the timeout counter advances during the Master Write transfer with the timeout function enabled, but the counter will be reset to the preset value when the OUT transfer from the USB host to the relevant endpoint is received and begin recounting (see the figure below). It means that the time until timeout is "from the point when the last transfer from the USB host to the relevant endpoint has occurred during the Master Write transfer to the preset time," rather than "from the point when the Master Write transfer has begun to the preset time."

If you do not use the timeout function, be sure to set the timeout_en bit of Master Write Timeout register to "Disable 0" before starting the Master Write transfer. In that case, the transfer will not finish until reaching the preset Master Write End Address.

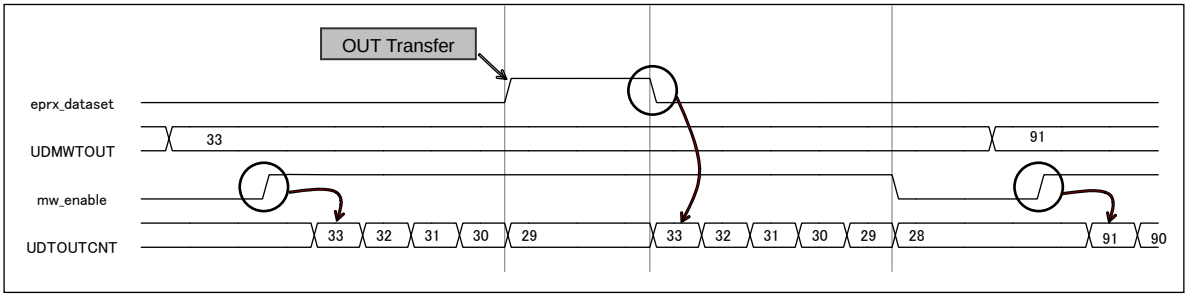


Figure 3.16.11 Example of MW timeout count

(6) Aborting of Master Write transfers

You can abort Master Write transfers with the following operation:

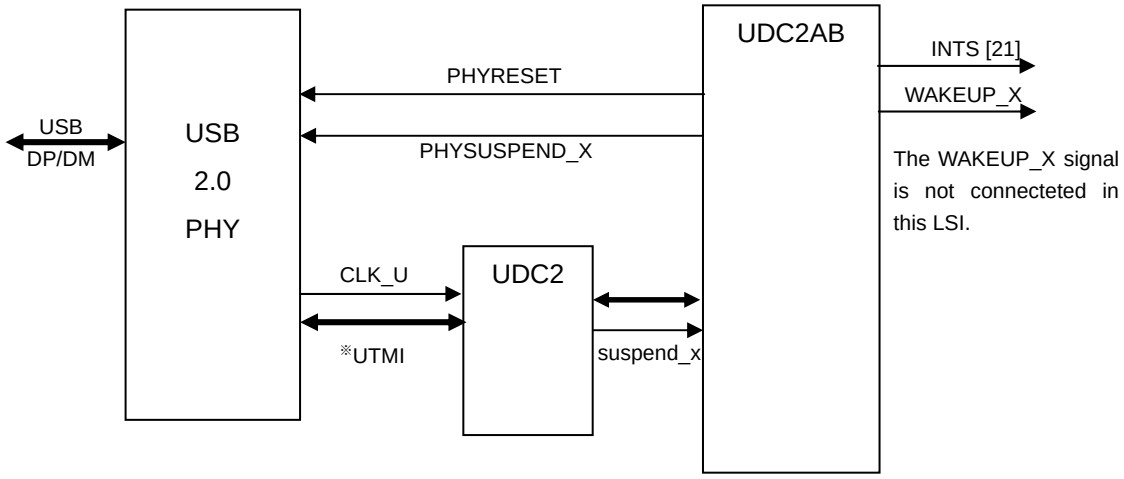
1. Use UDC2 Command register to set the status of the relevant endpoint to Disable (EP_Disable).
2. In order to stop the Master Write transfer, set 1 (Abort) to the mw_abort bit of DMAC Setting register.
3. In order to confirm the transfer is aborted, check the <mw_enable> of DMAC Setting register was disabled to 0. Subsequent operations should not be made while the <mw_enable> is 1. (Information on the address where the transfer ended when aborted can be confirmed with Master Write Current Address and Master Write AHB Address registers.)
4. In order to initialize the Master Write transfer block, set 1 (Reset) to the mw_reset bit of DMAC Setting register.
5. Use Command register (EP_FIFO_Clear) to initialize the FIFO for the relevant endpoint.
6. Use UDC2 Command register to set the status of the relevant endpoint to Enable (EP_Enable).

3.16.2.10 USB Power Management Control

In USB, operations related to power management including detection of USB bus power supply, suspending and resuming are also prescribed in addition to normal packet transfers. This section discusses about how to control those operations.

Below is a connection diagram of signals related to power management control.

Note: Be sure to see the USB 2.0 Specification for details of operations.



※UTMI: USB 2.0 Transceiver Macrocell Interface

Figure 3.16.12 Connection diagram of control signals

3.16.2.11 USB Bus Power Detecting Sequence

(1) Connect

This section describes the sequence when detecting the power supply. After detecting the bus power from the USB host (VBUS), initialize UDC2AB and UDC2 with the following procedures:

1. Use the pw_resetb bit of Power Detect Control register to make software reset. (The pw_resetb bit is not automatically released and should be cleared by software.)
2. Make an access to UDC2AB and UDC2 registers to make necessary initial settings.
3. Use UDC2 Command register to issue the USB Ready command. UDC2 notifies the USB host of the connection via PHY. This condition enables UDC2 to accept USB_RESET from the USB host.
4. Once USB_RESET from the USB host is detected, UDC2 initializes the registers inside UDC2 and enumeration with the USB host becomes available. When USB_RESET is detected, the int_usb_reset/int_usb_reset_end interrupt occurs.

Note: While UDC2AB originally has the function to assert the int_powerdetect interrupt when VBUS is detected, it is not supported for this LSI. UDPWCTL<pw_detect> always indicates 0.

(2) Disconnect

When the USB bus power is disconnected, UDC2AB makes notification by an external interrupt. Since master transfers will not automatically stop in such circumstances, you need to make an abort process. Then use the pw_resetb bit of Power Detect Control register to make software reset.

In case the system employs the control to stop CLK_H (AHB end) while USB is suspended, no interrupt will be notified even if the power is disconnected while CLK_H is stopped. In such cases, resuming of CLK_H is required using the WAKEUP_X output signal. See section 3.16.2.13 “(4) Signal operations when suspended and resumed (disconnected)" for more information.

3.16.2.12 USB_RESET

USB_RESET may be received not only when the USB host is connected but also at any timing.

UDC2AB asserts the int_usb_reset/int_usb_reset_end interrupt when UDC2 has received USB_RESET and returns to the default state. At this time, master transfers will not automatically stop. Use the abort function to end the transfers. Values are initialized by USB_RESET for some registers of UDC2, while they are retained for other registers (refer to the section of UDC2).

Resetting of UDC2 registers when USB_RESET is recognized should be made after the int_usb_reset_end interrupt has occurred. This is because UDC2 initializes UDC2 registers at the time it deasserts the usb_reset signal.

3.16.2.13 Suspend/Resume

(1) Shift to the suspended state

UDC2AB makes notification of detecting the suspended state of UDC2 by the int_suspend_resume interrupt and the suspend_x bit of Power Detect Control register. Since master transfers will not automatically stop in this circumstance, you should use the aborting function of each master transfer to make forcible termination if needed. In case PHY needs to be suspended (clock stop) after the necessary processes finished by software, you can set the phy_suspend bit of Power Detect Control register to make UDC2AB assert PHYSUSPEND_X which will put PHY in suspended state.

(2) Resuming from suspended state

UDC2AB makes notification of detecting the resuming state from the USB host by the int_suspend_resume interrupt and the suspend_x bit of Power Detect Control register. (In case the wakeup_en bit of Power Control register is set to be enabled when CLK_H is stopped, notification is made by the WAKEUP_X output signal.)

Note: While UDC2AB originally has the function to assert the Wakeup signal, it is not supported for this LSI.

Since the suspend signal to PHY (PHYSUSPEND_X) is automatically deasserted when resuming, controlling by software is not necessary unlike the case of suspending.

When resuming is recognized, make settings again for restarting master transfers.

(3) Remote wakeup from suspended state

When suspended, (in case PHY is in the suspended state) clocks for UDC2AB and UDC2 supplied by PHY (CLK_U) are stopped. Setting the phy_remote_wkup bit of Power Detect Control register to 1 in this state will make UDC2AB assert udc2_wakeup to UDC2 while deasserting the PHYSUSPEND_X signal. When the clock (CLK_U) output from PHY resumes after a certain period and the clock is supplied, UDC2 will automatically start the resuming operation.

(4) Signal operations when suspended and resumed (disconnected)

Based on the above descriptions, the signal operations when suspended and resumed (disconnected) are illustrated below.

Refer to "Figure 3.16.13 Operation of suspend/resume signals (when CLK_H is stopped)", "Figure 3.16.14 Operation of suspend/disconnect signals (when CLK_H is stopped)" if CLK_H should be stopped when resuming (disconnecting) from the USB host. Refer to "Figure 3.16.15 Operation of suspend/resume signals (when CLK_H is operating)" if CLK_H should be not stopped. Refer to "Figure 3.16.16 Operation of suspend/remote wakeup signals" for remote wakeup from UDC2AB.

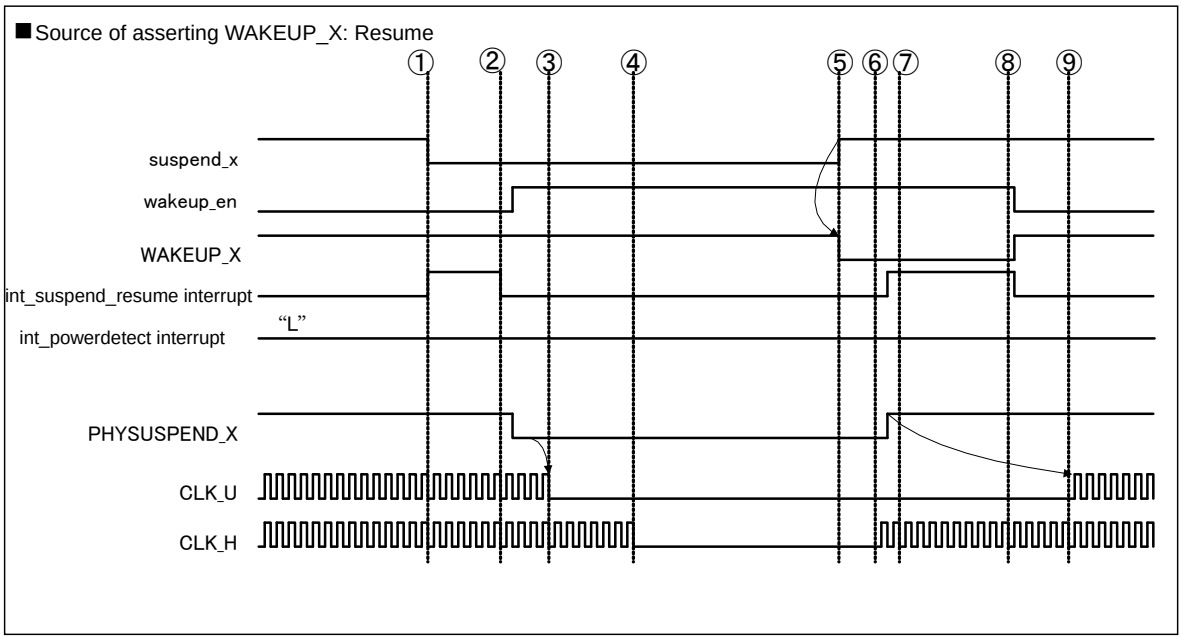


Figure 3.16.13 Operation of suspend/resume signals (when CLK_H is stopped)

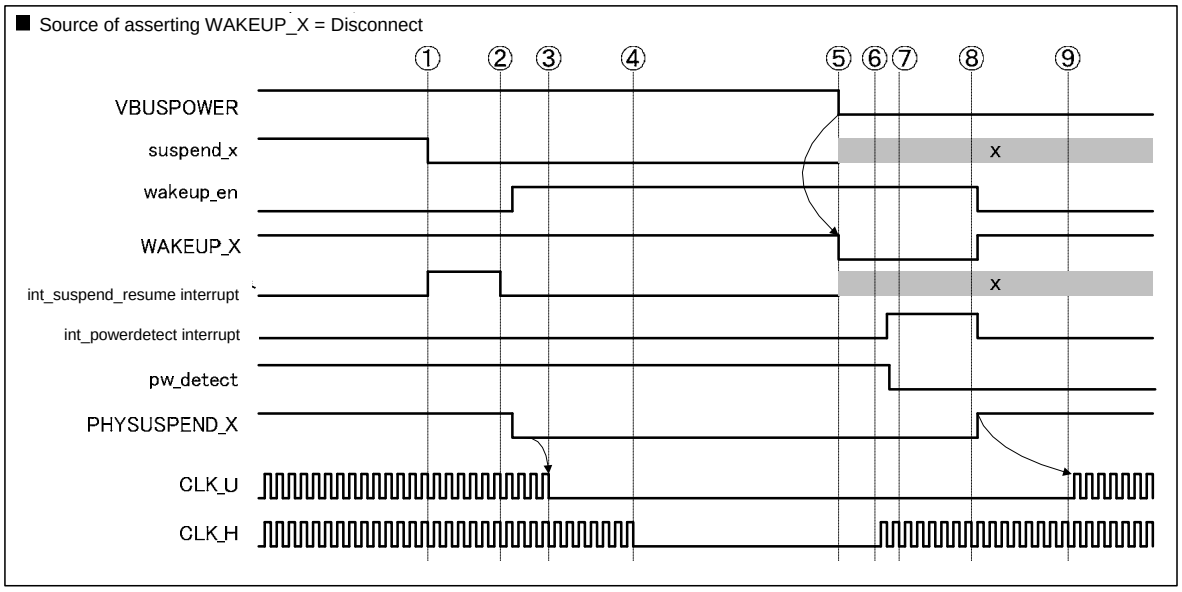


Figure 3.16.14 Operation of suspend/disconnect signals (when CLK_H is stopped)

Signal operation of Figure 3.16.13 and Figure 3.16.14:

① The int_suspend_resume interrupt occurs by detecting the suspended state on the USB bus.

② By the int_suspend_resume interrupt, the interrupt source is cleared by software and the phy_suspend bit of Power Detect Control register is set to 1.

③ Setting the phy_suspend bit will assert the PHYSUSPEND_X output signal to 0 which will stop the supply of CLK_U.

④ After setting the wakeup_en bit of Power Detect Control register to 1 by software, CLK_H can be stopped.

⑤ By detecting Resume on the USB bus or disconnecting (VBUS disconnected), the WAKEUP_X output signal will be asserted to 0 asynchronously.

⑥ Supply of CLK_H is started by the WAKEUP_X output signal. With the supply of CLK_H, the int_suspend_resume or the int_powerdetect interrupts will occur.

(If the rise of suspend_x is detected, the PHYSUSPEND_X output signal will be automatically deasserted.)

⑦ 2.5 μs after the interrupt is asserted (time required for the signal to stabilize when VBUS is disconnected), check the pw_detect bit of the Power Detect Control register.

Depending on the external interrupt,

proceed to ⑧-a: WAKEUP_X is asserted by Resume.

proceed to ⑧-b: WAKEUP_X is asserted by Disconnect.

<When Resumed>

⑧-a Software clears the interrupt source and the wakeup_en bit to deassert the WAKEUP_X output signal.

⑨-a Resumes from suspended state

<When Disconnected>

⑧-b Clears the phy_suspend bit to 0 by software and deasserts the PHYSUSPEND_X output signal. Also clears the interrupt source and the wakeup_en bit to deassert the WAKEUP_X output signal.

⑨-b Sets the pw_resetb bit of Power Detect Control register and initializes UDC2AB.

Note: While UDC2AB originally has the function to assert the Wakeup signal, it is not supported for this LSI.

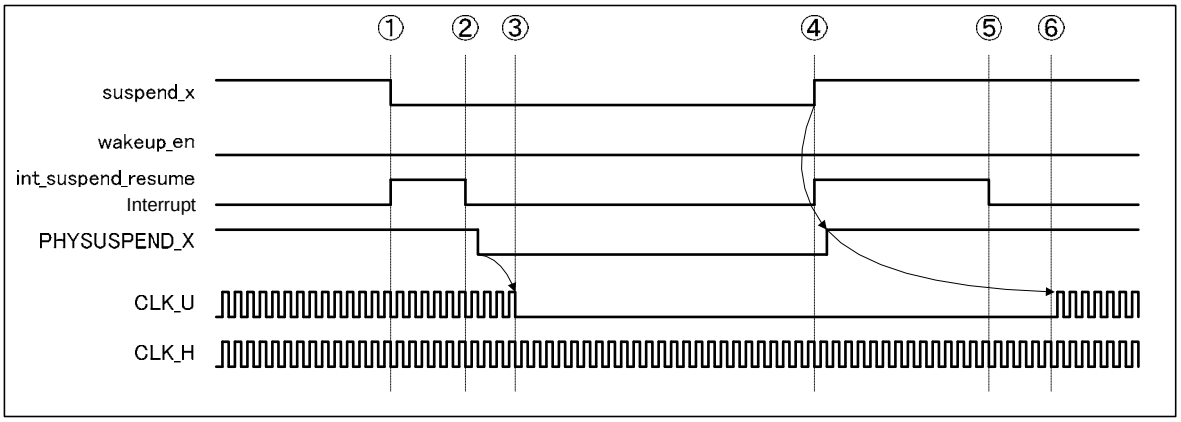


Figure 3.16.15 Operation of suspend/resume signals (when CLK_H is operating)

① The int_suspend_resume interrupt occurs by detecting the suspended state on the USB bus.

② By the int_suspend_resume interrupt, the interrupt source is cleared and the phy_suspend bit of Power Detect Control register is set to 1 by software.

③ Setting the phy_suspend bit will assert the PHYSUSPEND_X output signal which will stop the supply of CLK_U.

④ The int_suspend_resume interrupt occurs by detecting Resume on the USB bus.

By detecting the rise of suspend_x, the PHYSUSPEND_X output signal will be deasserted to 1.

⑤ By the int_suspend_resume interrupt, the interrupt source is cleared by Software.

⑥ Deasserting the PHYSUSPEND_X output signal will resume the supply of CLK_U.

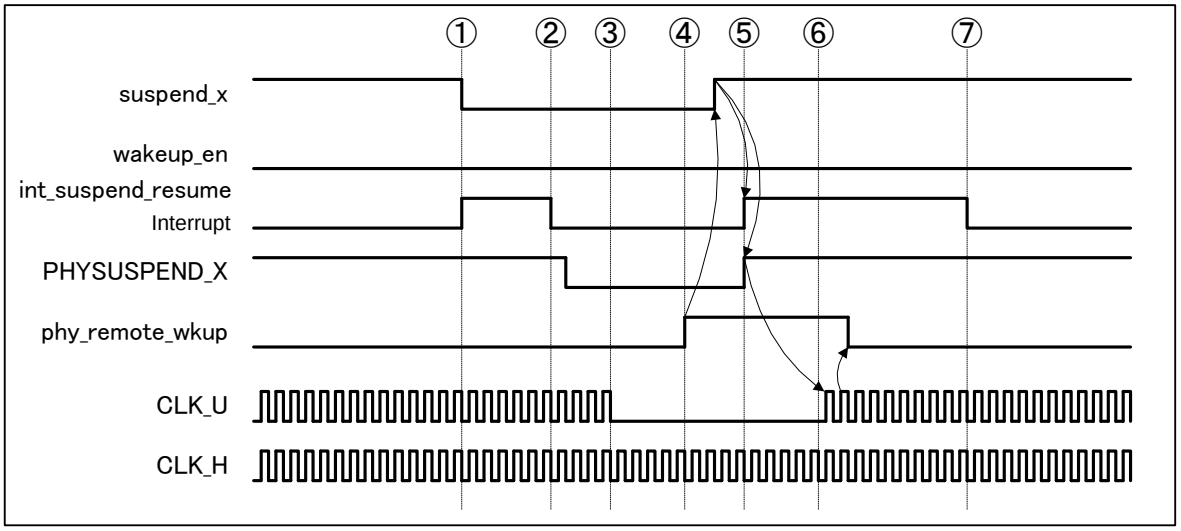


Figure 3.16.16 Operation of suspend/remote wakeup signals

① The int_suspend_resume interrupt occurs by detecting the suspended state on the USB bus.

② By the int_suspend_resume interrupt, the interrupt source is cleared and the phy_suspend bit of Power Detect Control register is set to 1 by software.

③ Setting the phy_suspend bit will assert the PHYSUSPEND_X output signal to 0 which will stop the supply of CLK_U.

④ When requesting remote wakeup, set the <phy_remote_wkup> bit of Power Detect Control register to 1. Setting the phy_remote_wkup bit will cause UDC2 to make a remote wakeup request on the USB bus. Also, suspend_x will be deasserted to 1 asynchronously.

⑤ Deasserting suspend_x will cause the int_suspend_resume interrupt to occur and the PHYSUSPEND_X output signal to be deasserted to 1.

⑥ Deasserting the PHYSUSPEND_X output signal will resume CLK_U. The phy_remote_wkup bit will be automatically cleared.

⑦ Clears the int_suspend_resume interrupt source.

3.16.3 Overview of UDC2

UDC2 is a core which controls connection of USB functions to the Universal Serial Bus. UDC2 automatically processes the USB protocol and its PHY-end interface can be accessed via UTMI.

UDC2 has the following functions and features:

- Supports Universal Serial Bus Specification Rev. 2.0.
- Supports both High-Speed (HS) and Full-Speed (FS) (Low-Speed is not supported).
- Supports Chirp.
- Processes USB protocol.
- Detects SOF/USB_RESET/SUSPEND/RESUME.
- Generates and checks packet IDs.
- Generates and checks data synchronization bits (DATA0/DATA1/DATA2/MDATA).
- Checks CRC5, generates and checks CRC16.
- Supports PING.
- Supports 4 transfer modes (Control/Interrupt/Bulk/Isochronous).
- Supports 4 endPoint.
- Supports Dual Packet Mode (except for endpoint 0).
- Endpoints 1 to 3 can directly access FIFO (Endpoint-I/F).
- Supports USB 2.0 Transceiver Macrocell Interface (UTMI) (16 bits @ 30 MHz).

3.16.3.1 Internal Block Structure of UDC2

The following are the block structure and outline of each block of UDC2.

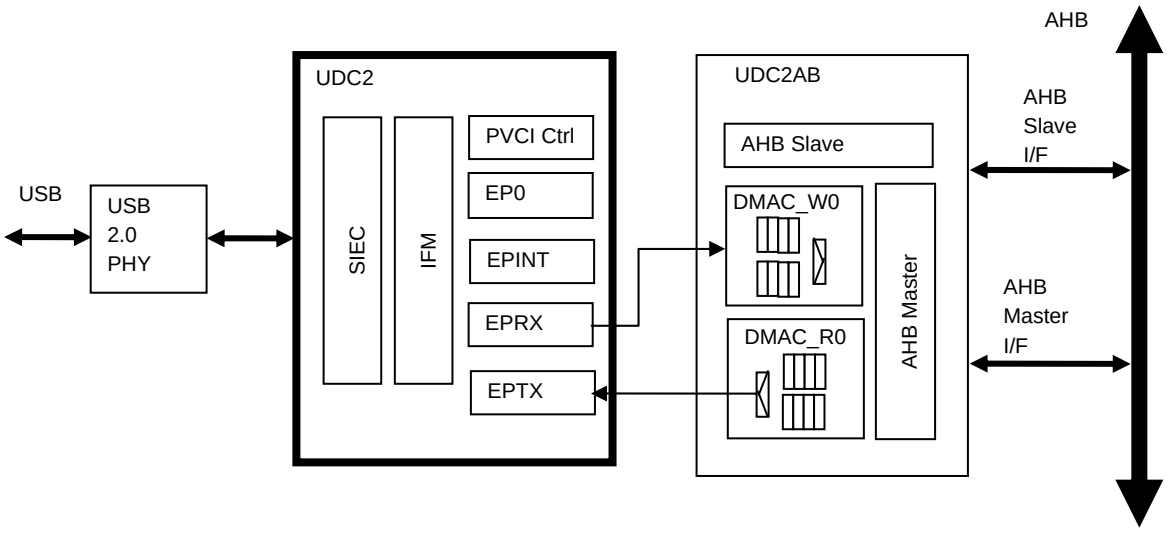


Figure 3.16.17 Block diagram of UDC2

(1) SIEC (Serial Interface Engine Control) block

This block manages the protocol in USB. Its major functions are:

- Checks and generates PIDs.
- Checks and generates CRCs.
- Checks device addresses.
- Manages transfer speed (HS/FS).
- Controls PHY (transfer speed (HS/FS), mode, etc.).
- Generates test modes.

(2) IFM block

This block controls SIEC and endpoints. Its major functions are:

- Writes the received data to the relevant endpoints when received an OUT-Token.
- Reads the transmit data from the relevant endpoints when an IN-Token is received.
- Controls and manages the status of UDC2.0.

(3) PVCIIF block

This block controls reading and writing between IFM and external register access bus (PVCI).

PVCI bus accesses via UDC2AB.

(4) EP0 block

This block controls sending and receiving data in Control transfers. When sending or receiving data with DATA-Stage of Control transfers, you should access the FIFO in this block via PVCI-I/F.

(5) EPx block

This block controls sending and receiving data of EPx (x = 1, 2, 3). FIFO can be directly accessed via the Endpoint-I/F. The Endpoint-I/F can make burst transfers.

Please note there are two endpoints; one for sending and another for receiving. Direction of endpoints (send/receive) will be fixed on a hardware basis.

3.16.3.2 Specifications of Flags

The UDC2 core outputs various events on USB as flags when they occur. This section discusses those flags.

(1) USB_RESET

Asserts "H" while receiving USB_RESET. Since UDC2 returns to the Default-State by receiving USB_RESET, the application also needs to return to the Default-State.

In Full-Speed operation, UDC2 asserts this flag when SE0 on the USB bus was recognized for 2.5 μs or longer. In High-Speed operation, the flag is asserted when SE0 was recognized for 3 ms or longer, after determining whether USB_RESET or suspended state. Then, after UDC2 has driven Chirp-K for about 1.5 ms the flag will be deasserted when either one of the following states was recognized:

1. Chirp from the host (K-J-K-J-K-J) was recognized.
2. 2 ms or longer has passed without recognizing Chirp from the host (K-J-K-J-K-J).

Note: While the time when the host begins Chirp and the driving time of Chirp-K and Chirp-J depend on the host, asserting period of the USB_RESET flag is around 1.74 ms to 3.5 ms.

(2) INT_SETUP

In Control transfers, asserts "H" after receiving the Setup-Token. When this interrupt is recognized, the software should read the Setup-Data storage register (8 bytes) to make judgment of request. This interrupt will be deasserted by writing 1 into the relevant bit (bit 0) of INT register. INT register should be cleared at the point the interrupt was recognized.

(3) INT_STATUS_NAK

In Control transfers, when the host proceeds to the STATUS-Stage and transmits packets while UDC2 is processing the DATA-Stage (before issuing the "Setup_Fin" command), UDC2 will return "NAK" and asserts this flag to "H". When this interrupt is recognized, the software should issue the "Setup_Fin" command from the Command register to make the STATUS-Stage of UDC2 end. This interrupt will be deasserted by writing 1 into the relevant bit (bit 1) of INT register. INT register should be cleared at the point the interrupt was recognized.

(4) INT_STATUS

In Control transfers, asserts "H" after finishing the STATUS-Stage normally. This interrupt will be deasserted by writing 1 into the relevant bit (bit 2) of INT register. INT register should be cleared at the point the interrupt was recognized.

(5) INT_EP0

In the DATA-Stage of Control transfers, asserts "H" when "ACK" was sent or received (when the transaction finished normally). This interrupt will be deasserted by writing 1 into the relevant bit (bit 5) of INT register. INT register should be cleared at the point the interrupt was recognized.

(6) INT_EP

In endpoints other than Endpoint 0, asserts “H” when “ACK” was sent or received (when the transaction finished normally). In that case, which endpoint the transfer was made can be identified by checking INT_EP register. This interrupt will be deasserted by writing 1 into the relevant bit (bit 6) of INT register, or by writing 1 into all bits set in INT_EP register. INT register should be cleared at the point the interrupt was recognized.

(7) INT_RX_ZERO

“H” is asserted when Zero-Length data is received. In Control transfers, however, “H” is asserted only when Zero-Length data is received in the DATA-Stage. It will not be asserted when Zero-Length data is received in the STATUS-Stage. Which endpoint has received the data can be identified by reading the bits [11:8] of Command register or checking INT_RX_DATA0 register. This interrupt will be deasserted by writing 1 into the relevant bit (bit 3) of INT register, or by writing 1 into all bits set in INT_RX_DATA0 register. INT_RX_DATA0 register should be cleared at the point the interrupt was recognized.

(8) INT_SOF

Asserts “H” when SOF was received. This interrupt will be deasserted by writing 1 into the relevant bit (bit 4) of INT register. INT register should be cleared at the point the interrupt was recognized.

SOF is a packet indicating the start of a frame (μ frame). It is transmitted from the host to devices every 1 ms in the Full-Speed transfers, and every 125 μs in the High-Speed transfers.

(9) INT_NAK

In endpoints other than Endpoint 0, asserts “H” when NAK is transmitted. In that case, which endpoint has transmitted the NAK can be identified by checking INT_NAK register. This interrupt will be deasserted by writing 1 into the relevant bit (bit 7) of INT register, or by writing 1 into all bits set in INT_NAK register. By default, this flag will not be asserted when NAK was transmitted. Therefore, you should write 0 into the relevant endpoint of INT_NAK_MASK register to release the mask in order to use this flag.

3.16.3.3 Registers

UDC2 has the following registers:

- Device Status
  - Address-State register
  - Frame register
  - USB-Testmode register
  - Command register
- Setup Data Storage
  - bRequest-bmRequestType register
  - wValue register
  - wIndex register
  - wLength register
- Interrupt Control
  - INT register
  - INT_EP register
  - INT_EP_MASK register
  - INT_RX_DATA0 register
  - INT_NAK register
  - INT_NAK_MASK register
- EP0 Control Status
  - EP0_MaxPacketSize register
  - EP0_Status register
  - EP0_Datasize register
  - EP0_FIFO register
- EPx Control Status
  - EPx_MaxPacketSize register
  - EPx_Status register
  - EPx_Datasize register
  - EPx_FIFO register

Table 3.16.2 shows the register map of UDC2.

Table 3.16.2 Register map

Address =(0xF440_0000)

	Register Name	Address (base +)	Description
UDC2 *2), *3)	UD2ADR	0x0200	UDC2 Address-State Register
	UD2FRM	0x0204	UDC2 Frame Register
	UD2TMD	0x0208	UDC2 USB-Testmode Register
	UD2CMD	0x020C	UDC2 Command Register
	UD2BRQ	0x0210	UDC2 bRequest-bmRequestType Register
	UD2WVL	0x0214	UDC2 wValue Register
	UD2WIDX	0x0218	UDC2 wIndex Register
	UD2WLGTH	0x021C	UDC2 wLength Register
	UD2INT	0x0220	UDC2 INT Register
	UD2INTEP	0x0224	UDC2 INT_EP Register
	UD2INTEPMSK	0x0228	UDC2 INT_EP_MASK Register
	UD2INTRX0	0x022C	UDC2 INT_RX_DATA0 Register
	UD2EP0MSZ	0x0230	UDC2 EP0_MaxPacketSize Register
	UD2EP0STS	0x0234	UDC2 EP0_Status Register
	UD2EP0DSZ	0x0238	UDC2 EP0_Datasize Register
	UD2EP0FIFO	0x023C	UDC2 EP0_FIFO Register
	UD2EP1MSZ	0x0240	UDC2 EP1_MaxPacketSize Register
	UD2EP1STS	0x0244	UDC2 EP1_Status Register
	UD2EP1DSZ	0x0248	UDC2 EP1_Datasize Register
	UD2EP1FIFO	0x024C	UDC2 EP1_FIFO Register
	UD2EP2MSZ	0x0250	UDC2 EP2_MaxPacketSize Register
	UD2EP2STS	0x0254	UDC2 EP2_Status Register
	UD2EP2DSZ	0x0258	UDC2 EP2_Datasize Register
	UD2EP2FIFO	0x025C	UDC2 EP2_FIFO Register
	Reserved	0x0260 to 0x03FC	
	UD2INTNAK	0x0330	UDC2 INT_NAK Register
	UD2INTNAKMSK	0x0334	UDC2 INT_NAK_MASK Register
	Reserved	0x0338 to 0x03FC	

The following sections describe the registers in UDC2 in detail.
The descriptions of each bit have the following meanings:

(Example)

EPx_Datasize (EPx_Datasize register)

Address = (0xF440_0000) + (0x0000)

Bit	Bit Symbol (Note 1)	Type (Note 2)	Reset Value (Note 3)	Description
[31:11]	–	–	Undefined	Read as undefined.
[10:0]	size[10:0]	RO	0y00000000000, (-)(-)(-)(-)(-)(-) (-)(-)(-)(-)	Indicates the number of valid data bytes stored in EPx_FIFO. In the Dual Packet mode, the number of data bytes to be accessed first will be shown.

Note 1: Bit Symbol

Name of each bit.

Those shown as "–" are reserved bits which cannot be written. 0 will be returned when read.

Note 2: Register properties

RO : Read only. Write is ignored.

WO : Write only. “0” will be returned when read.

R/W : Read/Write

Note 3: Reset value

“Reset Value” is the initial value for the bit after resetting (1 or 0). Initial values after ”USB_ RESET” are shown in parentheses and those bits which will not be reset by “USB_RESET” are indicated with a hyphen.

The following subsections describe each register in detail.

(1) Device Status Registers

1. UD2ADR (Address-State register)

Address = (0xF440_0000) + (0x0200)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	stage_err	RO	0y0	Indicates whether Control transfers finished normally up to the STATUS-Stage. 0y0: Other than below conditions 0y1: Received the Setup-Token in DATA-Stage/STATUS-Stage or "STALL" transmission.
[14]	ep_bi_mode	R/W	0y0, (-)	Selects whether to use the endpoint bidirectionally as a driver. (Note 2) 0y0: Single direction 0y1: Dual direction
[13:12]	cur_speed[1:0]	RO	0y01,(Note1)	Indicates the present transfer mode on the USB bus. 0y00: Reserved 0y01: Full-Speed 0y10: High-Speed 0y11: Reserved
[11]	Suspend	RO	0y0	Indicates whether or not UDC2 is in suspended state. 0y0: Normal, 0y1: Suspended
[10]	Configured	R/W	0y0	Sets the present device state of UDC2. 0y001:Default (to be set when the DeviceAddress=0 was specified by the Set_address request in Default/Address state (this will be set by the hardware when USB_RESET is received)) 0y010:Addressed (to be set when ConfigurationVallue = 0 was specified by the Set_configuration request after the Set_address request finished normally and in the Address/Configured state) 0y100:Configured (to be set when the Set_config request is received)
[9]	Addressed	R/W	0y0	
[8]	Default	R/W	0y0,(1)	
[7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	dev_adr[6:0]	R/W	0y0000000	Sets the device address assigned by the host.

Note 1: The initial value of cur_speed[1:0] (bits [13:12]) after USB_RESET is “0y10” (high-Speed) if the Chirp sequencehas been sucessful, and “0y01” (Full-Speed) if it has failed.

Note 2 : About TMPA900CM, EP0: single direction / dual direction . EP1, EP2 and EP3: single direction. only

[Description]

a. <stage_err>

Indicates whether Control transfers finished normally up to the STATUS-Stage.

1 will be set when the Setup-Token is received in DATA-Stage/STATUS-Stage or in the case of “STALL” transmission. When set, it will be cleared when the next Control transfer has been finished normally.

0y0: Other than above conditions

0y1: Received the Setup-Token in DATA-Stage/STATUS-Stage or “STALL” transmission.

b. <ep_bi_mode>

Selects whether to use the endpoint bidirectionally as a driver.

Setting this bit to 1 will enable an endpoint number to be used bidirectionally in USB communication.

0y0: Single direction

0y1: Dual direction

c. <cur_speed[1:0]>

Indicates the present transfer mode on the USB bus.

0y00: Reserved

0y01: Full-Speed

0y10: High-Speed

0y11: Reserved

d. <suspend>

Indicates whether or not UDC2 is in suspended state.

0y0: Normal

0y1: Suspended

e. <configured>, <addressed>, <default>

Set the present device state of UDC2. This should be set in accordance with the request received from the host. Please note that you should not set 1 to more than one bit.

0y001: default (to be set when the DeviceAddress = 0 was specified by the Set_address request in Default/Address state (this will be set by the hardware when USB_RESET is received))

0y010: addressed (to be set when ConfigurationVallue = 0 was specified by the Set_configuration request after the Set_address request finished normally and in the Address/Configured state)

0y100: configured (to be set when the Set_config request is received)

f. <dev_adr[6:0]>

Sets the device address assigned by the host.

The device address should be set after Set_address has finished normally (after STATUS-Stage finished normally).

2. UD2FRM (Frame register)

Address = (0xF440_0000) + (0x0204)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	create_sof	R/W	0y0	Sets whether to generate the SOF flag internally when the SOF from the host is unavailable due to a bus error. 0y0: Generates no flag 0y1: Generates a flag
[14]	–	–	Undefined	Read as undefined. Write as zero.
[13:12]	f_status[1:0]	RO	0y10, (-)(-)	Indicates the status of the frame number. 0y00: Before 0y01: Valid 0y10: Lost
[11]	–	–	Undefined	Read as undefined. Write as zero.
[10:0]	frame[10:0]	RO	0y00000000000	Indicates the frame number when SOF is received.

[Description]

a. <create_sof>

Sets whether to generate the SOF flag internally when the SOF from the host is unavailable due to a bus error. This should be set if you wish to synchronize frames by SOF in Isochronous transfers. If enabled, the internal frame time counter will operate and the SOF flag will be output even when the SOF-Token could not be received successfully.

0y0: Generates no flag

0y1: Generates a flag

b. <f_status[1:0]>

Indicates the status of the frame number.

0y00: Before: Will be set if the Micro SOF/SOF was not received when 1frame-time (HS:125 μs/FS:1 ms) has passed after receiving the Micro SOF/SOF when Create_sof is enabled. In the Frame register, the frame number received in the last Micro SOF/SOF has been set.

0y01: Valid: Will be set when the Micro SOF/SOF was received. Indicates a valid frame number is set in the Frame register.

0y10: Lost: Indicates that the frame number maintained by the host is not synchronized with the value of Frame register. Accordingly, this will be set in the following cases:

1. When the system was reset or suspended
2. If the next Micro SOF/SOF was not received when 2 frame-time (HS: 125 × 2 us/FS: 1 × 2 ms) has passed after receiving the previous Micro SOF/SOF when Create_sof is enabled.

However, since the same frame number of Micro SOF will be sent eight times in a row in High-Speed transfers, the frame number sent from the host may seem to be synchronized with the value of Frame register even in the Lost status. Please note, however, they are not actually synchronized when considering the frame number and the number of times that frame number was sent. Also note that transition to the Lost status only happens after the system was reset or when it is suspended if Create_sof is disabled.

c. <frame[10:0]>

Indicates the frame number when SOF is received.

This will be valid when f_status is “Valid”. Should not be used if f_status is “Before” or “Lost” as correct values are not set.

3. UD2TMD (USB-Testmode register)

Address = (0xF440_0000) + (0x0208)

Bit	Bit Symbol	Type	Reset Value	Description
[31:13]	–	–	Undefined	Read as undefined. Write as zero.
[12]	packet	RO	0y0	Indicates the test mode currently set. 0y0001: test_j 0y0010: test_k 0y0100: se0_nak 0y1000: test_packet
[11]	se0_nak	RO	0y0	
[10]	test_k	RO	0y0	
[9]	test_j	RO	0y0	
[8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	t_sel[7:0]	R/W	0x00	Sets the test mode.

[Description]

a. <packet>, <se0_nak>, <test_k>, <test_j>,
Indicates the test mode currently set.
0y0001: test_j
0y0010: test_k
0y0100: se0_nak
0y1000: test_packet

b. <t_sel[7:0]>
Sets the test mode. Set the value of TestModeSelectors specified by Set_Feature.

4. UD2CMD (Command register)

Address = (0xF440_0000) + (0x020C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	int_toggle	R/W	0y0	Makes the DATA-PID toggle when Handshake is not received in Interrupt-IN transfers. 0y0: Do not toggle when not received 0y1: Toggle when not received as well
[14:12]	–	–	Undefined	Read as undefined. Write as zero.
[11:8]	rx_nullpkt_ep[3:0]	RO	0y0000, (-)(-)(-)(-)	Indicates the receiving endpoint when Zero-Length data is received.
[7:4]	ep[3:0]	R/W	0y0000	Sets the endpoint where the command to be issued will be valid.
[3:0]	com[3:0]	R/W	0y0000	Sets the command to be issued for the endpoint selected in ep[3:0]. 0x0: Reserved 0x1: Setup_Fin 0x2: Set_DATA0 0x4: EP_Stall 0x5: EP_Invalid 0x6: Reserved 0x7: EP_Disable 0x8: EP_Enable 0x9: All_EP_Invalid 0xA: USB_Ready 0xB: Setup_Received 0xC: EP_EOP 0xD: EP_FIFO_Clear 0xE: EP_TX_0DATA 0xF: Reserved

[Description]

a. <int_toggle>

Makes the DATA-PID toggle when Handshake is not received in Interrupt-IN transfers.

0y0: Do not toggle when not received

0y1: Toggle when not received as well

b. <rx_nullpkt_ep[3:0]>

Indicates the receiving endpoint when Zero-Length data is received.

When the "int_rx_zero" flag is asserted, read this bit to check to which endpoint it was asserted. Once Zero-Length data is received and the endpoint number is retained, the value of this register will be retained until Zero-Length data is received next time or hardware reset (reset_x = 0) is made. If there is more than one endpoint of OUT direction, this bit will be renewed each time Zero-Length data is received. In that case, INT_RX_DATA0 register can be used to identify which endpoint has received the data.

c. <ep[3:0]>

Sets the endpoint where the command to be issued will be valid. (Do not specify an endpoint not existing.)

d. <com[3:0]>

Sets the command to be issued for the endpoint selected in ep[3:0].

0x0: Reserved — Not to be specified.

0x1: Setup_Fin — (Should be issued only for EP0.)

This is a command for setting the end of DATA-Stage in Control transfers. As UDC2 continues to send back "NAK" to the STATUS-Stage until this command is issued, the command should be issued when the DATA-Stage finishes or INT_STATUS_NAK was received.

Note: "Setup_Fin" command after reading all received data during data stage of Control-WR.

0x2: Set_DATA0 — (Can be issued to EPs except EP0. Should not be issued to EP0.)

A command for clearing toggling of endpoints. While toggling is automatically updated by UDC2 in normal transfers, this command should be issued if it needs to be cleared by software.

0x3: EP_Reset — (Can be issued to any EP.)

A command for clearing the data and status of endpoints. Issue this command when you want to reset an endpoint in such cases as setting endpoints of Set_Configuration and Set_Interface or resetting the endpoint by Clear_Feature. This command will reset the following 5 points:

1. Clear the bits 13-12 (toggle) of EP0/EPx_Status register to DATA0
2. Clear the bits 11-9 (status) of EP0/EPx_Status register to Ready
3. Clear the bit 12 (dset) of EP0/EPx_MaxPacketSize register and clear the EP0/EPx_Datasize register
4. Clear the bit 15 (tx_0data) of EP0/EPx_MaxPacketSize register
5. Clear the bit 15 (tx_0data) of EP0/EPx_MaxPacketSize register

UDC2 makes toggling control by hardware for every transfer. If this command is issued when a transfer of endpoints is in progress, toggling of the relevant endpoint will also be cleared which may cause the synchronization with the host be lost. As in the case of receiving requests as mentioned above, the command should be issued when it is possible to make synchronization with the host.

0x4: EP_Stall — (Can be issued to any EP.)

A command for setting the status of endpoints to "Stall". Issue this command when you want to set the status of an endpoint to "Stall" in such cases as stalling an endpoint by Set_Feature. When this command is issued, "STALL" will be always sent back for the endpoint set. However, the Stall status of EP0 will be cleared when the Setup-Token is received.

This command should not be issued for endpoints where Isochronous transfers are used, since transfers are made without Handshake in Isochronous transfers. Even if this command is issued for endpoints where Isochronous transfers are set (by t_type (bits[3:2]) of EPx_Status register), "STALL" will not be sent back.

0x5: EP_Invalid — (Can be issued to EPs except EP0. Should not be issued to EP0.)

A command for setting the status of endpoints to "Invalid". Please issue this command when disabling endpoints that will not be used when using Set_Config or Set_Interface to set endpoints. When this command is issued, the endpoints set will make no response. This command should not be issued while transfers of each endpoint are in progress.

0x6: Reserved	Not to be specified.
0x7: EP_Disable	(Can be issued to EPs except EP0. Should not be issued to EP0.) A command for making an endpoint disabled. When this command is issued, "NAK" will be always sent back from the endpoint set. This command should not be issued for endpoints where Isochronous transfers are used, since transfers are made without Handshake in Isochronous transfers. Even if this command is issued for endpoints where Isochronous transfers are set (by t_type (bits[3:2]) of EPx_Status register), "NAK" will not be sent back.
0x8: EP_Enable	(Can be issued to EPs except EP0. Should not be issued to EP0.) A command for making an endpoint enabled. Issue this command to cancel the disabled status set by "EP_Disable" command.
0x9: All_EP_Invalid	(Setting for EP is invalid.) A command for setting the status of all endpoints other than EP0 to "Invalid". Issue this command when you want to apply the "EP_Invalid" command for all endpoints. Issue this command when processing Set_Configuration and Set_Interface like the "EP_Invalid" command.
0xA: USB_Ready	(Should be issued only for EP0.) A command for making connection with the USB cable. Issue this command at the point when communication with the host has become effective after confirming the connection with the cable. Pull-Up of DDP will be made only after this command is issued, and the status of cable connection will be sent to the host. Please note that the device state of UDC2 (bits[10:8] of Address-state register) will be set to "Default" when this command was issued.
0xB: Setup_Received	(Should be issued only for EP0.) A command for informing UDC2 that the SETUP-Stage of a Control transfer was recognized. Issue this command after accepting the INT_SETUP interrupt and the request code was recognized. As UDC2 continues to send back "NAK" to the DATA-Stage/STATUS-Stage until this command is issued, the command should be issued at the end of the INT_SETUP interrupt processing routine.
0xC: EP_EOP	(Can be issued to any EP.) A command for informing UDC2 that the transmit data has been written. Issue this command when transmitting data with byte size smaller than the maximum transfer bytes (FIFO capacity of the endpoint or MaxPacketSize, whichever smaller). Issuing this command will set the Dataset flag and the data will be sent back to IN-Token from the host. It should not be used when setting Zero-Length data or data of MaxPacketSize.
0xD: EP_FIFO_Clear	(Can be issued to any EP.) A command for clearing the data of an endpoint. The bit 12 (dset) of EPx_MaxPacketSize register and the EPx_Datasize register will be cleared at the same time. Issue this command when you want to clear the data currently stored in the FIFO before transmitting the data to the host and set the latest data, for instance in Interrupt transfers. If this command is issued while accessing the Endpoint-I/F, the FIFO of the endpoint will not be successfully cleared. When issuing this command, epx_val of Endpoint-I/F should be set to 0 before issuing.

0xE: EP_TX_0DATA	(Can be issued to any EP.) A command for setting Zero-Length data to an endpoint. Issue this command when you want to transmit Zero-Length data. In the case of transmitting Zero-Length data in Bulk-IN transfers and others to indicate the final transfer, read EPx_Datasize register and confirm it is 0 (no data exists in the FIFO of EPx) before setting this command. In the case of writing data from Endpoint-I/F, set this command after the data was written and epx_val became 0. When this command was set, bit 15 (tx_0data) of EPx_MaxPacketSize register of the endpoint will be set. Ensure that this tx_0data becomes 0 before setting the next data. In Isochronous-IN transfers, Zero-Length data will be automatically transmitted to the IN-Token if no data is set in the FIFO of the endpoint. This command should not be issued in that case.
0xF: Reserved	Not to be specified.

Settings for the following commands will be suspended when issued during a USB transfer, which will be executed after the USB transfer has finished. Suspension of the command will take place for each endpoint.

0x2: Set_DATA0

0x3: EP_Reset

0x4: EP_Stall

0x5: EP_Invalid

0x7: EP_Disable

0x8: EP_Enable

0x9: All_EP_Invalid

0xD: EP_FIFO_Clear

0xE: EP_TX_0DATA

Therefore, when commands were issued successively for the same endpoint while a USB transfer is in progress, commands will be overwritten and only the one last issued will be valid. If you need to issue commands to an endpoint successively, poll Epx_Status/ Epx_Datasize register to confirm that the command has become valid before issuing next ones. Also, when making an access to the Endpoint-I/F immediately after clearing the FIFO using the EP_Reset/EP_FIFO_Clear command, poll EPx_Datasize register to confirm that the command has become valid before resuming the access to the Endpoint-I/F.

For Endpoint 0, the following commands will be invalid until the Setup_Received command is issued after receiving the Setup-Token:

0x1: Setup_Fin

0x2: Set_DATA0

0x3: EP_Reset

0x4: EP_Stall

0xC: EP_EOP

0xD: EP_FIFO_Clear

0xE: EP_TX_0DATA

When the “EP_Stall” command was set to EPx, “Stall” will be set to the bits[11:9] (status) of EPx_Status register. When EP_Disable was set, 1 will be set to the bit 8 (disable) of EPx_Status register. When these two commands (EP_Stall and EP_Disable) were set to the same EPx and the status becomes “Stall” with disable = 1, “STALL” will be transmitted in the transfer.

When the “EP_Invalid” command was set to EPx, “Invalid” will be set to the status of EPx_Status register. When the two commands (EP_Invalid and EP_Disable) were set to the same EPx and the status becomes “Invalid” with disable = 1, no response will be made in the transfer.

When EPx_Status register has disable = 1 and EPx_MaxPacketSize register has bit 15 (tx_0data) = 1, Zero-Length data will be transmitted once in the transfer. After the Zero-Length data was successfully transferred, “NAK” will be transmitted.

(2) Setup-Data Storage Registers

These registers overwrite the Setup-Data they received each time after receiving a Setup-Token. When the INT_SETUP interrupt has occurred, read these registers to determine the type of the request.

1. UD2BRQ (bRequest-bmRequestType register)

Address = (0xF440_0000) + (0x0210)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:8]	request[7:0]	RO	0x00	Indicates the data of the second byte received with the Setup-Token (bRequest field).
[7]	dir	RO	0y0	Indicates the data of the first byte received with the Setup-Token (bRequestType field). Direction of Control transfers 0y0: Control-WR transfer 0y1: Control-RD transfer
[6:5]	req_type[1:0]	RO	0y00	Type of requests 0y00: Standard request 0y01: Class request 0y10: Vendor request 0y11: Reserved
[4:0]	recipient[4:0]	RO	0y00000	Requests are received by: 0y00000: Device 0y00001: Interface 0y00010: Endpoint 0y00011: etc. 0y00100-0y11111: Reserved

[Description]

a. <request[7:0]>

Indicates the data of the second byte received with the Setup-Token (bRequest field).

b. <dir>

Indicates the data of the first byte received with the Setup-Token (bRequestType field).

Direction of Control transfers

0y0: Control-WR transfer

0y1: Control-RD transfer

c. <req_type[1:0]>

Type of requests

0y00: Standard request

0y01: Class request

0y10: Vendor request

0y11: Reserved

d. <recipient[4:0]>

Requests are received by:

0y00000: Device

0y00001: Interface

0y00010: Endpoint

0y00011: etc.

0y00100-0y11111: Reserved

2. UD2WVL (wValue register)

Address = (0xF440_0000) + (0x0214)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:8]	value[15:8]	RO	0x00	Indicates the data of the fourth byte received with the Setup-Token (wValue (H) field).
[7:0]	value[7:0]	RO	0x00	Indicates the data of the third byte received with the Setup-Token (wValue (L) field).

[Description]

a. <value[15:8]>
Indicates the data of the fourth byte received with the Setup-Token (wValue (H) field).

b. <value[7:0]>
Indicates the data of the third byte received with the Setup-Token (wValue (L) field)

3. UD2WIDX (wIndex register)

Address = (0xF440_0000) + (0x0218)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:8]	index[15:8]	RO	0x00	Indicates the data of the sixth byte received with the Setup-Token (wIndex (H) field).
[7:0]	index[7:0]	RO	0x00	Indicates the data of the fifth byte received with the Setup-Token (wIndex (L) field).

[Description]

a. <index[15:8]>

Indicates the data of the sixth byte received with the Setup-Token (wIndex (H) field).

b. <index[7:0]>

Indicates the data of the fifth byte received with the Setup-Token (wIndex (L) field).

4. UD2WLGTH (wLength register)

Address = (0xF440_0000) + (0x021C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15:8]	length[15:8]	RO	0x00	Indicates the data of the eighth byte received with the Setup-Token (wLength (H) field).
[7:0]	length[7:0]	RO	0x00	Indicates the data of the seventh byte received with the Setup-Token (wLength (L) field).

[Description]

a. <length[15:8]>

Indicates the data of the eighth byte received with the Setup-Token (wLength (H) field).

b. <length[7:0]>

Indicates the data of the seventh byte received with the Setup-Token (wLength (L) field).

(3) Interrupt Control registers

1. UD2INT (INT register)

Address = (0xF440_0000) + (0x0220)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	m_nak	R/W	0y0	Sets whether or not to output "i_nak (bit 7)" to the INT_NAK pin. 0y0: Enable (output) 0y1: Disable (no output)
[14]	m_ep	R/W	0y0	Sets whether or not to output "i_ep (bit 6)" to the INT_EP pin. 0y0: Enable (output) 0y1: Disable (no output)
[13]	m_ep0	R/W	0y0	Sets whether or not to output "i_ep0 (bit 5)" to the INT_EP0 pin. 0y0: Enable (output) 0y1: Disable (no output)
[12]	m_sof	R/W	0y0	Sets whether or not to output "i_sof (bit 4)" to the INT_SOF pin. 0y0: Enable (output) 0y1: Disable (no output)
[11]	m_rx_data0	R/W	0y0	Sets whether or not to output "i_rx_data0 (bit 3)" to the INT_RX_ZERO pin. 0y0: Enable (output) 0y1: Disable (no output)
[10]	m_status	R/W	0y0	Sets whether or not to output "i_status (bit 2)" to the INT_STATUS pin. 0y0: Enable (output) 0y1: Disable (no output)
[9]	m_status_nak	R/W	0y0	Sets whether or not to output "i_status_nak(bit1)" to the INT_STATUS_NAK pin. 0y0: Enable (output) 0y1: Disable (no output)
[8]	m_setup	R/W	0y0	Sets whether or not to output "i_setup (bit 0)" to the INT_SETUP pin. 0y0: Enable (output) 0y1: Disable (no output)
[7]	i_nak	R/W	0y0	This will be set to 1 when NAK is transmitted by EPs except EP0.
[6]	i_ep	R/W	0y0	This will be set to 1 when transfers to EPs other than EP0 have successfully finished
[5]	i_ep0	R/W	0y0	This will be set to 1 when the transfer to EP0 has successfully finished.
[4]	i_sof	R/W	0y0	This will be set to 1 when the SOF-token is received or after 1 frame-time was counted in the create_sof mode.
[3]	i_rx_data0	R/W	0y0	This will be set to 1 when Zero-Length data is received.
[2]	i_status	R/W	0y0	This will be set to 1 when the STATUS-Stage has successfully finished in Control transfers at EP0.
[1]	i_status_nak	R/W	0y0	This will be set to 1 when “NAK” is returned during packet reception of the STATUS-Stage during Control-RD transfer to EP0.
[0]	i_setup	R/W	0y0	This will be set to 1 when the Setup-Token was received in Control transfers at EP0.

Note: The lower byte (bits 7-0) will be cleared by writing 1 to the relevant bits.

[Description]

a. <m_nak>

Sets whether or not to output "i_nak (bit 7)" to the INT_NAK pin.

0y0: Enable (output)

0y1: Disable (no output)

b. <m_ep>

Sets whether or not to output "i_ep (bit 6)" to the INT_EP pin.

0y0: Enable (output)

0y1: Disable (no output)

c. <m_ep0>

0y0: Enable (output)

0y1: Disable (no output)

d. <m_sof>

Sets whether or not to output "i_sof (bit 4)" to the INT_SOF pin.

0y0: Enable (output)

0y1: Disable (no output)

e. <m_rx_data0>

Sets whether or not to output "i_rx_data0 (bit 3)" to the INT_RX_ZERO pin.

0y0: Enable (output)

0y1: Disable (no output)

f. < m_status>

Sets whether or not to output "i_status (bit 2)" to the INT_STATUS pin.

0y0: Enable (output)

0y1: Disable (no output)

g. <m_status_nak>

Sets whether or not to output "i_status_nak(bit1)" to the INT_STATUS_NAK pin.

0y0: Enable (output)

0y1: Disable (no output)

h. <m_setup>

Sets whether or not to output "i_setup (bit 0)" to the INT_SETUP pin.

0y0: Enable (output)

0y1: Disable (no output)

i. <i_nak>

This will be set to 1 when NAK is transmitted by EPs except EP0.

(EPs to which you wish to output the INT_NAK flag can be selected using INT_NAK_MASK register). Writing 1 to this bit will make each bit of INT_NAK register cleared to 0.

j. <i_ep>

This will be set to 1 when transfers to EPs other than EP0 have successfully finished

(EPs to which you wish to output the flag can be selected using INT_EP_MASK register). Writing 1 to this bit will make each bit of INT_EP register cleared to 0.

k. < i_ep0>

This will be set to 1 when the transfer to EP0 has successfully finished.

l. < i_sof>

This will be set to 1 when the SOF-token is received or after 1 frame-time was counted in the create_sof mode.

m. <i_rx_data0>

This will be set to 1 when Zero-Length data is received. (EPs to which you wish to output the flag can be selected using INT_EP_MASK register). Writing 1 to this bit will make each bit of INT_RX_DATA0 register cleared to 0. This will not be set to 1 when Zero-Length data is received in the STATUS-Stage of Control-RD transfers.

n. <i_status>

This will be set to 1 when the STATUS-Stage has successfully finished in Control transfers at EP0. (This will be set to 1 when Zero-Length data is received in the STATUS-Stage and successfully finished in Control-RD transfers, and when Zero-Length data is transmitted in the STATUS-Stage and successfully finished in Control-WR transfers.)

o. <i_status_nak>

This will be set to 1 when the packet of STATUS-Stage is received in the Control-RD transfers at EP0. When this bit was set which means the DATA-Stage has finished, set the "Setup-Fin" command by the Command register to make the stage of UDC2 proceed to the STATUS-Stage. When receiving the data having the size of an integral multiple of MaxPacketSize (64 bytes: High-Speed) in the DATA-Stage of Control-WR transfers, Zero-Length data may be received to indicate the end of the DATA-Stage. After that, as the end of the DATA-Stage can be recognized by this i_status_nak when receiving the In-token in the STATUS-Stage, make UDC2 proceed to the STATUS-Stage.

p. <i_status_nak>

This will be set to 1 when the Setup-Token was received in Control transfers at EP0.

2. UD2INTEP (INT_EP register)

Address = (0xF440_0000) + (0x0224)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	0y0	Flags to indicate the transmitting/receiving status of EPs (except for EP0) 0y0: No data transmitted/received 0y1: Some data transmitted/received
[14]	Reserved	R/W	0y0	
[13]	Reserved	R/W	0y0	
[12]	Reserved	R/W	0y0	
[11]	Reserved	R/W	0y0	
[10]	Reserved	R/W	0y0	
[9]	Reserved	R/W	0y0	
[8]	Reserved	R/W	0y0	
[7]	Reserved	R/W	0y0	
[6]	Reserved	R/W	0y0	
[5]	Reserved	R/W	0y0	
[4]	Reserved	R/W	0y0	
[3]	i_ep3	R/W	0y0	
[2]	i_ep2	R/W	0y0	
[1]	i_ep1	R/W	0y0	
[0]	–	–	Undefined	Read as undefined. Write as zero.

Note: Will be cleared by writing 1 to the relevant bits.

[Description]

a. <i_ep[3:1]>

Flags to indicate the transmitting/receiving status of EPs (except for EP0)

The relevant bit will be set to 1 when the transfer to EPs other than EP0 has successfully finished. (EPs to which you wish to output the int_ep flag can be selected using INT_EP_MASK register.)

0y0: No data transmitted/received

0y1: Some data transmitted/received

3. UD2INTEPMSK (INT_EP_MASK register)

Address = (0xF440_0000) + (0x0228)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	0y0	Mask control of flag output 0y0: Enable (output) 0y1: Disable (no output)
[14]	Reserved	R/W	0y0	
[13]	Reserved	R/W	0y0	
[12]	Reserved	R/W	0y0	
[11]	Reserved	R/W	0y0	
[10]	Reserved	R/W	0y0	
[9]	Reserved	R/W	0y0	
[8]	Reserved	R/W	0y0	
[7]	Reserved	R/W	0y0	
[6]	Reserved	R/W	0y0	
[5]	Reserved	R/W	0y0	
[4]	Reserved	R/W	0y0	
[3]	m_ep3	R/W	0y0	
[2]	m_ep2	R/W	0y0	
[1]	m_ep1	R/W	0y0	
[0]	m_ep0	R/W	0y0	

Note: Will be cleared by writing 1 to the relevant bits.

[Description]

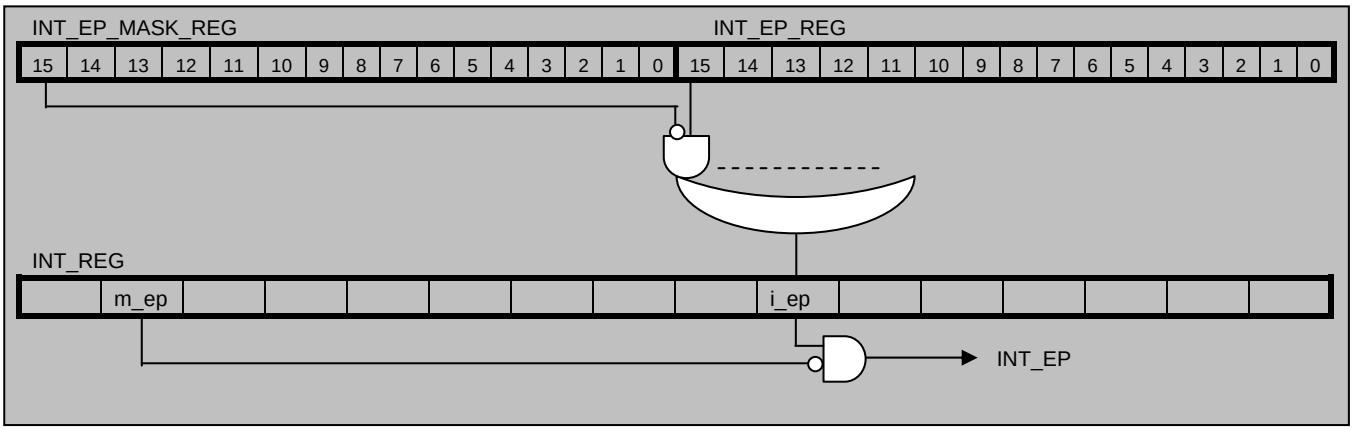
a. <m_ep[3:0]>

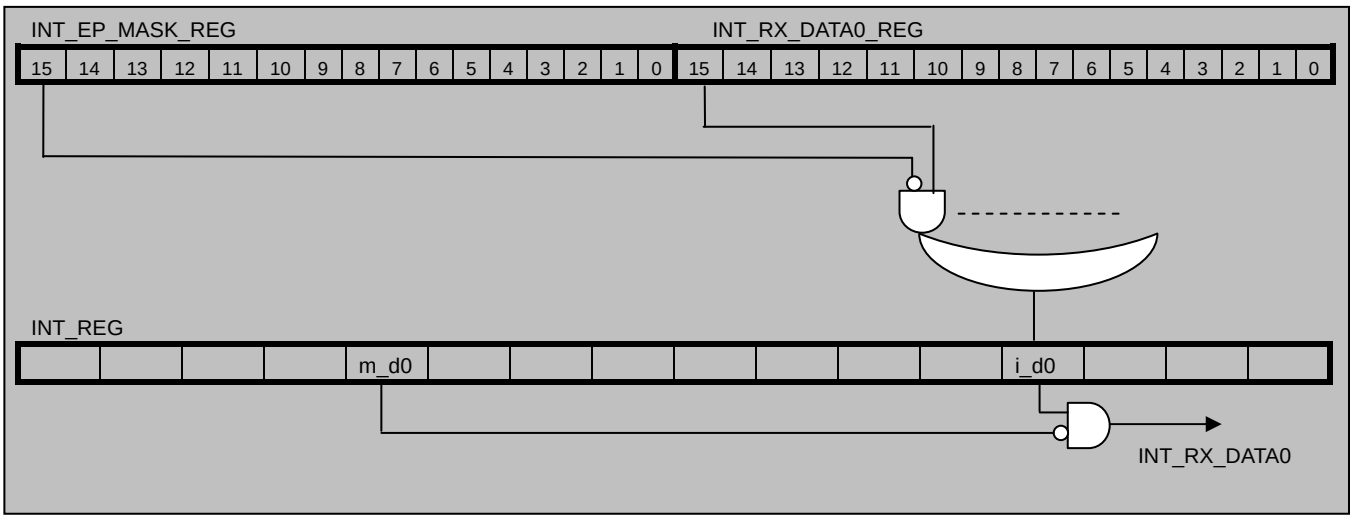
Mask control of flag output

Sets whether or not to output flags of INT_EP and INT_RX_DATA0 registers to the int_ep pin and the int_rx_zero pin respectively. When an EP is masked, each bit of INT_EP register will be set when the transfer of the relevant EP has successfully finished, but the int_ep pin will not be asserted. Similarly, when an EP is masked, each bit of INT_RX_DATA0 register will be set when Zero-Length data is received at the relevant EP, but the int_rx_zero pin will not be asserted. However, bit 0 is only valid for INT_RX_DATA0 register.

0y0: Enable (output)

0y1: Disable (no output)


INT_EP_MASK_REG
INT_EP_REG
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
INT_REG
m_ep
i_ep
INT_EP


INT_EP_MASK_REG
INT_RX_DATA0_REG
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
INT_REG
m_d0
i_d0
INT_RX_DATA0

An example of using i_ep/INT_EP/INT_EP_MASK is provided below for Endpoints 1 to 3.

1. When using Endpoint 1 and Endpoint 2 with DMA (Endpoint I/F) and using only Endpoint3 via PVCI-I/F

   After initialization, set 1 to the bits 1 and 2 of INT_EP_MASK register to mask them. Interrupt responses to EP3 will be identical whether bit 3 of INT_EP register or bit 6 of INT register is used. It may be better to use INT register alone in terms of efficiency since checking only one register will do. Use INT register for interrupt responses.

INT	bit 6:	Used as the interrupt source of EP3. This bit is also used when clearing.
	bit 14:	Used as the mask of the interrupt source of EP3.
INT_EP	bit 1:	Should be ignored.
	bit 2:	Should be ignored.
	bit 3:	Should be ignored.
INT_EP_MASK	bit 1:	Set 1 to mask the bit.
	bit 2:	Set 1 to mask the bit.
	bit 3:	Should be left as 0 without making any change.

2. When you have more than one EPx to be controlled by PVCI-I/F in addition to EP0

   The following descriptions are based on the assumption that EP2 and EP3 are controlled by PVCI-I/F, while EP1 uses DMA.

   After initialization, set 1 to INT_EP_MASK register of the EP to be used with DMA to mask it. When making interrupt responses for more than one EPs, be sure to use INT_EP register. Ignore i_ep of INT register and always enable m_ep as 0.

   Do not clear the source using i_ep of INT register. After the interrupt has occurred, you need to check INT and INT_EP registers to determine the source. When clearing the source, use each source bit of INT_EP interrupt to clear it.

INT	bit 6:	Should be ignored. Do not clear the source using this bit.
	bit 14:	Should be left as 0 without making any change.
INT_EP	bit 1:	Should be ignored.
	bit 2:	Used as the interrupt source of EP2. This bit is also used when clearing.
	bit 3:	Used as the interrupt source of EP3. This bit is also used when clearing.
INT_EP_MASK	bit 1:	Set 1 to mask the bit.
	bit 2:	Used as the mask of the interrupt source of EP2.
	bit 3:	Used as the mask of the interrupt source of EP3.

4. UD2INTRX0 (INT_RX_DATA0 register)

Address = (0xF440_0000) + (0x022C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	0y0	Flags for indicating Zero-Length data received at EP 0y0: No Zero-Length data received 0y1: Zero-Length data received
[14]	Reserved	R/W	0y0	
[13]	Reserved	R/W	0y0	
[12]	Reserved	R/W	0y0	
[11]	Reserved	R/W	0y0	
[10]	Reserved	R/W	0y0	
[9]	Reserved	R/W	0y0	
[8]	Reserved	R/W	0y0	
[7]	Reserved	R/W	0y0	
[6]	Reserved	R/W	0y0	
[5]	Reserved	R/W	0y0	
[4]	Reserved	R/W	0y0	
[3]	rx_d0_ep3	R/W	0y0	
[2]	rx_d0_ep2	R/W	0y0	
[1]	rx_d0_ep1	R/W	0y0	
[0]	rx_d0_ep0	R/W	0y0	

Note: Will be cleared by writing 1 to the relevant bits.

[Description]

a. <rx_d0_ep[3:0]>

Flags for indicating Zero-Length data received at EP

The relevant bit will be set to 1 when EPs have received Zero-Length data. (EPs to which you wish to output the int_rx_zero flag can be selected using INT_EP_MASK register.)

For bit 0 (Endpoint 0), it will be set to 1 only when Zero-Length data is received in the DATA-Stage while processing the request. Since it will not be set when Zero-Length data is received in the STATUS-Stage, use the int_status flag.

0y0: No Zero-Length data received

0y1: Zero-Length data received

5. UD2INTNAK (INT_NAK register)

Address = (0xF440_0000) + (0x0330)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	0y0	Flags to indicate the status of transmitting NAK at EPs (except for EP0) 0y0: No NAK transmitted 0y1: NAK transmitted
[14]	Reserved	R/W	0y0	
[13]	Reserved	R/W	0y0	
[12]	Reserved	R/W	0y0	
[11]	Reserved	R/W	0y0	
[10]	Reserved	R/W	0y0	
[9]	Reserved	R/W	0y0	
[8]	Reserved	R/W	0y0	
[7]	Reserved	R/W	0y0	
[6]	Reserved	R/W	0y0	
[5]	Reserved	R/W	0y0	
[4]	Reserved	R/W	0y0	
[3]	i_ep3	R/W	0y0	
[2]	i_ep2	R/W	0y0	
[1]	i_ep1	R/W	0y0	
[0]	–	–	Undefined	Read as undefined. Write as zero.

Note: Will be cleared by writing 1 to the relevant bits.

[Description]

a. <i_ep[3:1]>

Flags to indicate the status of transmitting NAK at EPs (except for EP0)

The relevant bit will be set to 1 when NAK is transmitted by EPs other than EP0. (EPs to which you wish to output the INT_NAK flag can be selected using INT_NAK_MASK register.)

0y0: No NAK transmitted

0y1: NAK transmitted

6. UD2INTNAKMSK (INT_NAK_MASK register)

Address = (0xF440_0000) + (0x0334)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	Reserved	R/W	0y0	Mask control of flag output 0y0: Enable (output) 0y1: Disable (no output)
[14]	Reserved	R/W	0y0	
[13]	Reserved	R/W	0y0	
[12]	Reserved	R/W	0y0	
[11]	Reserved	R/W	0y0	
[10]	Reserved	R/W	0y0	
[9]	Reserved	R/W	0y0	
[8]	Reserved	R/W	0y0	
[7]	Reserved	R/W	0y0	
[6]	Reserved	R/W	0y0	
[5]	Reserved	R/W	0y0	
[4]	Reserved	R/W	0y0	
[3]	m_ep3	R/W	0y0	
[2]	m_ep2	R/W	0y0	
[1]	m_ep1	R/W	0y0	
[0]	–	–	Undefined	Read as undefined. Write as zero.

Note: Will be cleared by writing 1 to the relevant bits.

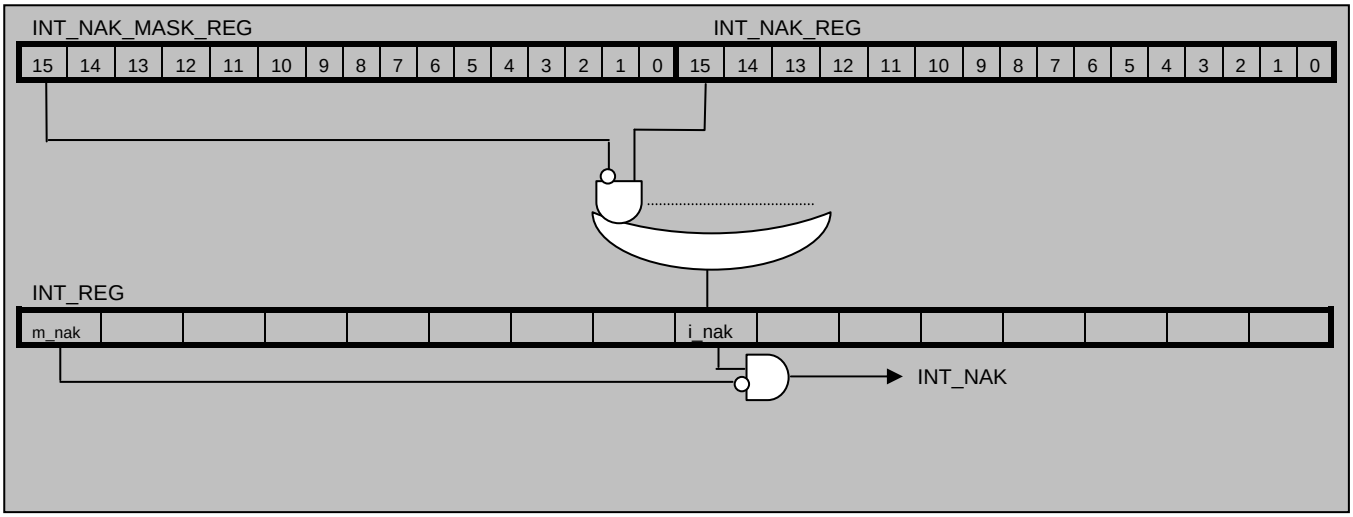
[Description]

a. <m_ep[3:1]>

Mask control of flag output

Sets whether or not to output flags of INT_NAK register to the int_nak pin respectively. When EPs are masked, each bit of INT_NAK register will be set when NAK is transmitted in the transfer of the relevant EP, but the int_nak pin will not be asserted.

0y0: Enable (output)
0y1: Disable (no output)


INT_NAK_MASK_REG
INT_NAK_REG
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
INT_REG
m_nak
i_nak
INT_NAK

(4) EP0 Control/Status Registers

1. UD2EP0MSZ (EP0_MaxPacketSize register)

Address = (0xF440_0000) + (0x0230)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	tx_0data	RO	0y0	When the "EP_TX_0DATA" command is issued to EP0 by Command register, this bit will be set to 1 which will be cleared to 0 after the Zero-Length data has been transmitted.
[14:13]	–	–	Undefined	Read as undefined. Write as zero.
[12]	dset	R/W	0y0, (-)	Indicates the status of EP0_FIFO. It will be cleared to 0 when the Setup-Token is received. 0y0: No valid data exists 0y1: Valid data exists
[11:7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	max_pkt[6:0]	R/W	0y1000000, (-)(-)(-)(-)(-)(-)(-)	Sets MaxPacketSize of EP0.

[Description]

a. <tx_0data>

When the “EP_TX_0DATA” command is issued to EP0 by Command register, this bit will be set to 1 which will be cleared to 0 after the Zero-Length data has been transmitted.

b. <dset>

Indicates the status of EP0_FIFO. It will be cleared to 0 when the Setup-Token is received.

0y0: No valid data exists

0y1: Valid data exists

c. <max_pkt[6:0]>

Sets MaxPacketSize of EP0.

2. UD2EP0STS (EP0_ Status register)

Address = (0xF440_0000) + (0x0234)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined.
[15]	ep0_mask	RO	0y0	0y0: Data can be written to EP0_FIFO. 0y1: No data can be written to EP0_FIFO.
[14]	–	–	Undefined	Read as undefined. Write as zero.
[13:12]	toggle[1:0]	RO	0y00	Indicates the present toggle value of EP0. 0y00: DATA0 0y01: DATA1 0y10: Reserved 0y11: Reserved
[11:9]	status[2:0]	RO	0y000	Indicates the present status of EP0. It will be cleared to "Ready" when the Setup-Token is received. 0y000: Ready 0y001: Busy 0y010: Error 0y011: Stall 0y100-0y111: Reserved
[8:0]	–	–	Undefined	Read as undefined.

[Description]

a. <ep0_mask>

Will be set to 1 after the Setup-Token is received. Will be cleared to 0 when the "Setup_Received" command is issued. No data will be written into the EP0_FIFO while this bit is 1.

0y0: Data can be written into EP0_FIFO

0y1: No data can be written into EP0_FIFO

b. <toggle[1:0]>

Indicates the present toggle value of EP0

0y00: DATA0

0y01: DATA1

0y10: Reserved

0y11: Reserved

c. <status[2:0]>

Indicates the present status of EP0. It will be cleared to "Ready" when the Setup-Token is received.

0y000: Ready (Indicates the status is normal)

0y001: Busy (To be set when returned "NAK" in the STATUS-Stage)

0y010: Error (To be set in case of CRC error in the received data, as well as when timeout has occurred after transmission of the data)

0y011: Stall (Returns "STALL" when data longer than the Length was requested in Control-RD transfers and the status will be set. It will be also set when "EP0-STALL" was issued by Command register.)

0y100-0y111: Reserved

3. UD2EP0DSZ (EP0_Datasize register)

Address = (0xF440_0000) + (0x0238)

Bit	Bit Symbol	Type	Reset Value	Description
[31:7]	–	–	Undefined	Read as undefined.
[6:0]	size[6:0]	RO	0y0000000, (-)(-)(-)(-)(-)(-)(-)	Indicates the number of valid data bytes stored in EP0_FIFO. It will be cleared to when the Setup-Token is received.

[Description]

a. <size[6:0]>

Indicates the number of valid data bytes stored in EP0_FIFO.

It will be cleared to when the Setup-Token is received.

4. UD2EP0FIFO (EP0_FIFO register)

Address = (0xF440_0000) + (0x023C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	data[15:0]	R/W	Undefined	Used for accessing data from PVCI-I/F to EP0.

[Description]

a. <data[15:0]>

Used for accessing data from PVCI-I/F to EP0.

For the method of accessing this register, see section 3.16.3.5 “(1) Control-RD transfer”, and section 3.16.3.5 “(3) Control-WR transfer (with DATA-Stage).”

The data stored in this register will be cleared when the request is received (when the INT_SETUP interrupt is asserted).

(5) EP1 Control/Status Registers

1. UD2EP1MSZ (EP1_MaxPacketSize register)

Address = (0xF440_0000) + (0x0240)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	tx_0data	RO	0y0	When the "EP1_TX_0DATA" command is issued to EP1 by Command register or Zero-Length data has been set at Endpoint-I/F, this bit will be set to 1. It will be cleared to 0 after the Zero-Length data has been transmitted.
[14:13]	–	–	Undefined	Read as undefined. Write as zero.
[12]	dset	RO	Note 1,Note 2	Indicates the status of EP1_FIFO. 0y0: No valid data exists 0y1: Valid data exists
[11]	–	–	Undefined	Read as undefined. Write as zero.
[10:0]	max_pkt[10:0]	R/W	0y00000000000 (-)(-)(-)(-)(-)(-) (-)(-)(-)(-)(-)	Sets MaxPacketSize of EP1. Note: See 3.16.4.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize for more information.

Note 1: The initial value of dset (bit 12) after reset_x is 1 when the EPx is a transmit endpoint, while it is 0 when the EPx is a receive endpoint.

Note 2: The initial value of dset (bit 12) after USB_RESET is 1 when the EPx is a transmit endpoint, while it is "Retain" when the EPx is a receive endpoint.

Note 3: Since the register structure is identical for all EPs through EP1 and EP3, only EP1 is described here. Addresses of registers for EP2 and EP3 can be confirmed in the register map.

Note 4: When EP3 is used in Dual, the maximum size of 1 packet is 32bytes.

[Description]

a. <tx_0data>

When the “EP1_TX_0DATA” command is issued to EP1 by Command register or Zero-Length data has been set at Endpoint-I/F, this bit will be set to 1. It will be cleared to 0 after the Zero-Length data has been transmitted.

b. <dset>

Indicates the status of EP1_FIFO.

0y0: No valid data exists

0y1: Valid data exists

c. <max_pkt[10:0]>

Sets MaxPacketSize of EP1.

Set this when configuring the endpoint when Set_Configuration and Set_Interface are received.

Set an even number for a transmit endpoint. On USB, when MaxPacketSize of a transmit endpoint is an odd number, set an even number to max_pkt and make the odd number of accesses to the endpoint. (For instance, set 1024 to max_pkt when the MaxPacketSize should be 1023 bytes.)

Note: For details, refer to section 3.16.4.2 “Appendix B About Setting an Odd Number of Bytes as MaxPacketSize”.

2. UD2EP1STS (EP1_Status register)

Address = (0xF440_0000) + (0x0244)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15]	pkt_mode	R/W	0y0	Select the packet mode of EP1. 0y0: Single mode 0y1: Dual mode
[14]	bus_sel	R/W	0y0	Select the bus to access to the FIFO of EP1. 0y0: Common bus access 0y1: Direct access
[13:12]	toggle[1:0]	RO	0y00	Indicates the present toggle value of EPx. 0y00: DATA0 0y01: DATA1 0y10: DATA2 0y11: MDATA
[11:9]	status[2:0]	RO	0y111	Indicates the present status of EP1. By issuing EP_Reset from Command register, the status will be "Ready." 0y000: Ready 0y001: Reserved 0y010: Error 0y011: Stall 0y100-0y110: Reserved 0y111: Invalid
[8]	disable	RO	0y0	Indicates whether transfers are allowed for EP1. 0y0: Allowed 0y1: Not Allowed
[7]	dir	R/W	0y0	Sets the direction of transfers for this endpoint. 0y0: OUT (Host-to-device) 0y1: IN (Device-to-host)
[6:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:2]	t_type[1:0]	R/W	0y00	Sets the transfer mode for this endpoint. 0y00: Control 0y01: Isochronous 0y10: Bulk 0y11: Interrupt
[1:0]	num_mf[1:0]	R/W	0y00	When the Isochronous transfer is selected, set how many times the transfer should be made in μ frames. 0y00: 1-transaction 0y01: 2-transaction 0y10: 3-transaction 0y11: Reserved

Note 1: Setting for this register should be made when configuring the endpoint when Set_Configuration and Set_Interface are received.

Note 2: Since the register structure is identical for EP1, EP2 and EP3, only EP1 is described here.
Addresses of registers for EP2 and EP3 can be confirmed in the register map.
Each EP depend on the produnct specification.
For EP1 which is fixed for IN transfers, dir can be set to "1" only.
For EP2 which is fixed for OUT transfers, dir can be set to "0" only.
For EP3 which is fixed for IN transfers, dir can be set to "1" only.

[Description]

a. <pkt_mode>

Selects the packet mode of EP1. Selecting the Dual mode makes it possible to retain two pieces of packet data for the EPx.

0y0: Single mode

0y1: Dual mode

b. <bus_sel>

Select the bus to access to the FIFO of EP1.

0y0: Common bus access

0y1: Direct access

c. <toggle[1:0]>

Indicates the present toggle value of EPx.

0y00: DATA0

0y01: DATA1

0y10: DATA2

0y11: MDATA

d. <status[2:0]>

Indicates the present status of EP1. By issuing EP_Reset from Command register, the status will be “Ready.”

0y000: Ready (Indicates the status is normal)

0y001: Reserved

0y010: Error (To be set in case a receive error occurred in the data packet, or when timeout has occurred after transmission. However, it will not be set when "Stall" or "Invalid" has been set.)

0y011: Stall (To be set when "EP-Stall" was issued by Command register.)

0y100-0y110: Reserved

0y111: Invalid (Indicates this endpoint is invalid)

e. <disable>

Indicates whether transfers are allowed for EP1. If "Not Allowed," "NAK" will be always returned for the Token sent to this endpoint.

0y0: Allowed

0y1: Not Allowed

f. <dir>

Sets the direction of transfers for this endpoint.

0y0: OUT (Host-to-device)

0y1: IN (Device-to-host)

Note 1: EP1 is fixed for IN transfers. Be sure to set to “1”.

Note 2: EP2 is fixed for OUT transfers. Be sure to set to “0”.

Note 3: EP3 is fixed for IN transfers. Be sure to set to “1”.

g. <t_type[1:0]>

Sets the transfer mode for this endpoint.

0y00: Control

0y01: Isochronous

0y10: Bulk

0y11: Interrupt

h. <num_mf[1:0]>

When the Isochronous transfer is selected, set how many times the transfer should be made in μ frames.

0y00: 1-transaction

0y01: 2-transaction

0y10: 3-transaction

0y11: Reserved

3. UD2EP1DSZ (EP1_Datasize register)

Address = (0xF440_0000) + (0x0248)

Bit	Bit Symbol	Type	Reset Value	Description
[31:11]	–	–	Undefined	Read as undefined.
[10:0]	size[10:0]	RO	0y00000000000, (-)(-)(-)(-)(-)(-) (-)(-)(-)(-)(-)	Indicates the number of valid data bytes stored in EP1_FIFO. In the Dual Packet mode, the number of data bytes to be accessed first will be shown.

Note: Since the register structure is identical for EP1, EP2 and EP3 only EP1 is described here.
Addresses of registers for EP2 and EP3 can be confirmed in the register map.

[Description]

a. <size[10:0]>

Indicates the number of valid data bytes stored in EP1_FIFO. In the Dual Packet mode, the number of data bytes to be accessed first will be shown.

4. UD2EP1FIFO (EP1_FIFO register)

Address = (0xF440_0000) + (0x024C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:0]	data[15:0]	–	Undefined	Used for accessing data from PVCI-I/F to EPx.

Note: Since the register structure is identical for EP1, EP2 and EP3, only EP1 is described here.
Addresses of registers for EP2 and EP3 can be confirmed in the register map.

[Description]

a. <data[15:0]>

Used for accessing data from PVCI-I/F to EPx.

3.16.3.4 USB Device Response

UDC2 initializes the inside of UDC2 and sets various registers when hardware reset is detected, USB_RESET is detected, and an enumeration response is made. This section discusses the operations of UDC2 in each status as well as how to control them externally.

(1) When hardware reset is detected

Be sure to reset hardware for UDC2 after the power-on operation. After the hardware reset, UDC2 initializes internal registers and all endpoints are in the invalid status, which means the device itself is “Disconnected.”

In order to make the status of UDC2 to “Default,” issue the “USB_Ready”" command. Issuing this command will put UDC2 in the "Full-Speed" mode, enable the Pull-Up resistance of DDP and notify the host of “Connect.”

In this status, only the USB_RESET signal is accepted from the host.

(2) When USB_RESET is detected

UDC2 initializes internal registers when Bus Reset (USB_RESET) is detected on the USB signal, putting the device in the “Default” status. In this status only Endpoint 0 gets “Ready” enabling enumeration with the host.

The mode of UDC2 will be “HS-Chirp” and Chirp operation with the host will start. When Chirp from the host is normally received, the mode of UDC2 turns to High-Speed (HS) and subsequent transfers between the hosts will be made in the HS mode. If Chirp from the host is not received, subsequent transfers between the hosts will be made in the Full-Speed (FS) mode.

The current transfer mode can be judged by reading the bits[13:12] of Address-state register.

(3) When “Set_address” request is received

By setting 0y010 to the bits[10:8] and the received address value to the bits[6:0] of Address-state register after receiving the “Set_address” request, UDC2 will be in the “Addressed” status. Setting for this register should be made after the Control transfer has successfully finished (after the STATUS-Stage has ended).

Transfers to endpoints other than Endpoint 0 cannot be made in this status.

(4) When “Set_configuration” and “Set_interface” requests are received

By setting 0y100 to the bits[10:8] of Address-state register after receiving the “Set_configuration” and “Set_interface” requests, UDC2 will be in the “Configured” status.

In the “Configured” status, you can make transfers to the endpoint to which status settings have been made.

In order to make the endpoint “Ready,” the following settings should be made:

- Set the maximum packet size to EPx_MaxPacketSize register
- Set the transfer mode to EPx_Status register
- Issue the EP_Reset command to Command register

Endpoints will be available for transmitting and receiving data after these settings have been made.

Figure 3.16.18 shows the “Device State Diagram”.

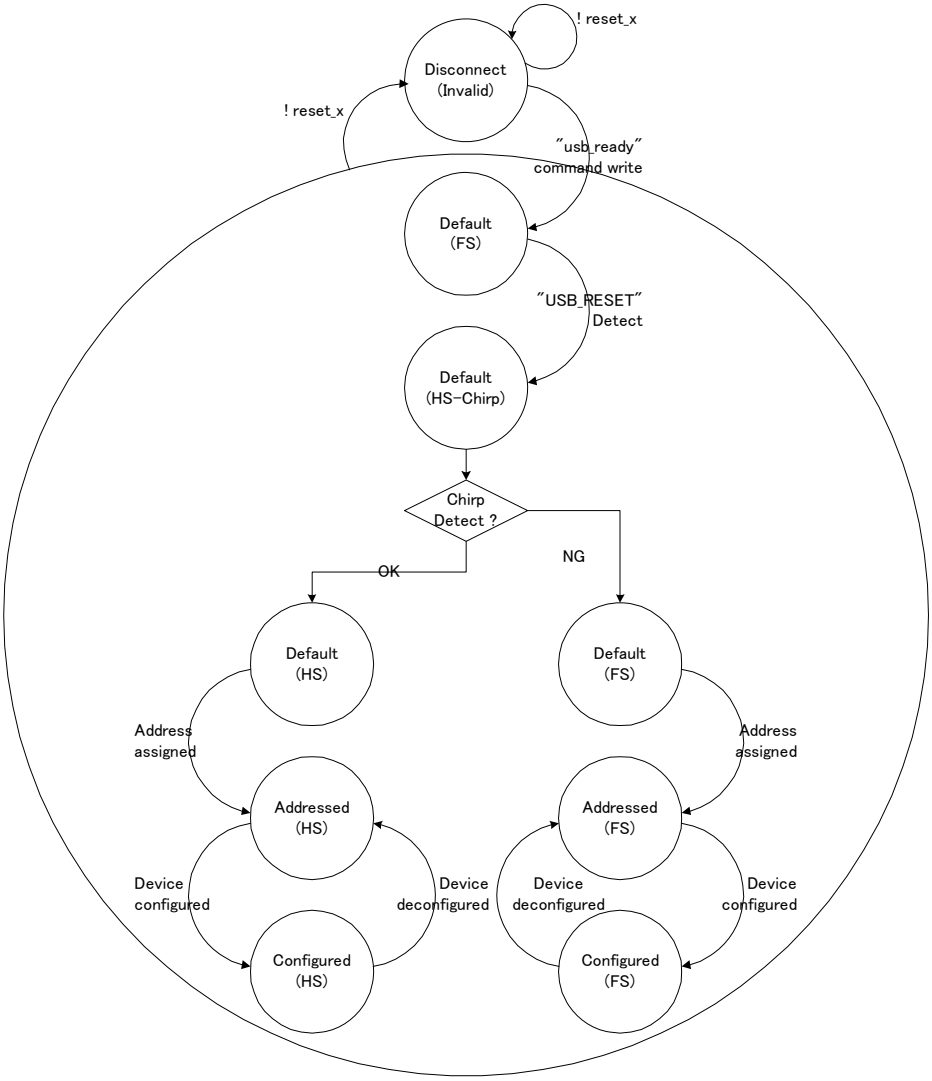


Figure 3.16.18 Device state diagram

3.16.3.5 Flow of Control in Transfers of Endpoints

- Endpoint 0

Endpoint 0 supports Control transfers and used to make enumeration and other device control operations. Please note that Endpoint 0 can be used in the single packet mode only.

Control transfers consist of the following three stages:

SETUP-Stage DATA-Stage STATUS-Stage

The types of transfer are categorized into the following major types:

- Control-RD transfer
- Control-WR transfer (without DATA-Stage)
- Control-WR transfer (with DATA-Stage)

UDC2 makes control of those three stages by hardware. Flows in each type of transfer are described below.

(1) Control-RD transfer

The flow of control in Control-RD transfers is shown below.

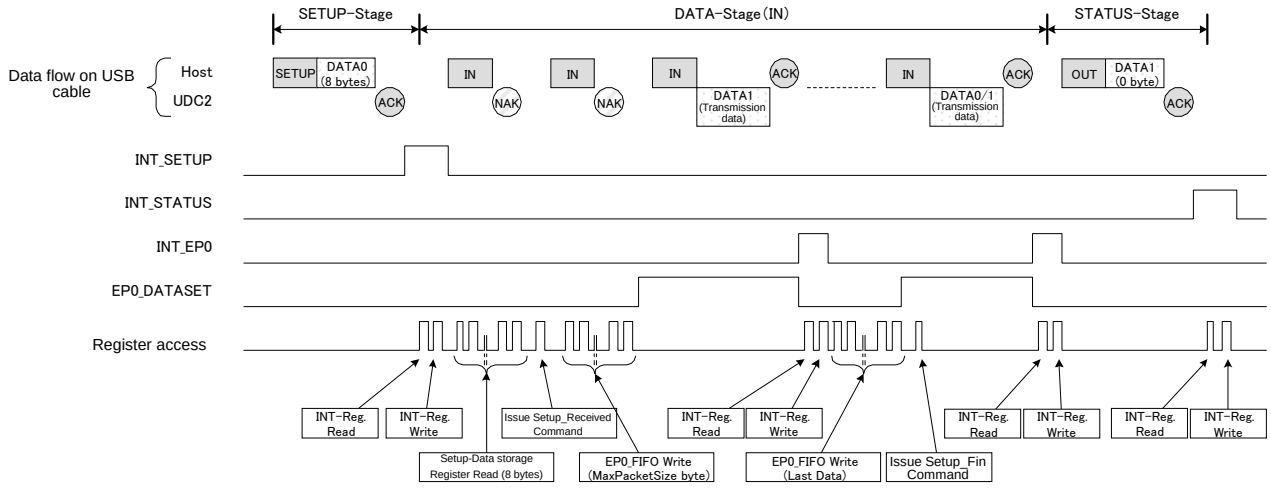


Figure 3.16.19 Flow of control in Control-RD transfer

The following description is based on the assumption that the bit 12 (dset) of EP0_MaxPacketSize register is set to “EP0_DATASET flag”.

1. SETUP-Stage

UDC2 asserts the INT_SETUP flag when it has received the Setup-Token. This flag can be cleared by writing 1 into the bit 0 (i_setup) of INT register. In case flags are combined externally, read the INT register to confirm which flag is asserted and write “1” into the relevant bit.

Then read Setup-Data storage registers (bRequest-bmRequestType, wValue, wIndex, and wLength registers) to determine the request.

Finally, issue the “Setup_Received” command to inform UDC2 that the SETUP-Stage has finished. Since UDC2 does not allow writing data into the Endpoint0-FIFO before this command is issued, it will keep returning "NAK" to the IN-Token from the host until the command is issued.

2. DATA-Stage

Write the data to be transmitted to the IN-Token into the Endpoint0-FIFO. If the byte size of the data to send is larger than the MaxPacketSize, divide them into groups of MaxPacketSize before writing. When the number of data reached the MaxPacketSize, the EP0_DATASET flag is asserted.

When the data have been transmitted to the IN-Token from the host with no problem, UDC2 deasserts the EP0_DATASET flag and asserts INT_EP0. Any data remaining to be transmitted should be written into the Endpoint0-FIFO.

If the size of the data to be written is smaller than the MaxPacketSize, issue the “EP_EOP” command to EP0 to inform UDC2 that it is a short packet. With this command, UDC2 recognizes the end of the packet and transmits the short packet data.

Finally, issue the “Setup_Fin” command to inform UDC2 that the DATA-Stage has finished.

3. STATUS-Stage

When the “Setup_Fin” command is issued, UDC2 will automatically make Handshake for the STATUS-Stage. When the STATUS-Stage finished with no problem, the INT_STATUS flag is asserted. When received a packet of STATUS-Stage from the host before the “Setup_Fin” command is issued, UDC2 will return “NAK” and asserts the INT_STATUS_NAK flag. Therefore, if this flag is asserted, be sure to issue the “Setup_Fin” command.

(2) Control-WR transfer (without DATA-Stage)

The flow of control in Control-WR transfer (without DATA-Stage) is shown below.

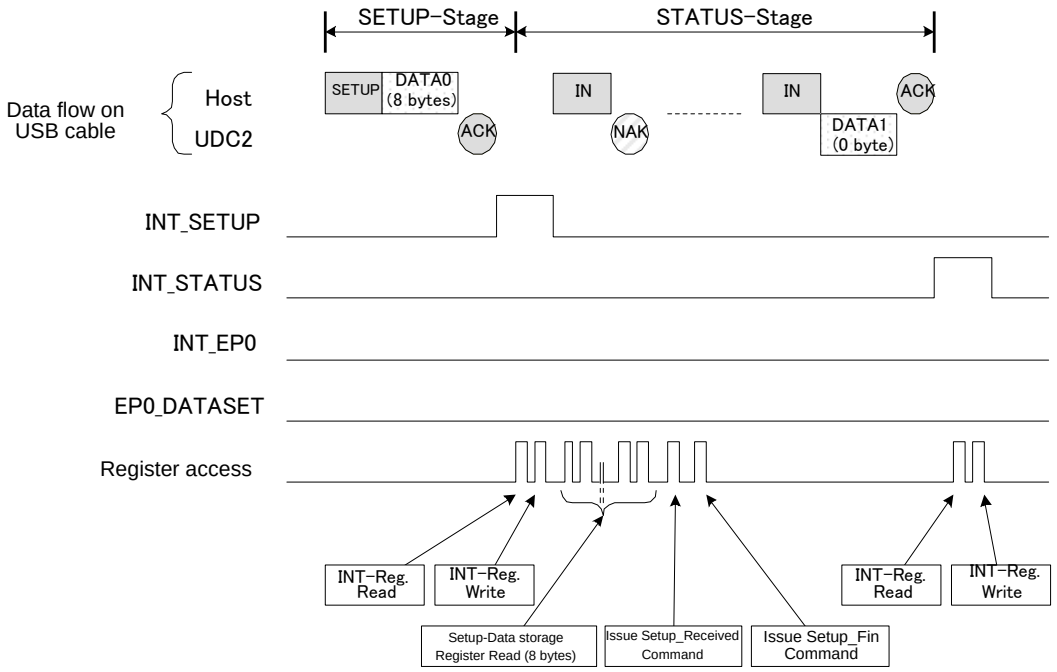


Figure 3.16.20 Flow of control in Control-WR transfers (without DATA-Stage)

1. SETUP-Stage

   To be processed in the same way as in the SETUP-Stage described in (1).

2. STATUS-Stage

   After issuing the “Setup_Received” command, make register accesses to UDC2 based on each request. Issue the “Setup_Fin” command when all the register accesses to UDC2 have finished. Subsequent processes are basically the same as the STATUS-Stage described in (1). UDC2 will keep on returning “NAK” until the “Setup_Fin” command is issued.

Note: While register accesses required for each request are made to UDC2 between 'Issuing the "Setup_Received" command' and 'Issuing the "Setup_Fin" command', register accesses are needed after the end of STATUS-Stage in some cases such as Set Address request and Set Feature (TEST_MODE). Processes required for the standard requests are described in (5).

(3) Control-WR transfer (with DATA-Stage)

The flow of control in Control-WR transfer (with DATA-Stage) is shown below.

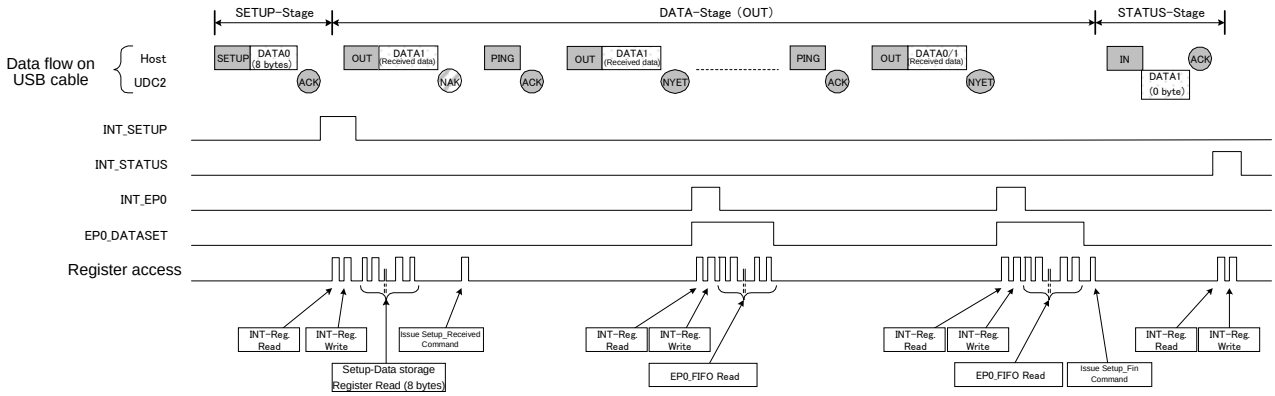


Figure 3.16.21 Flow of control in Control-WR transfers (with DATA-Stage)

1. SETUP-Stage

   To be processed in the same way as in the SETUP-Stage described in (1).

2. DATA-Stage

   When the data is received from the host with no problem, UDC2 asserts the EP0_DATASET flag and asserts the INT_EP0 flag. When this flag is asserted, read the data from EP0_FIFO after confirming the received data size in the EP0_Datasize register, or read the data from EP0_FIFO polling the EP0_DATASET flag.

   When the byte size of received data has been read, UDC2 deasserts the EP0_DATASET flag.

3. STATUS-Stage

   To be processed in the same way as in the STATUS-Stage described in (1).

Note: Figure 3.16.21 shows the flow in High-Speed transfers. In Full-Speed transfers, the "PING" packet shown in the figure is not issued. Also, the "NYET" packet is replaced by the "ACK" packet.

(4) Example of using the INT_STATUS_NAK flag

When processing requests without DATA-Stage, the INT_STATUS_NAK flag may get asserted by receiving STATUS-Stage from the host before clearing the INT_SETUP flag after it has been asserted, especially in High-Speed transfers. In case such multiple interrupts should be avoided as much as possible, you can use a method to mask the INT_STATUS_NAK flag for request having no DATA-Stage. In such case, basically set 1 to "m_status_nak" of INT register, while 0 should be set only when requests having DATA-Stage are received. (An example for Control-RD transfers is provided below.)

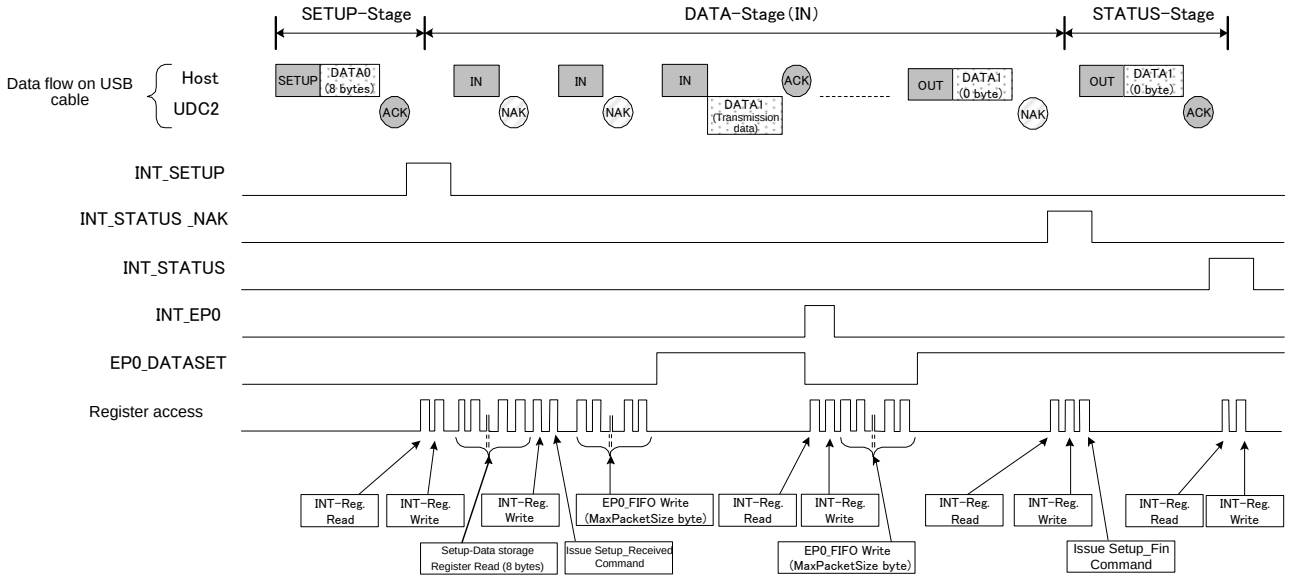


Figure 3.16.22 Example of using the INT_STATUS_NAK flag in Control-RD transfers

1. SETUP-Stage

After the INT_SETUP flag was asserted, clear the bit 0 (i_setup) of INT register. If the bit 1 (i_status_nak) is set to 1, it should be also cleared.

Then, if the request was judged to have DATA-Stage by reading Setup-Data storage registers, set the bit 9 (m_status_nak) of INT register to 0. Then issue the "Setup_Received" command.

2. DATA-Stage → STATUS-Stage

When the INT_STATUS_NAK flag was asserted, the device should also proceed to the STATUS-Stage. Clear the bit 1 (i_status_nak) of INT register and then issue the "Setup_Fin" command. Also, set 1 to the bit 9 (m_status_nak) of INT register in order to get ready for subsequent transfers.

(5) Processing when standard requests are received

Examples of making register accesses to UDC when standard requests are received are provided below. Descriptions of each request are basically provided for each state of the device (Default, Address, and Configured).

For the information on register accesses common to each request, see (1), (2) and (3).

You should note, however, descriptions provided below do not include the entire details of standard requests in USB 2.0. Since methods to access registers may vary depending on each user's usage, be sure to refer to the USB 2.0 specifications. You should also refer to the USB 2.0 specifications for “Recipient,” “Descriptor Types,” “Standard Feature Selectors,” “Test Mode Selectors” and other terms appear in the descriptions below.

- Standard requests for “ (1) Control-RD transfers”
  - • Get Status • Get Descriptor • Get Configuration
  - • Get Interface • Synch Frame
- Standard requests for “ (2) Control-WR transfer (without DATA-Stage) “
  - • Clear Feature • Set Feature • Set Address
  - • Set Configuration • Set Interface
- Standard requests for “ (3) Control-WR transfer (with DATA-Stage)”
  - • Set Descriptor

Note 1: Descriptions with double underlines refer to register accessed to UDC2.

Note 2: Writing accesses to Command register are described in the following manner for simplicity:

(Example 1) When writing 0x0 to bits 7-4 (ep) and 0x4 to bits 3-0 (com) of Command register

→ Issue the EP-Stall command to EP0

(Example 2) When writing the relevant endpoint to bits 7-4 (ep) and 0x5 to bits 3-0 (com) of Command register

→ Issue the EP-Invalid command to the relevant endpoint

(a) Get Status request

To this request, the status of the specified recipient is returned.

BmRequestType	BRequest	wValue	wIndex	wLength	Data
0y10000000 0y10000001 0y10000010	GET_STATUS	Zero	Zero Interface Endpoint	Two	Device, Interface, or Endpoint Status

- Common to all states:

  If the Endpoint/Interface specified by wIndex does not exist, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:
  - Recipient = Device : Write the information on the device (Figure 3.16.23) to EP0_FIFO register.
  - Recipient = Interface : Issue the EP-Stall command to EP0.
  - Recipient = Endpoint : If wIndex = 0 (EP0), write the information on Endpoint 0 (Figure 3.16.25) to EP0_FIFO register.
    If wIndex ≠ 0 (EPx), issue the EP-Stall command to EP0.

- Configured state:
  - Recipient = Device : Write the information on the device (Figure 3.16.23) to EP0_FIFO register.
  - Recipient = Interface : If the interface specified by wIndex exists, write the information on the interface (Figure 3.16.24) to EP0_FIFO register.
  - Recipient = Endpoint : If the endpoint specified by wIndex exists, write the information on the relevant endpoint (Figure 3.16.25) to EP0_FIFO register.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote Wakeup	Self Powered
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

Figure 3.16.23 Information on the device to be returned by Get Status request

・SelfPowered (D0) : 0 indicates the bus power while 1 indicates the self power.

・RemoteWakeup (D1) : 0 indicates the remote wakeup function is disabled while 1 indicates it is enabled.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

Figure 3.16.24 Information on the interface to be returned by Get Status request

- Please note that all bits are 0.

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	Halt
D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

Figure 3.16.25 Information on the endpoint to be returned by Get Status request

- Halt (D0): If this bit is 1, it indicates that the relevant endpoint is in the “Halt” state.

(b) Clear Feature request

To this request, particular functions are cleared or disabled.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y00000000 0y00000001 0y00000010	CLEAR_FEATURE	Feature Selector	Zero Interface Endpoint	Zero	None

- Common to all states:

  If Feature Selector (wValue) which cannot be cleared (disabled) or does not exist is specified, issue the EP-Stall command to EP0.

  If the Endpoint/Interface specified by wIndex does not exist, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  - Recipient = Device : If wValue = 1, disable the DEVICE_REMOTE_WAKEUP function at the user's end. No register access to UDC2 is required.
  - Recipient = Interface : Issue the EP-Stall command to EP0.
  - Recipient = Endpoint : If wIndex ≠ 0 (EPx), issue the EP-Stall command to EP0.
    If wValue = 0 and wIndex = 0 (EP0), clear the Halt state of Endpoint 0 but no register access to UDC2 is required.

- Configured state:

  - Recipient = Device : If wValue = 1, disable the DEVICE_REMOTE_WAKEUP function at the user's end. No register access to UDC2 is required.
  - Recipient = Interface : Issue the EP-Stall command to EP0. (Note)
  - Recipient = Endpoint : If wValue = 0 and wIndex ≠ 0 (EPx), issue the EP-Reset command to relevant endpoint. If wValue = 0 and wIndex = 0 (EP0), clear the Halt state of Endpoint 0 but no register access to UDC2 is required.

Note: Endpoint 0 is to be stalled based on the interpretation of the USB 2.0 specifications that "No Feature Selector exists for Interface" here. For more information, see the USB Specification.

(c) Set Feature request

To this request, particular functions are set or enabled.

bmRequestType	bRequest	wValue	wIndex		wLength	Data
0y00000000 0y00000001 0y00000010	SET_FEATURE	Feature Selector	Zero Interface Endpoint	Test Selector	Zero	None

- Common to all states:

When Recipient = Device and wValue = 2 are specified in a device supporting High-Speed, write the value of Test Selector (upper byte of wIndex) to the bits7-0(t_sel) of USB-testmode register within 3 ms after the STATUS-Stage has ended.

If, however, an invalid value (other than test_j, test_k, se0_nak, and test_packet) is specified for the Test Selector value, issue the EP-Stall command to EP0.

Note: When using a vendor-specific Test Selector other than standard ones, the appropriate operation should be made.

- If Feature Selector (wValue) which cannot be set (enabled) or does not exist is specified, issue the EP-Stall command to EP0.
- If the Endpoint/Interface specified by the lower byte of wIndex does not exist, issue the EP-Stall command to EP0.

- Default state:

Nothing is specified for the operation of devices by the USB 2.0 specifications except for the above-mentioned TEST_MODE.

- Address state:
  - Recipient = Device : If wValue = 1, enable the DEVICE_REMOTE_WAKEUP function at the user's end. No register access to UDC2 is required.
  - Recipient = Interface : Issue the EP-Stall command to EP0.
  - Recipient = Endpoint : If the lower byte of wIndex ≠ 0 (EPx), issue the EP-Stall command to EP0.
    If wValue = 0 and the lower byte of wIndex = 0 (EP0), make Endpoint 0 halt. (Note 2)

- Configured state:
  - Recipient = Device : If wValue = 1, enable the DEVICE_REMOTE_WAKEUP function at the user's end. No register access to UDC2 is required.
  - Recipient = Interface : Issue the EP-Stall command to EP0. (Note 1)
  - Recipient = Endpoint : If wValue = 0 and the lower byte of wIndex ≠ 0 (EPx), issue the EP-Stall command to the relevant endpoint. If wValue = 0 and the lower byte of wIndex = 0 (EP0), make Endpoint 0 halt. (Note 2)

Note 1: Endpoint 0 is to be stalled based on the interpretation of the USB specifications that "No Feature Selector exists for Interface" here. For more information, see the USB specifications.

Note 2: USB 2.0 specifications include such description that "Performing the Halt function for Endpoint 0 is neither necessary nor recommended." Accordingly, it can be interpreted that it is not necessary to set UDC2 to the Stall state in this case.

In order to actually make Endpoint 0 be in the Halt state, users have to manage the " Halt state."

Then, when a request is received in the "Halt state", such processes as to issue the EP-Stall command to EP0 in DATA-Stage/STATUS-Stage will be required. (Even if Endpoint0 is set to the Stall state, UDC2 will cancel the Stall state when the Setup-Token is received and will return "ACK.")

As such, the process when SetFeature/ClearFeature is received for Endpoint 0 varies depending on user's usage.

(d) Set Address request

To this request, device addresses are set.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y00000000	SET_ADDRESS	Device Address	Zero	Zero	None

For this request, make register accesses shown below within 2 ms after the STATUS-Stage has ended.

(The device address should not be changed before the Setup_Fin command is issued.)

- Default state:

  When wValue = 0

  Keeps the default state. No register access to UDC2 is required.

  When wValue ≠ 0

  Set wValue to bits 6-0 (dev_adr) and 0y010 to bits 10-8 (Device_State) of Address-State register. UDC2 will be put in the address state.

- Address state:

  When wValue = 0

  Set 0x00 to bits 6-0 (dev_adr) and 0y001 to bits 10-8 (Device_State) of Address-State register. UDC2 will be put in the Default state.

  When wValue ≠ 0

  Set wValue to bits 6-0 (dev_adr) of Address-State register.

  UDC2 will be set to the new device address.

- Configured state: Nothing is specified for the operation of devices by the USB 2.0 specification.

(e) Get Descriptor request

To this request, the specified descriptor is returned.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y10000000	GET_DESCRIPTOR	Descriptor Type and Descriptor Index	Zero or Language ID	Descriptor Length	Descriptor

- Common to all states:
  - Write the descriptor information specified by wValue to EP0_FIFO register for the byte size specified by wLength. If the byte size to write is larger than the MaxPacketSize of Endpoint 0, you need to divide the data to write it several times (see (1) Control-RD transfer for more information). (If the length of the descriptor is longer than wLength, write the information for wLength bytes from the beginning of the descriptor. If the length of the descriptor is shorter than wLength, write the full information for the descriptor.)
  - If the descriptor specified by wValue is not supported by the user, issue the EP-Stall command to EP0.

(f) Set Descriptor request

To this request, the descriptor is updated or added.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y00000000	SET_DESCRIPTOR	Descriptor Type and Descriptor Index	Language ID or zero	Descriptor Length	Descriptor

- Common to all states:

  When this request is not supported, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state & Configured state:

  Read the information on the descriptor received by UDC2 from EP0_FIFO register.

(g) Get Configuration request

To this request, the Configuration value of the current device is returned.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y10000000	GET_CONFIGURATION	Zero	Zero	One	Configuration Value

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  Write 0x00 to EP0_FIFO register [7:0]. As this is not configured, 0 should be returned.

- Configured state:

  Write the current Configuration value to EP0_FIFO register [7:0].

  Since this has been configured, values other than 0 should be returned.

(h) Set Configuration request

To this request, device configuration is set.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y00000000	SET_CONFIGURATION	Configuration Value	Zero	Zero	None

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  When wValue = 0

  ・ Keeps the address state. No register access to UDC2 is required.

  When wValue ≠ 0 and the wValue is a Configuration value matching the descriptor

  ・ Set 0y100 to bits 10-8 (Device_State) of Address-State register.

  <For endpoints to use>

  ・ Set MaxPacketSize to bit10-0(max_pkt) of EPx_MaxPacketSize register.

  ・ Set respective values to bit 15 (pkt_mode), bit 14 (bus_sel), bit 7 (dir), bits 3-2 (t_type), and bits 1-0 (num_mf) of EPx_Status register.

  ・ Issue the EP-Reset command to the relevant endpoints.

  When wValue ≠ 0 and the wValue is a Configuration value not matching the descriptor

  ・ Issue the EP-Stall command to EP0.

- Configured state:

  When wValue = 0

  ・ Set 0y010 to bits 10-8 (Device_State) of Address-State register.

  ・ Issue the All-EP-Invalid command.

  When wValue ≠ 0 and it is a Configuration value matching the descriptor

  <For endpoints to use>

  ・ Set MaxPacketSize to bits 10-0 (max_pkt) of EPx_MaxPacketSize register.

  ・ Set respective values to bit 15 (pkt_mode), bit 14 (bus_sel), bit 7 (dir), bits 3-2 (t_type), and bits 1-0 (num_mf) of EPx_Status register.

  ・ Issue the EP-Reset command to the relevant endpoints.

  <For endpoints to become unused>

  ・ Issue the EP-Invalid command to the relevant endpoints.

  When wValue ≠ 0 and the wValue is a Configuration value not matching the descriptor

  ・ Issue the EP-Stall command to EP0.

(i) Get Interface request

To this request, the AlternateSetting value set by the specified interface is returned.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y10000001	GET_INTERFACE	Zero	Interface	One	Alternate Setting

- Common to all states:

  If the interface specified by wIndex does not exist, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  Issue the EP-Stall command to EP0.

- Configured state:

  Write the current Alternate Setting value of the interface specified by wIndex to EP0_FIFO register [7:0].

(j) Set Interface request

To this request, the Alternate Setting value of the specified interface is set.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y00000001	SET_INTERFACE	Alternate Setting	Interface	Zero	None

- Common to all states:

  If the interface specified by wIndex does not exist or the Alternate Setting specified by wValue does not exist, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  Issue the EP-Stall command to EP0.

- Configured state:

  <For the endpoints to use in Alternate Setting of the specified interface>

  - Set MaxPacketSize to bits 10-0 (max_pkt) of EPx_MaxPacketSize register.
  - Set respective values to bit 15 (pkt_mode), bit 14 (bus_sel), bit 7 (dir), bits 3-2 (t_type), and bits 1-0 (num_mf) of EPx_Status register.
  - Issue the EP-Reset command to the relevant endpoints.

  <For endpoints to become unused>

  - Issue the EP-Invalid command to the relevant endpoints.

(k) Synch Frame request

To this request, the Synch Frame of the endpoint is returned.

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0y10000010	SYNCH_FRAME	Zero	Endpoint	Two	Frame Number

- Common to all states:

  If this request is not supported by the endpoint specified by wIndex, issue the EP-Stall command to EP0.

- Default state:

  Nothing is specified for the operation of devices by the USB 2.0 specifications.

- Address state:

  Issue the EP-Stall command to EP0.

- Configured state:

  Write the Frame Number of the endpoint specified by wIndex to EP0_FIFO register.

- Endpoints other than Endpoint 0

  Endpoints other than Endpoint 0 support Bulk (send/receive), Interrupt (send/receive), and Isochronous (send/receive) transfers and are used to transmit and receive data. They also support the Dual Packet mode which enables high-speed data communication.

3.16.3.6 Suspend/Resume States

UDC2 enters into a suspended state based on the signal condition from the host. It also returns from the suspended state by resuming operation by the host or UDC2.

Shifting between the states is described below.

(1) Shift to the suspended state

Though the host issues SOF with given intervals (HS: 125 μs, FS: 1 ms) in the normal state, it will stop issuing this SOF to the device when it tries to make the device suspended and the data on the USB signal line will be unchanged keeping the idle state. UDC2 is always monitoring the "line_state" from PHY and makes judgment of whether it is in the suspended state or USB_RESET when the idle state is detected for 3 ms or longer. If judged to be in the suspended state, it will assert "suspend_x" to "L" and enter in the suspended state.

Please note **accesses to registers will be unavailable** while UDC2 is suspended, since supply of CLK from PHY will stop.

(2) Resuming from suspended state

Resuming from the suspended state can be made in two ways; by outputting a resuming state from the host and by way of remote wakeup from UDC2 (outputting a resuming state).

Resuming process in each case is described below.

(3) Resuming by an output from the host

When a resuming state is output by the host, UDC2 deasserts "suspend_x" to "H" to declare resuming from the suspended state.

(4) Resuming by way of remote wakeup from UDC2

The remote wakeup function may not be supported by some applications, and it needs to be permitted by the USB host at the time of bus enumeration. You should not assert "wakeup" unless permitted by the system.

If permitted by the system, asserting the "wakeup" pin will make UDC2 output a resuming state to the host to start resuming. Please note that the clock supply from PHY is stopped when UDC2 is suspended, so you should keep asserting wakeup until it resumes. The remote wakeup should be operated after 2 ms or more has passed after suspend_x was asserted to "L".

3.16.4 USB-Spec2.0 Device Controller Appendix

3.16.4.1 Appendix A System Power Management

In USB, operations related to the enumeration and power control signals (DDP/DDM signals) for reset and suspend from the host are also prescribed, in addition to normal transfer operations. This Appendix provides information about the specifications of USB 2.0 PHY to be connected and clock control on the system level required for processes related to the DDP/DDM signals. For details of each process, please be sure to check the USB Specification Revision 2.0, PHY Specification, and the UTMI Specification Version 1.05.

*1) Reset: The operation of the DDP/DDM signals for initializing the USB device (hereafter called “the device”) from the USB host (hereafter called “the host”). After reset, enumeration is performed and then normal transfer operations such as Bulk transfers begin. Upon being connected, the device is always reset. The device also needs to support reset operation at any other arbitrary timing. During the reset period, Chirp operation is performed to determine whether the device operates in High-Speed (HS) or Full-Speed (FS) mode.

*2) Suspend: If no bus activity on the DDP/DDM lines including SOF is initiated by the host for 3 ms or longer, the device needs to be put in the suspend mode to reduce power consumption. In this case, the device is required to perform certain operations such as stopping the clock.

*3) Resume: The operation of the DDP/DDM signals for resuming normal operation from the suspend mode. Resume operation can be initiated either by the host or the device. Resume operation from the device is called “remote wakeup”.

1. Connect/Disconnect Operations

(1) Connect Operation

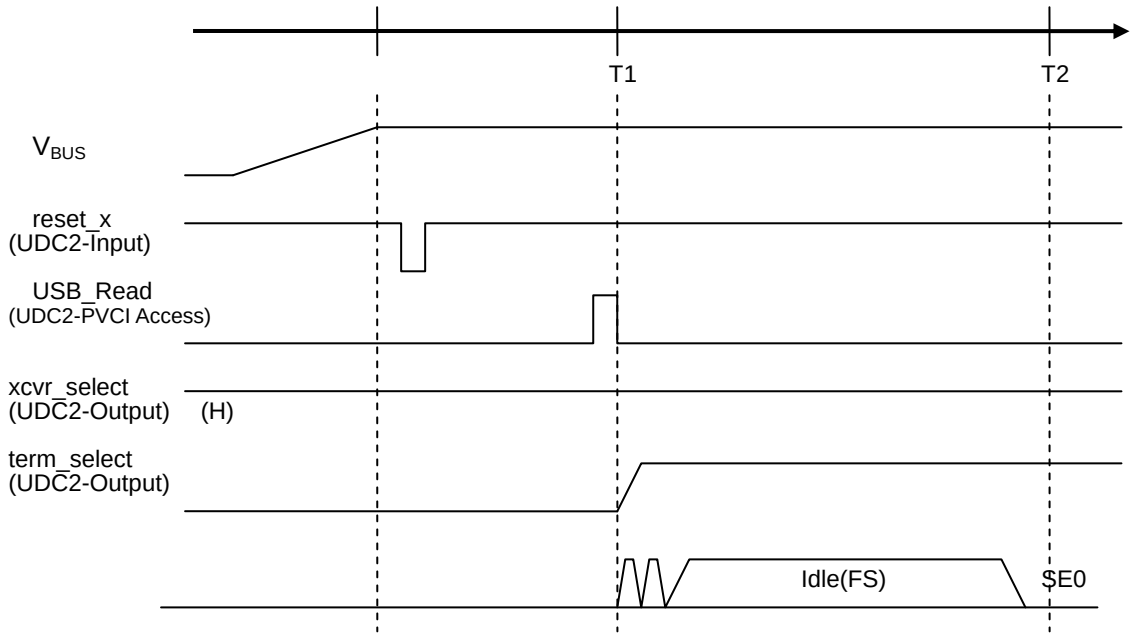


Figure 3.16.26 Connect operation timing

- T0: Vbus detection

  When Vbus is detected, a system reset (reset_x input) should be applied to UDC2.

  xcvr_select is “H” and term_select is “L”.

- T1: Device connect (no later than 100 ms after T0) *4) Based on the USB 2.0 Specification.

  The device must enable DDP pull-up no later than 100 ms after Vbus detection (T0) to notify the host of the connected state. Therefore, when Vbus is detected and the device is ready to communicate with the host, the system should access the Command register in UDC2 to set the USB_Ready command. When USB_Ready is accepted, UDC2 drives term_select “H”. This makes USB 2.0 PHY enable DDP pull-up.

- T2: USB reset start (more than 100 ms after T1)

(2) Disconnect Operation

When a disconnected state is detected, it is recommended to apply a system reset to UDC2.

2. Reset Operation *1

* The “reset” here refers to the “Reset Signaling” defined in the USB 2.0 Specification, not the system reset (reset_x) to UDC2.

(1) When Operating in HS Mode after Reset

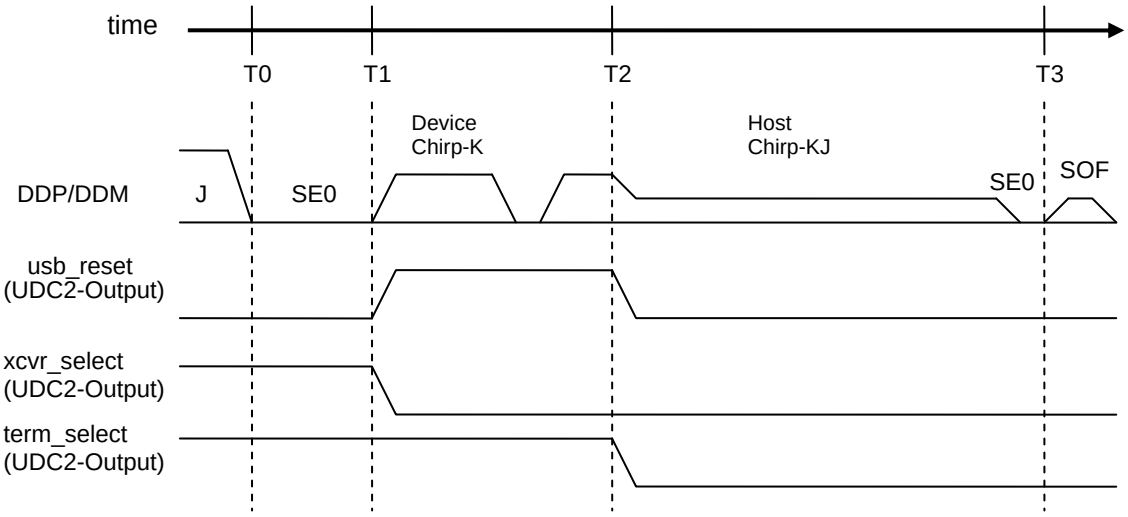


Figure 3.16.27 Reset operation timing (HS mode after Chirp)

- T0: Reset start

  Upon recognizing SE0 from the host, UDC2 starts counting to recognize the reset.

- T1: Reset recognition (more than 2.5 μs after T0)

  When UDC2 detects SE0 for more than approximately 68 μs after T0, it recognizes the reset from the host and drives usb_reset “H”. At the same time, UTMI starts the device Chirp-K operation.

- T2 : HS operation start (approximately 1.74 ms to 2 ms after T1)

  When the host supports HS mode, UDC2 detects Chirp-KJ from the host and drives usb_reset “L” within 1.74 ms to 2 ms after T1. (The period in which rsb_reset remains “H” depends on the host.) From this point, UDC2 operates in HS mode.

- T3 : Reset end (more than 10 ms after T0)

  After completion of Chirp-KJ from the host, SE0 and the packet (SOF) is transmitted. This is the end of reset operation. The reset period from the host lasts a minimum of 10 ms.

(2) When Operating in FS Mode after Reset

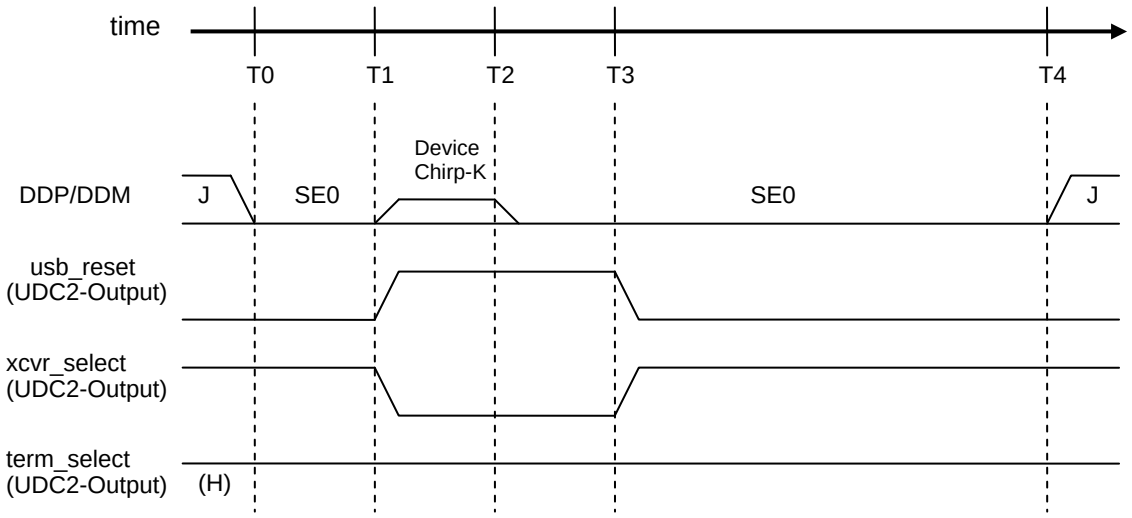


Figure 3.16.28 Reset operation timing (FS mode after Chirp)

- T0: Reset start

  Upon recognizing SE0 from the host, UDC2 starts counting to recognize the reset.

- T1: Reset recognition (more than 2.5 μs after T0)

  When UDC2 detects SE0 for more than approximately 68 μs after T0, it recognizes the reset from the host and drives usb_reset "H". At the same time, UTMI starts the device Chirp-K operation.

- T2: Device Chirp-K complete (more than 1.0 ms after T1)

  UDC2 completes the device Chirp-K operation approximately 1.5 ms after T1.

- T3: FS operation start (1.0 ms to 2.5 ms after T2)

  When the host supports FS mode, the host chirp-KJ operation is not performed. If no host Chirp-KJ is detected in approximately 2 ms after T2, UDC2 initiates FS mode. At this point, usb_reset is driven "L". The period in which usb_reset remains "H" is approximately 3.5 ms.

- T4: Reset end (more than 10 ms after T0)

  When SE0 from the host finishes and the device enters an idle state, it indicates the end of reset operation. The reset period from the host lasts a minimum of 10 ms.

(3) Notes on Reset Operation

- Initialization of registers after reset

  When the reset from the host is completed (when usb_reset changes from "H" to "L"), all the internal registers of UDC2 are initialized. (For the initial value of each register, refer to the Section 3.16.3.3 Registers.)

  Note that registers that are set while usb_reset is "H" are also initialized. Therefore, the UDC2 registers should be set after the reset period is completed.

- DMA transfer (Endpoint-I/F access) after reset

  When a reset from the host occurs during DMA transfer, the EPx_Status register is initialized and the bus access mode is set to "common bus access". Therefore, DMA transfer cannot be continued properly. When a reset occurs, the DMA controller must also be initialized.

  In the enumeration operation after reset, configure each endpoint and then initialize the endpoints by setting the EP_Reset command in the Command register.

3. Suspend Operation *2

(1) Suspend Operation in HS Mode

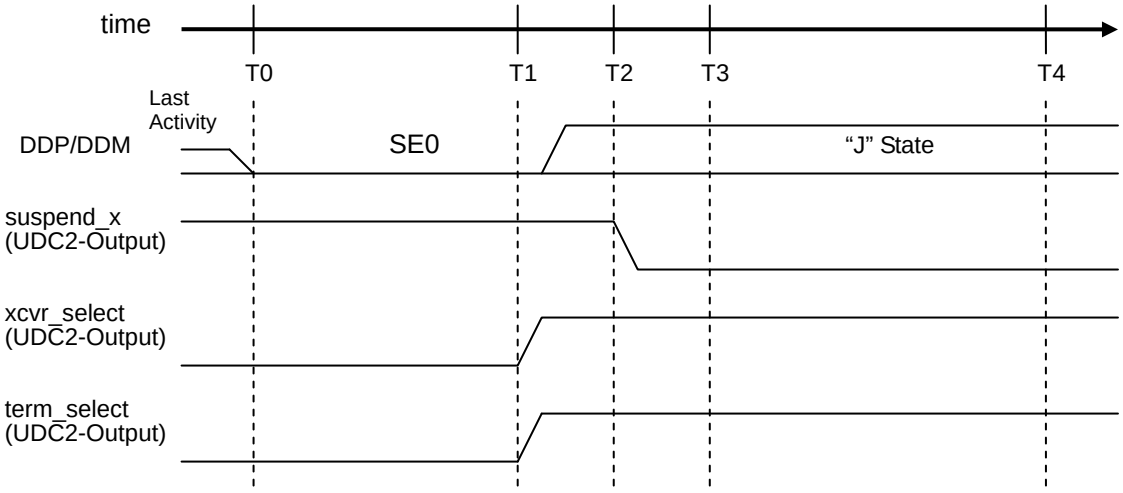


Figure 3.16.29 Suspend operation timing in HS mode

- T0: End of bus activity

  When the end of bus activity from the host (the end of packet) is detected, UDC2 starts counting to recognize suspend/reset.

- T1: Transition to FS mode (3.0 ms to 3.125 ms after T0)

  When SE0 is detected for more than 3 ms after T0, UDC2 enters FS mode and drives xcvr_select and term_select "H". (This makes USB 2.0 PHY enable DDP pull-up.) At this point, UDC2 cannot determine whether the host is initiating suspend or reset operation.

- T2: Recognition of suspend (100 µs to 875 µs after T1)

  When the "J" state is detected on the DDP/DDM line in approximately 110 µs after T1, UDC2 recognizes suspend and drives suspend_x "L". When the line state does not change to "J" and remains "SE0", UDC2 prepares for reset instead of suspend. In this case, refer to "2. Reset Operation".

- T3: Remote wakeup start enable (5 ms after T0)

  Resume operation from the device (remote wakeup) is enabled 5 ms after T0. For details, refer to section "4.(2) Resume Operation by the Device (Remote Wakeup)".

- T4: Transition to suspend state (10 ms after T0)

  The device must enter the suspend state no later than 10 ms after T0. Processes required of the device system to enter the suspend state, such as stopping the clock supply from USB 2.0 PHY, must be performed during this period.

(2) Suspend Operation in FS Mode

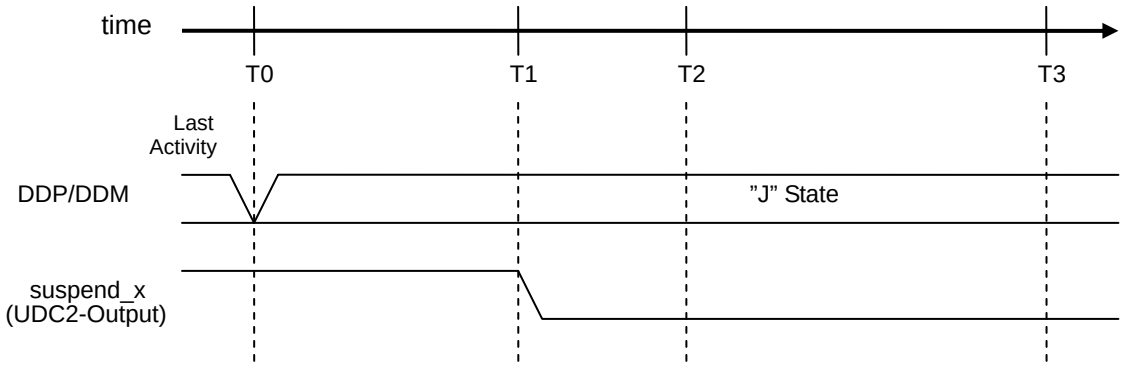


Figure 3.16.30 Suspend operation timing in FS mode

- T0: End of bus activity

  When the end of bus activity from the host (the end of packet) is detected, UDC2 starts counting to recognize suspend.

- T1: Recognition of suspend (3 ms after T0)

  When the "FS-J" is detected for more than 3 ms after T0, UDC2 recognizes suspend and drives suspend_x "L".

- T2: Remote wakeup start enable (5 ms after T0)

  Resume operation from the device (remote wakeup) is enabled 5 ms after T0. For details, refer to "4(2) Resume Operation by the Device (Remote Wakeup)"

- T3: Transition to suspend state (10 ms after T0)

  The device must enter the suspend state no later than 10 ms after T0. Processes required of the device system to enter the suspend state, such as stopping the clock supply from USB 2.0 PHY, must be performed during this period.

(3) Notes on Suspend Operation

- Clock control in suspend operation

  When the SuspendM input (UTMI) to USB 2.0 PHY is enabled at suspend, the clock supply from PHY to UDC2 is stopped. If UDC2 needs to use the clock from PHY after suspend_x becomes "L", supend_x should not be directly connected to PHY. The SuspendM input to PHY should be enabled after the system determines that the clock supply from PHY can be stopped.

  (When the clock input (30 MHz) to UDC2 is stopped, the internal registers of UDC2 cannot be accessed across PVCI-I/F and Endpoint-I/F.)

- USB 2.0 PHY clock control

  The SuspendM input (UTMI) to USB 2.0 PHY should not be directly connected to the suspend_x output of UDC2. It should be controlled by the system. As explained earlier, suspend_x of UDC2 is activated by communication with the USB host. Therefore, if the USB host is not connected, suspend_x retains the hardware reset value of "H". At this time, if suspend_x and SuspendM are directly connected, the clock supply from USB 2.0 PHY is not stopped and system power consumption cannot be saved.

- Internal registers during the suspend state

  During the suspend state, UDC2 retains the internal register values, the contents of FIFOs, and the state of each flag. These values and states are also retained after the suspend state is exited by resume operation.

4. Resume Operation *3

(1) Resume Operation by the Host

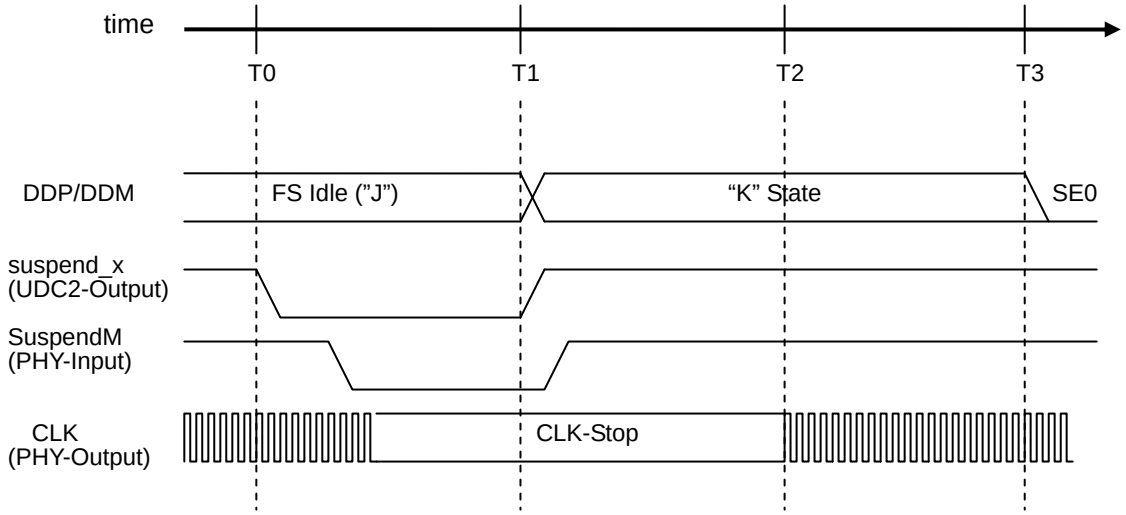


Figure 3.16.31 Resume operation timing by the host

- T0: suspend_x output of UDC2 = “L”

- T1: Start of host resume (No timing specifications)

  The host starts resume operation (“FS-K”) at arbitrary timing to wake up the device from the suspend state. At this point, UDC2 sets suspend_x to “H”. (Even if the clock input to UDC2 is stopped, suspend_x becomes “H”.) After checking that suspend_x = “H”, disable the SuspendM (UTMI) input to PHY to resume the clock output from USB 2.0 PHY.

- T2: Resuming of clock supply from USB 2.0 PHY (Depends on the PHY specifications.)

- T3: End of host resume (more than 20 ms after T1)

  The host resume operation (“FS-K”) lasts for more than 20 ms, and completes after “SE0”. UDC2 resumes operating at the same speed (HS/FS) as before the suspend state was entered.

(2) Resume Operation by the Device (Remote Wakeup)

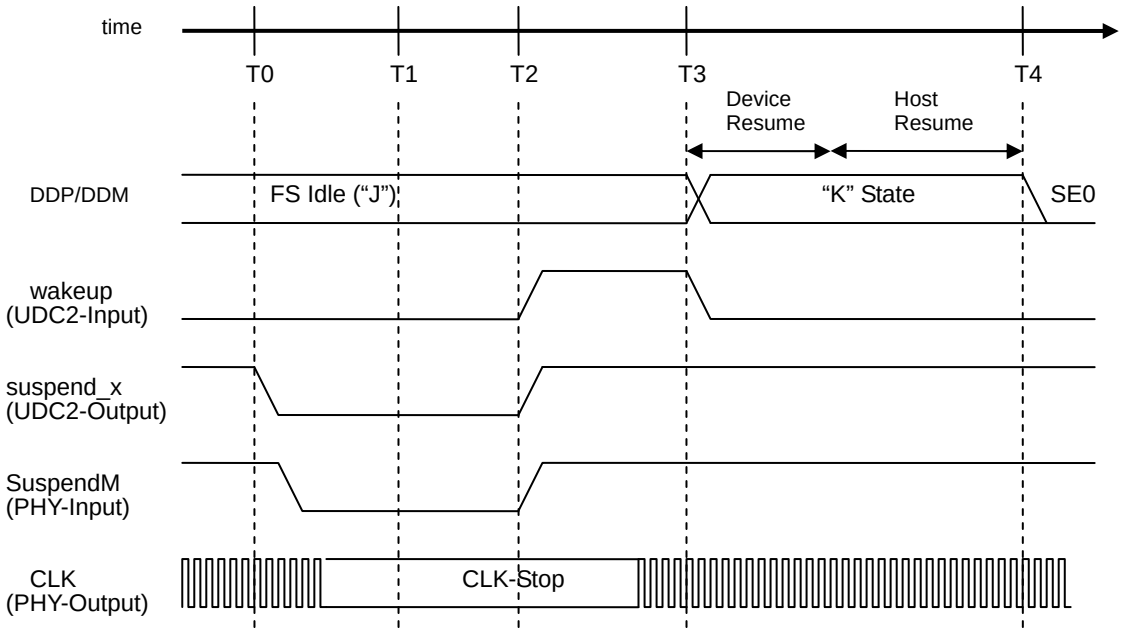


Figure 3.16.32 Remote wakeup operation timing

- T0: suspend_x output of UDC2 = "L"

- T1: Remote wakeup start enable (more than 2 ms after T0)

  The device can be brought out of the suspend state by using the wakeup input of UDC2. This is called remote wakeup. Note that the USB specification prohibits remote wakeup for 5 ms after start of the suspend state. The wakeup signal should be set to "H" a minimum of 2 ms after T0 as 3 ms have already elapsed from the start of suspend operation to T0.

- T2: Wakeup input to UDC2 = "H" (after T1)

  Set the wakeup signal to "H". No timing requirements are specified for this operation. At this point, UDC2 sets suspend_x to "H". (Even if the clock input to UDC2 is stopped, suspend_x becomes "H".) Because UDC2 requires the clock input to start resume operation ("FS-K"), the SuspendM (UTMI) input to USB 2.0 PHY should be disabled. Then, keep wakeup at "H" until clock supply is resumed.

- T3: Start of device resume (Depends on the PHY specifications.)

  When the clock input from PHY to UDC2 is resumed, UDC2 starts the device resume ("FS2-K"). The device resume period is approximately 2 ms. After confirming the device resume, the host starts the host resume operation.

- T4: End of host resume (more than 20 ms after T3)

  The host resume operation ("FS-K") lasts for more than 20 ms, and completes after "SE0". UDC2 resumes operating at the same speed (HS/FS) as before the suspend state was entered.

(3) Notes on Resume Operation

- Restriction on use of remote wakeup

  To support remote wakeup as the device system, the device must notify the host in the Configuration descriptor that the remote wakeup function is enabled. Even if remote wakeup is supported, it is disabled by default. Remote wakeup can only be used after it is enabled by a request from the host. Use of remote wakeup using the wakeup input is allowed only when these conditions are satisfied.

  When using this function, be sure to refer to 3.16.3.6 of the USB 2.0 Specification which offers detailed description.

3.16.4.2 Appendix B About Setting an Odd Number of Bytes as MaxPacketSize

1. Setting an odd number in the EPx_MaxPacketSize register

The USB specification allows MaxPacketSize (hereafter referred to as MPS) of each endpoint to be set as either an odd or even number of bytes for Isochronous and Interrupt transfers. (For Control and Bulk transfers, only an even number can be set.)

In UDC2, MPS is set through max_pkt (bits[10:0]) of the EPx_MaxPacketSize register. The endpoint FIFOs of UDC2 only support even numbers of bytes. It is therefore recommended that MSP be set as an even number of bytes as a general rule.

When using MPS by odd bytes, it is possible to make “max_pkt“ into odd number. However, there are restrictions shown in Table 3.16.3 by the access method of a bus. In the case of endpoint direct access, an odd number cannot be set in max_pkt for a transmit endpoint. In this case, an even number should be set in max_pkt and write accesses to the endpoint FIFO should be controlled to implement an odd number of maximum write bytes. (For example, when MPS = 1023 bytes, max_pkt should be set to 1024 bytes.)

Table 3.16.3 Restrictions on the setting of max_pkt

	Receive endpoint	Transmit endpoint
Common bus access (PVCI-I/F)	An odd or even number can be set.	An odd or even number can be set.
Endpoint direct access (Endpoint-I/F)	An odd or even number can be set.	Only an even number can be set.

Based on the above, the following pages describe how to set an odd number of bytes as MPS for each bus access method.

2. Receive endpoint & common bus access

Either an odd or even number of bytes can be set in max_pkt. The access method is the same for both cases.

3. Transmit endpoint & common bus access

Either an odd or even number of bytes can be set in max_pkt.

However, the following points must be observed in making common bus accesses for writing the maximum number of bytes with max_pkt = odd number.

The following shows an example in which max_pkt = 5 and the maximum number of bytes (5 bytes) are to be written.

- In the last access (5th byte), make sure that udc_be = 0y01.
- Because it is access of MPS, Do not issue the EP-EOP command in the Command register.

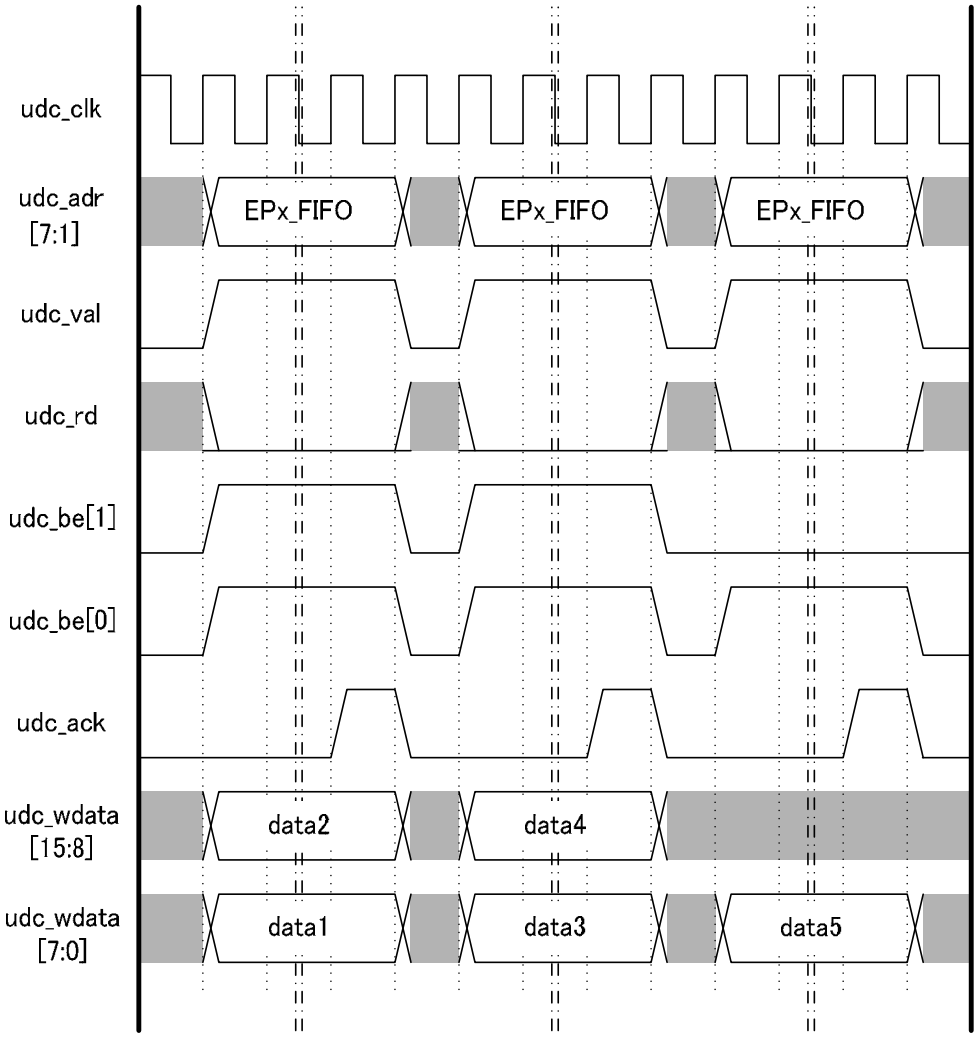


Figure 3.16.33 MPS write access with max_pkt = odd number (common bus access)

4. Receive endpoint & endpoint direct access

Either an odd or even number can be set in max_pkt. The access method is the same for both cases.

5. Transmit endpoint & endpoint direct access

Only an even number of bytes can be set in max_pkt. To use an odd number of bytes as MPS for a transmit endpoint, the following settings are required.

(Example: MPS = 1023)

- Set max_pkt = 1024.
- The maximum number of bytes that can be written to the endpoint is 1023 bytes. (It is not allowed to write the 1024th byte.)
- "wMaxPacketSize" of the Endpoint descriptor to be managed by firmware should be set to 1023. (This is the value to be sent to the USB host by the Get Descriptor request.)

The following shows an example in which max_pkt = 1024 and the maximum number of bytes (1023 bytes) are to be written.

- In the last access (1023rd byte), make sure that epx_w_be = 0y01.

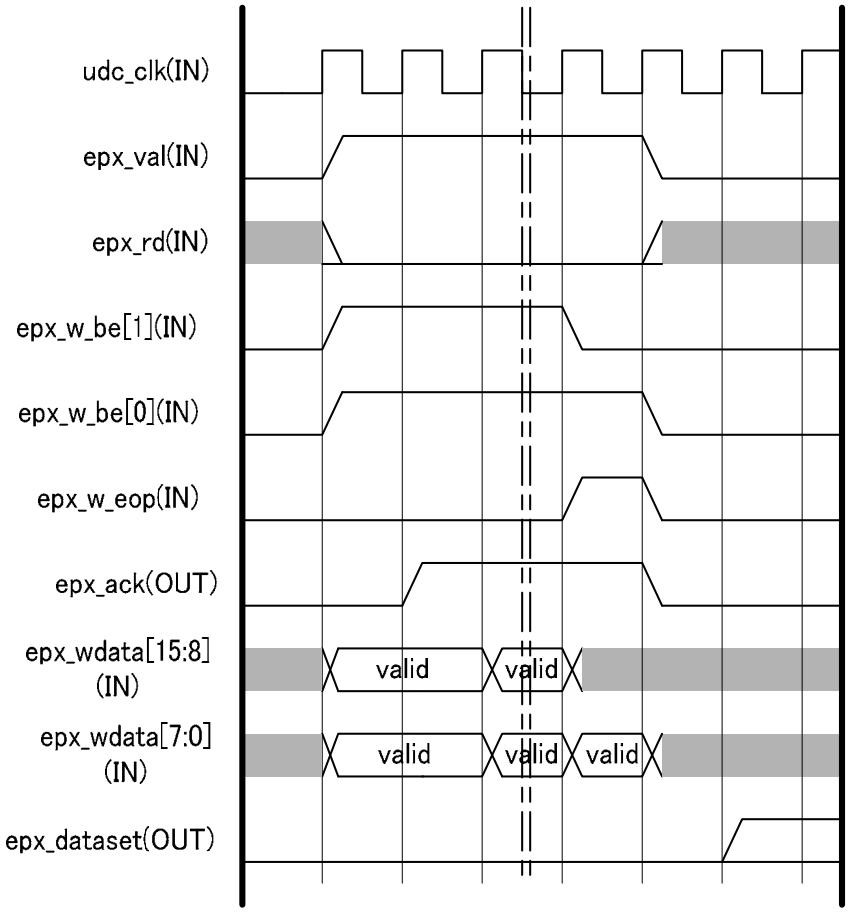


Figure 3.16.34 MPS (odd number) write access with max_pkt = even number (endpoint direct access)

3.17 I²S (Inter-IC Sound)

The TMPA901CM contains a serial input/output circuit compliant with the I²S format.

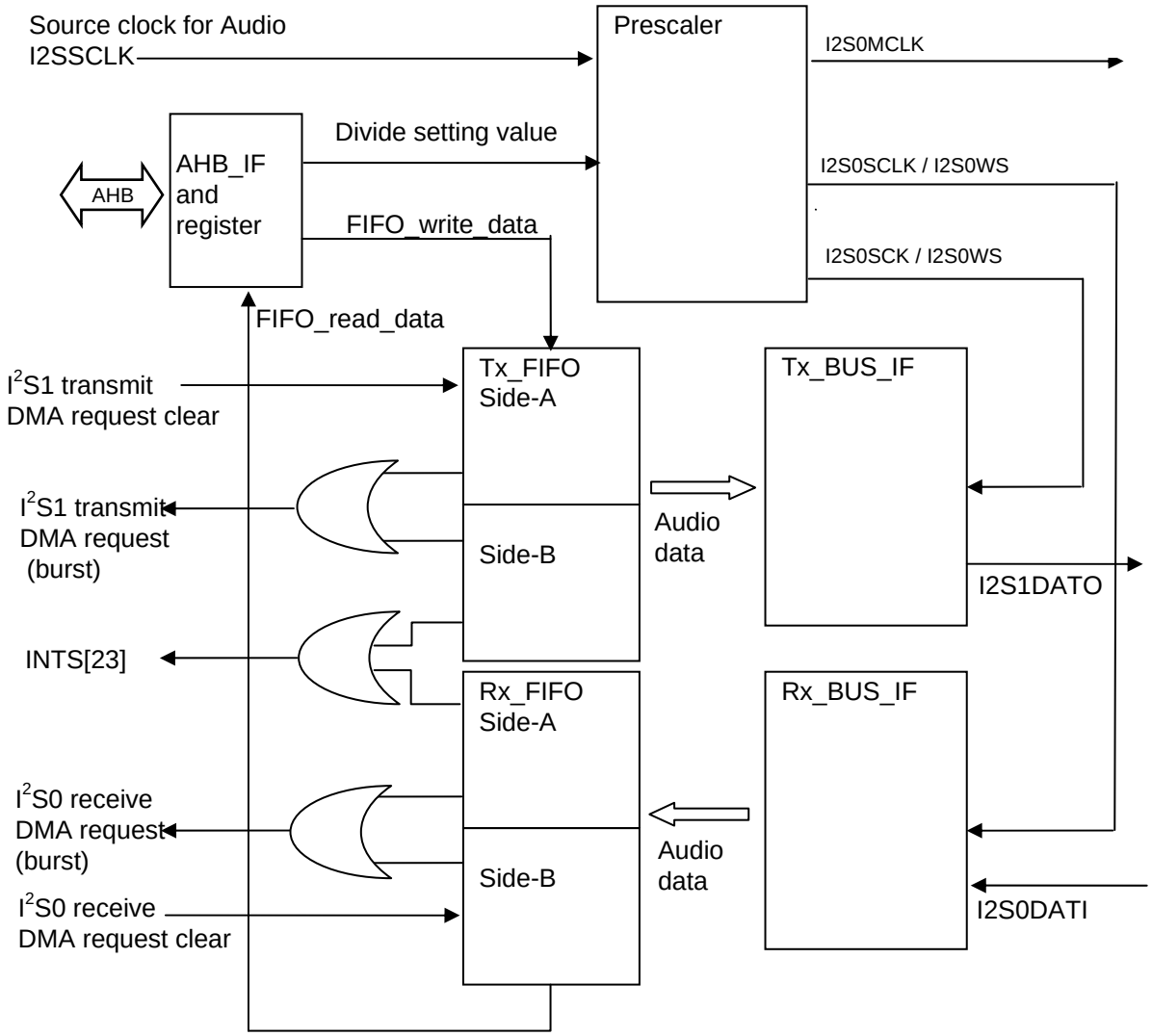
By connecting an external audio LSI, such as an AD converter or DA converter, the I²S interface can support the implementation of a digital audio system.

The I²S of this LSI has the following characteristics:

Table 3.17.1 I²S operation characteristics

Modes	Full-duplex master mode Full-duplex slave mode Clock through mode	
Channel	Channel 0	Channel 1
Transmit/Receive	Receive only	Transmit only
Data formats	(1) I²S format-compliant (2) Stereo/monaural (3) MSB first/LSB first selectable (4) Left-justified supported (synchronous to WS, no delay)	
Pins used	(1) I2S0SCLK (clock input/output) (2) I2S0DATI (data input) (3) I2S0WS (word select input/output) (4) I2S0MCLK (master clock output)	(1) I2S1DATO (data output)
Clocks	(1) I2SWS can be set to 1/256, 1/384 or 1/512 of the master clock. (2) the internal clock (X1) can be used as the source clock. (3) The master clock can be generated by dividing down the source clock to 1, 1/2 or 1/4.	
FIFO buffer	2 × 8 words	2 × 8 words
Data length	16 bits only	16 bits only
Interrupts	FIFO overflow interrupt FIFO underflow interrupt	FIFO overflow interrupt FIFO underflow interrupt

3.17.1 Block diagram



3.17.2 Operation Mode Descriptions

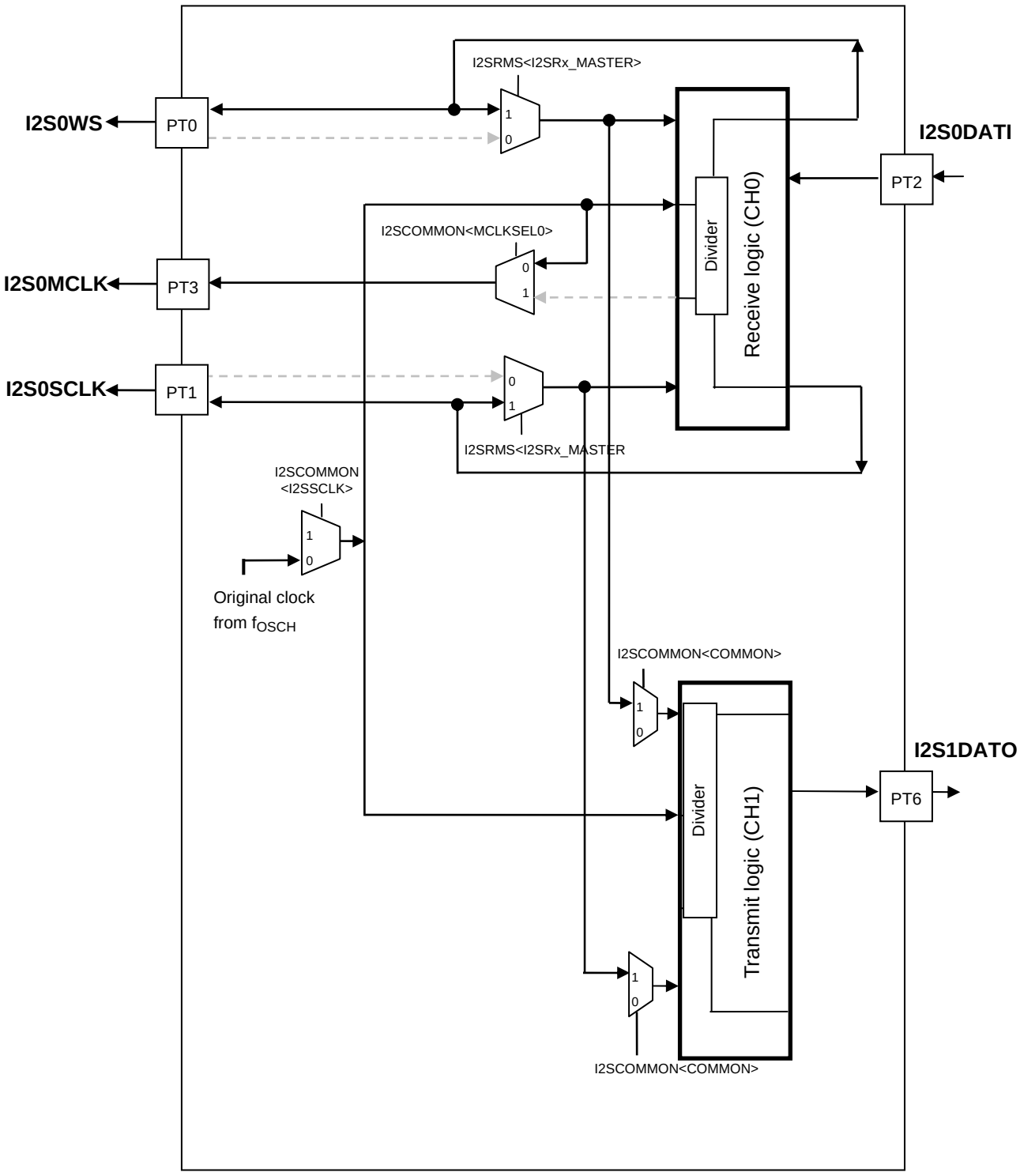
The I^2S circuit can operate simultaneously both receive and transmit communication by using the full-duplex master mode and the full-duplex slave mode.

The following pages explain the I^2S operation modes.

3.17.2.1 Mode Example 1

(Full-duplex Master, <I2SSCLK> = 0, <MCLKSEL0> = 0, <COMMON> = 1, <I2SRx_MASTER> = 1)

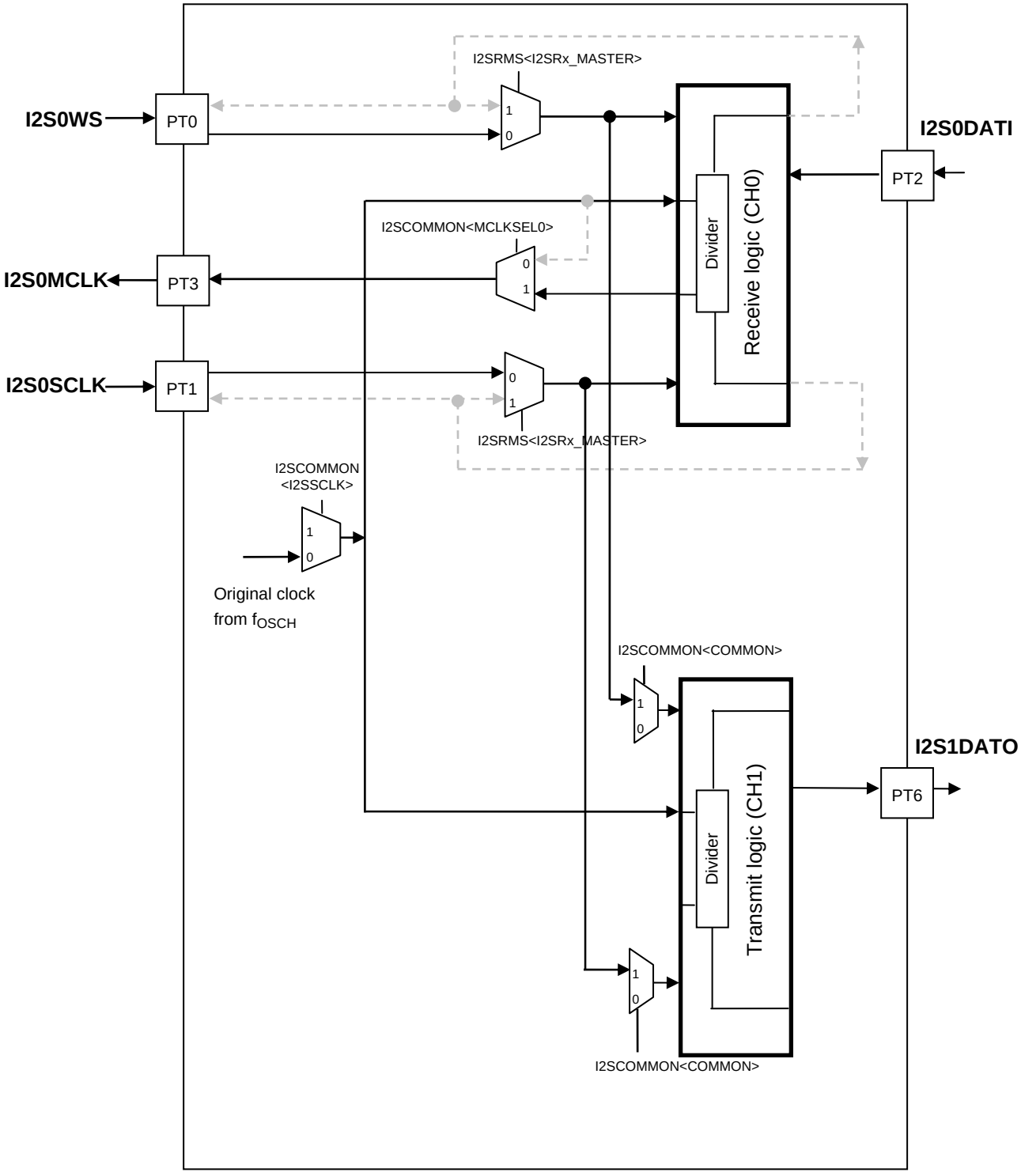
This mode that the receive logic (Ch0) is set as a master. The transmit operations (Ch1) is performed in synchronization with I2S0WS and I2S0SCLK that are output from the receive logic.



3.17.2.2 Mode Example 2

(Full-duplex Slave, <I2SSCLK> = 0, <MCLKSEL0> = 1, <COMMON> = 1, <I2SRx_MASTER> = 0)

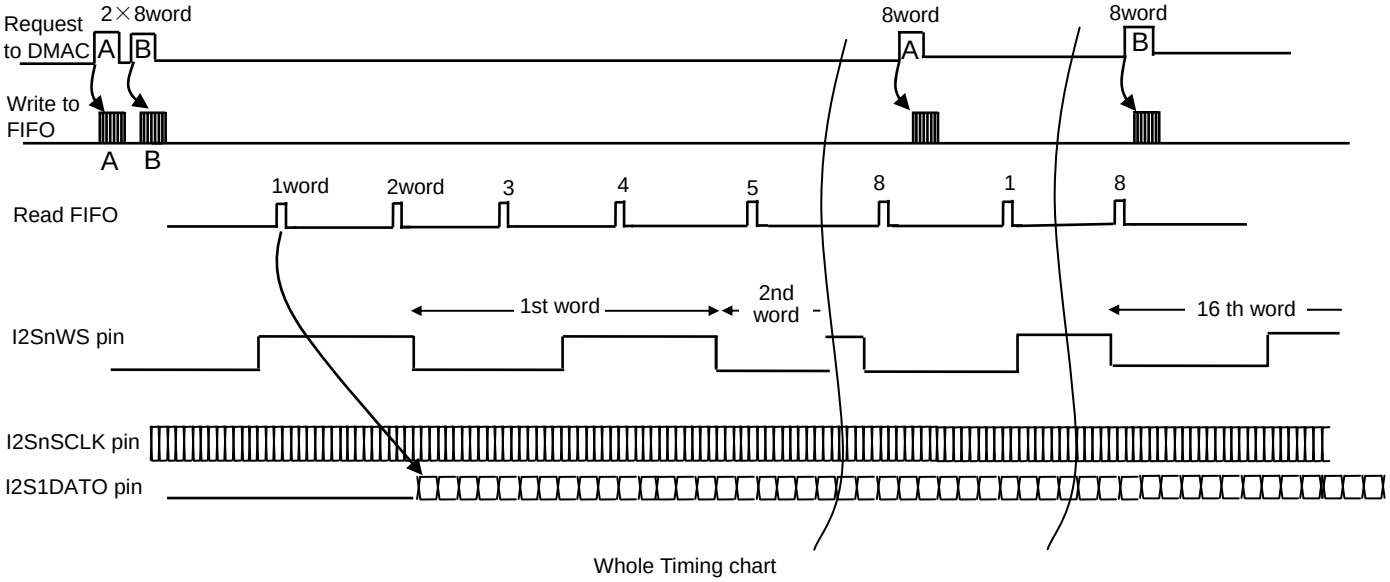
This mode that the receive logic (Ch0) is set as a slave. The transmit operations (Ch1) is performed in synchronization with I2S0WS and I2S0SCLK that are output from the other transmit logic.



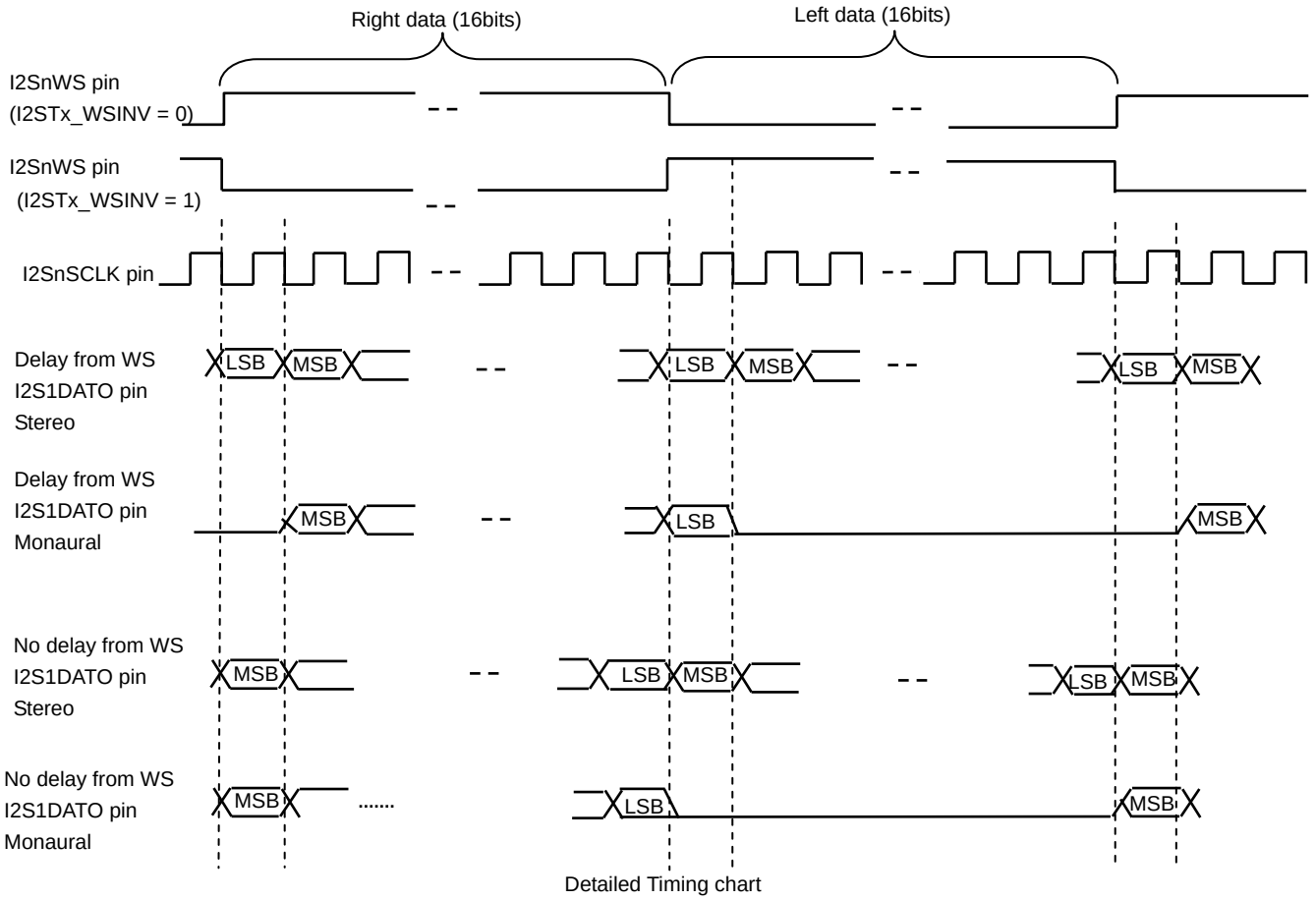
3.17.3 Operation Description

3.17.3.1 I^2S Output format

(I2STCON<I2STx_DELAYOFF> = 0, Delay from WS)



Whole Timing chart



Detailed Timing chart

3.17.4 Register Descriptions

The following lists the SFRs.

Base address = 0xF204_0000

Register Name	Address (base+)	Description
I2STCON	0x0000	Tx Control Register
I2STSLVON	0x0004	Tx I^2S Slave Control Register
I2STFCLR	0x0008	Tx FIFO Clear Register
–	0x000C	Reserved
–	0x0010	Reserved
–	0x0014	Reserved
I2STDMA1	0x0018	Tx DMA Ready Register
–	0x001C	Reserved
I2SRCON	0x0020	Rx Control Register
I2SRSLVON	0x0024	Rx I^2S Slave WS/SCK Control Register
I2SFRFCLR	0x0028	Rx FIFO Clear Register
I2SRMS	0x002C	Rx Master/Slave Select Register
I2SRMCON	0x0030	Rx Master I2S0WS/I2S0SCLK Period Register
I2SRMSTP	0x0034	Rx Master Stop Register
I2SRDMA1	0x0038	Rx DMA Ready Register
–	0x003C	Reserved
I2SCOMMON	0x0044	Common WS/SCK and Loop Setting Register
I2STST	0x0048	I^2S Tx Status Register
I2SRST	0x004C	I^2S Rx Status Register
I2SINT	0x0050	I^2S Interrupt Register
I2SINTMSK	0x0054	I^2S Interrupt Mask Register
I2STDAT	0x1000 to 0x1FFF	Transmit FIFO Window DMA Target
I2SRDAT	0x2000 to 0x2FFF	Receive FIFO Window DMA Target

1. I2STCON (Tx Control Register)

Address = (0xF204_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:14]	–	–	Undefined	Read as undefined. Write as zero.
[13:12]	I2STx_RLCH_CUT	R/W	0y00	Stereo/Monaural output setting 0y00: Stereo setting (both channel output) 0y01: Monaural setting (Right-side channel output) 0y10: Monaural setting (Left-side channel output) 0y11:Don't setting
[11:9]	–	–	Undefined	Read as undefined. Write as zero.
[8]	I2STx_BITCNV	R/W	0y0	MSB sign bit inversion 0y0: Not inverted 0y1: Inverted
[7:4]	–	–	Undefined	Read as undefined. Write as zero.
[3]	I2STx_UNDERFLOW	R/W	0y0	Data output at FIFO underflow 0y0: 0 is output. 0y1: The current data is held.
[2]	I2STx_MSBINV	R/W	0y0	LSB/MSB first 0y0: MSB first 0y1: LSB first
[1]	I2STx_WSINV	R/W	0y0	WS channel definition inversion 0y0: WS = 1 (RCH), WS = 0 (LCH) 0y1: WS channel definition inverted WS = 0 (RCH), WS = 1 (LCH)
[0]	I2STx_DELAYOFF	R/W	0y0	Relationship between Data output timing and WS 0y0: Delay of 1CLOCK from WS0y1: No delay from WS

[Description]

a. <I2STx_RLCH_CUT>

Stereo/monaural (Right-side channel output, Leftt-side channel output) output setting.

0y00: Stereo setting (both channel output)

0y01: Monaural setting (Right-side channel output)

0y10: Monaural setting (Left-side channel output)

0y11: Don't setting

b. <I2STx_BITCNV>

Specifies whether to invert the MSB (sign bit).

0y0: Not inverted

0y1: Inverted

c. <I2STx_UNDERFLOW>

If the valid data of the internal output FIFO becomes empty states, the data output is kept. This bit defines that output data. (SD output data when FIFO UnderFlow).

0y0: 0 is output.

0y1: The current data is held.

d. <I2STx_MSBINV>

Selection from LSB/MSB first.

0y0: MSB first

0y1: LSB first

e. <I2STx_WSINV>

Specifies whether to invert the channel definition of WS.

0y0: WS = 1 (RCH), WS = 0 (LCH)

WS signal is at High output, the WS is defined as Right Channel.

WS signal is at Low output, the WS is defined as Left Channel.

0y1: WS = 0 (RCH), WS = 1 (LCH)

WS signal is at Low output, the WS is defined as Right Channel.

WS signal is at High output, the WS is defined as Left Channel.

f. <I2STx_DELAYOFF>

Selects Relationship between Data output timing and WS.

0y0: Delay of 1CLOCK from WS

0y1: No delay from WS

2. I2SRCON (Rx Control Register)

Address = (0xF204_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:14]	–	–	Undefined	Read as undefined. Write as zero.
[13:12]	I2SRx_RCH_CUT	R/W	0y00	Stereo/Monaural output setting 0y00: Stereo setting (both channel output) 0y01: Monaural setting (Right-side channel output) 0y10: Monaural setting (Left-side channel output) 0y11:Don't setting
[11:9]	–	–	Undefined	Read as undefined. Write as zero.
[8]	I2SRx_BITCNV	R/W	0y0	MSB (sign bit) inversion 0y0: Not inverted 0y1: Inverted
[7:3]	–	–	Undefined	Read as undefined. Write as zero.
[2]	I2SRx_MSBINV	R/W	0y0	LSB/MSB first 0y0: MSB first 0y1: LSB first
[1]	I2SRx_WSINV	R/W	0y0	WS channel definition inversion 0y0: WS= 1 (RCH), WS= 0 (LCH) 0y1: WS channel definition inverted WS= 0 (RCH), WS= 1 (LCH)
[0]	I2SRx_DELAYOFF	R/W	0y0	Relationship between Data output timing and WS 0y0: Delay of 1CLOCK from WS 0y1: No delay from WS

[Description]

a. <I2SRx_RLCH_CUT>

Stereo/monaural (Right-side channel output, left-side channel output) output setting.

0y00: Stereo setting (both channel output)

0y01: Monaural setting (Right-side channel output)

0y10: Monaural setting (Left-side channel output)

0y11: Don't setting

b. <I2SRx_BITCNV>

Specifies whether to invert the MSB (sign bit).

0y0: Not inverted

0y1: Inverted

c. <I2SRx_UNDERFLOW>

Selects the data to be output when an underflow occurs in the FIFO.

0y0: 0 is output.

0y1: The current data is held.

d. <I2SRx_MSBINV>

Selection from LSB/MSB first.

0y0: MSB first

0y1: LSB first

e. <I2SRx_WSINV>

Specifies whether to invert the channel definition of WS.

0y0: WS = 1 (RCH), WS = 1 (LCH)

0y1: WS channel definition inverted

WS = 0 (RCH), WS = 1 (LCH)

f. <I2SRx_DELAYOFF>

Selects Relationship between Data output timing and WS.

0y0: Delay of 1CLOCK from WS

0y1: No delay from WS

3. I2STSLVON (Tx I^2S Slave Control Register)

Address = (0xF204_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2STx_SLAVE	R/W	0y0	Transmit output enable 0y0: OFF 0y1: ON (FIFO read enabled)

[Description]

a. <I2STx_SLAVE>

When this bit is set (0→ 1), the internal status (I2STST<I2STxSTATUS>) changes as follows:

SBY (standby) → PRE_ACT → ACT

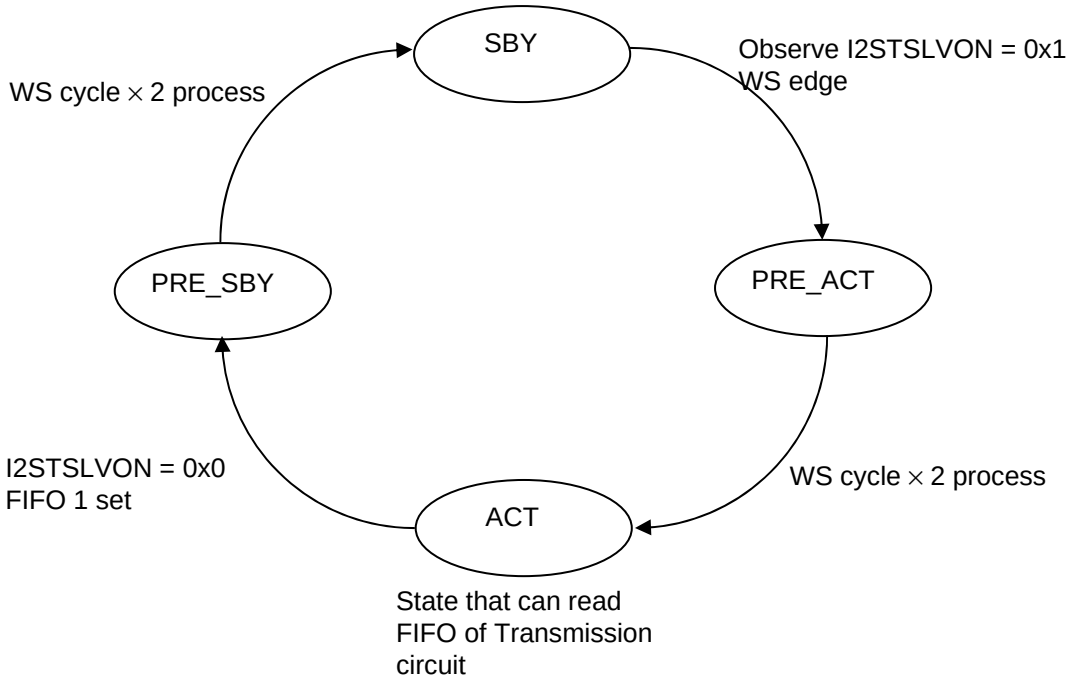
In the ACT state, the data stored in the FIFO is output.

When this bit is cleared (1 → 0), the internal status (I2STST<I2STxSTATUS>) changes as follows:

ACT → PRE_SBY → SBY

In the SBY state, no data is output from the FIFO even when it contains data.

Transmission circuit State machine



Note: The current status of internal operation can be read by I2STST<I2STxSTATUS>

4. I2SRSLVON (Rx I^2S Slave Control Register)

Address = (0xF204_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2SRx_SLAVE	R/W	0y0	Write the FIFO for receiver 0y0: OFF 0y1: ON (FIFO write enabled)

[Description]

a. <I2SRx_SLAVE>

When this bit is set (0 → 1), the internal status (I2SRST<I2SRxSTATUS>) changes as follows:

SBY (standby) → PRE_ACT → ACT

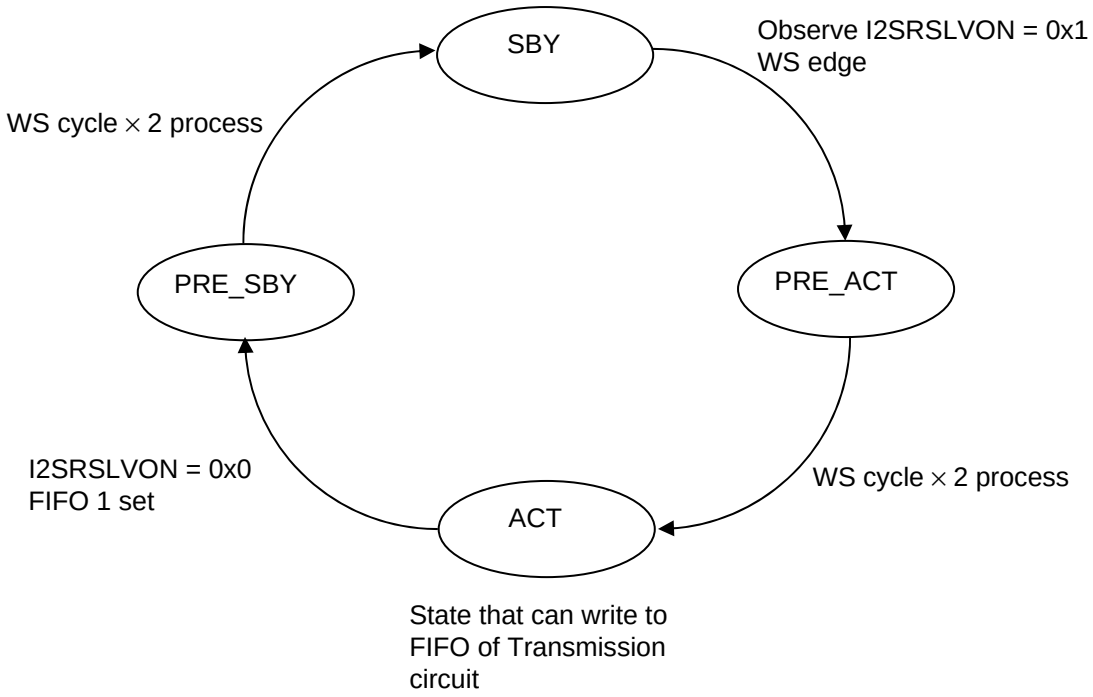
In the ACT state, data is captured into the FIFO.

When this bit is cleared (1 → 0), the internal status (I2SRST<I2SRxSTATUS>) changes as follows:

ACT → PRE_SBY → SBY

In the SBY state, no data is captured into the FIFO even when input data is present.

Receive circuit State machine



Note: The current status of internal operation can be read by I2STST<I2STxSTATUS>

5. I2STFCLR (Tx FIFO Clear Register)

Address = (0xF204_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2STx_FIFOCLR	R/W	0y0	FIFO Pointer clear 0y0: Invalid 0y1: FIFO Pointer clear

[Description]

a. <I2STx_FIFOCLR>
Do not clear the FIFO during DMA transfer as it may destroy the transmit data.
This bit is always read as 0.

6. I2SFRFCLR (Rx FIFO Clear Register)

Address = (0xF204_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2SRx_FIFOCLR	R/W	0y0	FIFO Pointer clear 0y0: Invalid 0y1: FIFO Pointer clear

[Description]

a. <I2SRx_ FIFOCLR>
Do not clear the FIFO during DMA transfer as it may destroy the receive data.
This bit is always read as 0.

7. I2SRMS (Rx Master/Slave Register)

Address = (0xF204_0000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2SRx_MASTER	R/W	0y0	Master/slave select 0y0: Slave 0y1: Master (Internally generated I2S0WS and I2S0SCLK are output to an external device.)

[Description]

a. <I2SRx_MASTER>
Selects between receive master and receive slave.
I2SCOMMON<COMMON> is set to 1, this bit selects between full-duplex master and full-duplex slave.

8. I2SRMCON (Rx Master I2S0WS/I2S0SCLK Period Register)

Address = (0xF204_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3:2]	I2SRx_WS_DIV[1:0]	R/W	0y0	Ratio source clock to I2S0MCLK 0y00: 1/1 0y01: 1/2 0y10: 1/4 0y11: Do not set
[1:0]	I2SRx_SCLK_DIV[1:0]	R/W	0y0	Ratio I2S0MCLK to I2S0SCLK 0y00: 1/8 0y01: 1/12 0y10: 1/16 0y11: Do not set

[Description]

a. <I2SRx_WS_DIV[1:0]>, <I2SRx_SCLK_DIV[1:0]>
I2SCOMMON<COMMON> is set to 1, full-duplex mode is enabled.

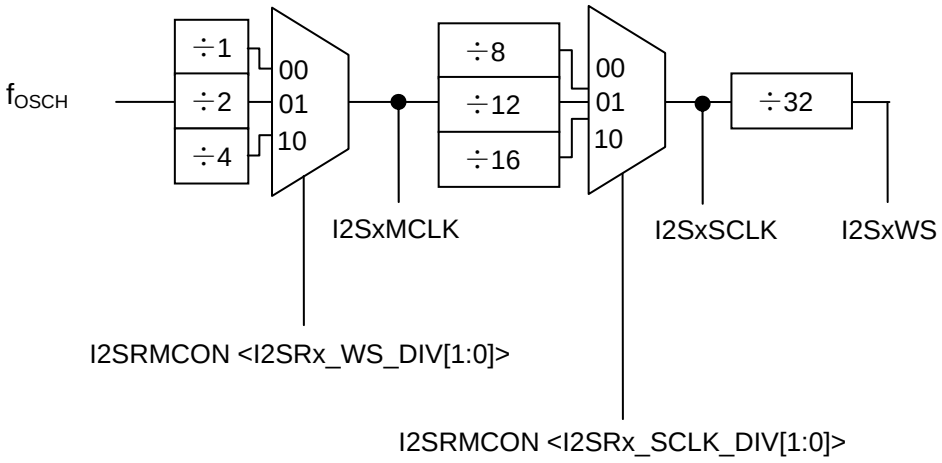


Table 3.17.2 Clock setting table

I2SRMCON I2SRx_WS_DIV[1:0]	Ratio of I2S0MCLK to source clock	Ratio of I2S0SCLK to source clock	I2SRMCON I2SRx_SCLK_DIV[1:0]	Ratio of I2S0WS to source clock
0y00: (1/1)	1/1	1/8	0y00: (1/256)	1/256
0y01: (1/2)	1/2	1/16	0y00: (1/256)	1/512
0y10: (1/4)	1/4	1/32	0y00: (1/256)	1/1024
0y00: (1/1)	1/1	1/12	0y01: (1/384)	1/384
0y01: (1/2)	1/2	1/24	0y01: (1/384)	1/768
0y10: (1/4)	1/4	1/48	0y01: (1/384)	1/1536
0y00: (1/1)	1/1	1/16	0y10: (1/512)	1/512
0y01: (1/2)	1/2	1/32	0y10: (1/512)	1/1024
0y10: (1/4)	1/4	1/64	0y10: (1/512)	1/2048

Table 3.17.3 Audio sampling setting examples based on 48 kHz

f_{OSCH}	I2SRMCON I2SRx_WS_DIV[1:0]	I2S0MCLK frequency (Ratio to source clock)	I2S0SCLK frequency (Ratio to source clock)	I2SRMCON I2SRx_SCLKS_DIV[1:0]	I2S0WS frequency (Ratio to source clock)
12.288 MHz	0y00: (1/1)	12.288 MHz (1/1)	1536 kHz (1/8)	0y00: (1/256)	48 kHz (1/256)
	0y01: (1/2)	6.144 MHz (1/2)	768 kHz (1/16)	0y00: (1/256)	24 kHz (1/512)
	0y10: (1/4)	3.072 MHz (1/4)	384 kHz (1/32)	0y00: (1/256)	12 kHz (1/1024)
18.432 MHz	0y00: (1/1)	18.432 MHz (1/1)	1536 kHz (1/12)	0y01: (1/384)	48 kHz (1/384)
	0y01: (1/2)	9.216 MHz (1/2)	768 kHz (1/24)	0y01: (1/384)	24 kHz (1/768)
	0y10: (1/4)	4.608 MHz (1/4)	384 kHz (1/48)	0y01: (1/384)	12 kHz (1/1536)
24.576 MHz	0y00: (1/1)	24.576 MHz (1/1)	1536 kHz (1/16)	0y10: (1/512)	48 kHz (1/512)
	0y01: (1/2)	12.288 MHz (1/2)	768 kHz (1/32)	0y10: (1/512)	24 kHz (1/1024)
	0y10: (1/4)	6.144 MHz (1/4)	384 kHz (1/64)	0y10: (1/512)	12 kHz (1/2048)

9. I2SRMSTP (Rx Master Stop Register)

Address = (0xF204_0000) + (0x0034)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2SRx_MSTOP	R/W	0y0	I2SRx master stop: 0y0: Do not stop I2S0WS/I2S0SCLK 0y1: Stop I2S0WS/I2S0SCLK (I2S0WS/I2S0SCLK = 0)

[Description]

a. <I2SRx_MSTP>

This bit is used to stop (= Fixed Low level) I2S0WS and I2S0SCLK from the master. It is not normally used.

Before setting this register, make sure that I2SRx is in the SBY state (I2SRST<I2SRx_STATUS[1:0]>＝0y00). Operation is not guaranteed in other cases.

The default setting is not to stop I2S0WS and I2S0SCLK. Therefore, after master-related settings are made, I2S0WS and I2S0SCLK are immediately output.

10. I2STDMA1 (Tx DMA Ready Register)

Address = (0xF204_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2STx_DMAREADY1	R/W	0y0	I2STx DMA ready: 0y0: Disable 0y1: Enable

[Description]

a. <I2STx_ DMAREADY1>

This register indicates the DMA ready state to the hardware logic.

This register should set to "1" by software after both the DMA and I2S operational configuration are completed, then the hardware logic can be recognized the DMA ready. And the hardware logic monitors the FIFO and starts DMA transfer.

Note: To disable this register, make sure whether the DMA transfer is completed first and the I2STSLVON register set to 0y0, the I2STST<I2STx_STATUS> is SBY and then this register can be set to 0y0.

11. I2SRDMA1 (Rx DMA Ready Register)

Address = (0xF204_0000) + (0x0038)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	I2SRx_DMAREADY1	R/W	0y0	I2SRx DMA ready: 0y0: Disable 0y1: Enable

[Description]

a. <I2SRx_ DMAREADY1>

This register indicates the DMA ready state to the hardware logic.

This register should set to "1" by software after both the DMA and I2S operational configuration are completed, then the hardware logic can be recognized the DMA ready. And the hardware logic monitors the FIFO and starts DMA transfer.

Note: To disable this register, make sure whether the DMA transfer is completed first and the I2STSLVON register set to 0y0, the I2STST<I2STx_STATUS> is SBY and then this register can be set to 0y0.

12. I2SCOMMON (Common WS/SCK and Loop Setting Register)

Address = (0xF204_0000) + (0x0044)

Bit	Bit Symbol	Type	Reset Value	Description
[31:6]	–	–	Undefined	Read as undefined. Write as zero.
[5]	Reserved	WO	0y0	Read as undefined. Write as zero.
[4]	MCLKSEL0	WO	0y0	Master clock to be output from the receive logic: 0y0: Audio source clock 0y1: Divided-down audio source clock
[3]	MCLKSEL1	WO	0y0	Master clock to be output from the transmit logic: 0y0: Audio source clock 0y1: Divided-down audio source clock
[2]	I2SSCLK	WO	0y0	Audio source clock: 0y0: f_{OSCH} 0y1: Reserved
[1]	LOOP	R/W	0y0	Loop setting 0y0: Loop disabled 0y1: Loop enabled
[0]	COMMON	R/W	0y0	Full-duplex mode setting 0y0: Reserved 0y1: Full-duplex and should be set to 0y1

[Description]

a. <MCLKSEL0>
Selects the master clock to be output from the receive logic.
0y0: Audio source clock
0y1: Divided-down audio source clock

b. <MCLKSEL1>
Selects the master clock to be output from the transmit logic.
0y0: Audio source clock
0y1: Divided-down audio source clock

c. <I2SSCLK>
Selects the audio source clock to be used.
0y0: f_{OSCH}
0y1: Reserved

d. <LOOP>

Specifies loop setting what the transmit pin outputs the data via transmit operation by inputting the serial data from receive pin.

0y0: Loop disabled

0y1: Loop enabled

e. <COMMON>

When this bit is set to 1, the SCK and WS input signals of I2SRx are also used by I2STx. In this case, the settings made for the transmmit master have no effect.

0y0: Reserved

0y1: Full-duplex mode and should be set to 0y1

13. I2STST (I^2S Tx Status Register)

Address = (0xF204_0000) + (0x0048)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined.
[3:2]	I2STx_STATUS[1:0]	RO	0y00	FIFO status: 0y00: SBY 0y01: PreACT 0y10: PreSBY 0y11: ACT
[1]	I2STx_FIFOFULL	RO	0y0	FIFO full status: 0y0: Not full 0y1: Full
[0]	I2STx_FIFOEMPTY	RO	0y1	FIFO empty status: 0y0: Not empty 0y1: Empty

[Description]

a. <I2STx_STATUS[1:0]>
Indicates the FIFO status.
0y00: SBY
0y01: PreACT
0y10: PreSBY
0y11: ACT

b. <I2STx_FIFOFULL>
Indicates the FIFO full status.
0y0: Not full
0y1: Full

c. <I2STx_FIFOEMPTY>
Indicates the FIFO empty status.
0y0: Not empty
0y1: Empty

14. I2SRST (I^2S Rx Status Register)

Address = (0xF204_0000) + (0x004C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined.
[3:2]	I2SRx_STATUS[1:0]	RO	0y00	FIFO write status: 0y00: SBY 0y01: PreACT 0y10: PreSBY 0y11: ACT
[1]	I2SRx_FIFOFULL	RO	0y0	FIFO full status: 0y0: Not full 0y1: Full
[0]	I2SRx_FIFOEMPTY	RO	0y1	FIFO empty status: 0y0: Not empty 0y1: Empty

[Description]

a. <I2SRx_STATUS [1:0]>
Indicates the FIFO write status.
0y00: SBY
0y01: PreACT
0y10: PreSBY
0y11: ACT

b. <I2SRx_FIFOFULL>
Indicates the FIFO full status.
0y0: Not full
0y1: Full

c. <I2SRx_FIFOEMPTY>
Indicates the FIFO empty status.
0y0: Not empty
0y1: Empty

15. I2SINT (I^2S Interrupt Register)

Address = (0xF204_0000) + (0x0050)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3]	I2SRx_OVERFLOW_INT	R/W	0y0	Rx FIFO overflow interrupt: Read: 0y0: No interrupt 0y1: Interrupt generated Write: 0y0: Invalid 0y1: Clear
[2]	I2SRx_UNDERFLOW_INT	R/W	0y0	Rx FIFO underflow interrupt: Read: 0y0: No interrupt 0y1: Interrupt generated Write: 0y0: Invalid 0y1: Clear
[1]	I2STx_OVERFLOW_INT	R/W	0y0	Tx FIFO overflow interrupt: Read: 0y0: No interrupt 0y1: Interrupt generated Write: 0y0: Invalid 0y1: Clear
[0]	I2STx_UNDERFLOW_INT	R/W	0y0	Tx FIFO underflow interrupt: Read: 0y0: No interrupt 0y1: Interrupt generated Write: 0y0: Invalid 0y1: Clear

[Description]

a. <I2SRx_OVERFLOW_INT>, <I2SRx_UNDERFLOW_INT>, <I2STx_OVERFLOW_INT>, <I2STx_UNDERFLOW_INT>

This register indicates the interrupt status of each interrupt source. To monitor the FIFO error status by using each interrupt source, the corresponding bit of the interrupt mask register (I2SINTMSK) must be cleared.

When an interrupt is generated from one of these sources, the interrupt controller generates an I2SINT interrupt. The interrupt source can be identified by monitoring each interrupt source bit of the I2SINT register.

Each bit of this register is cleared to 0 by writing 1.

Note: This register is corresponded when writing "0" to the corresponding bit of the interrupt mask register (I2SINTMSK).

16. I2SINTMSK (I^2S Interrupt Mask Register)

Address = (0xF204_0000) + (0x0054)

Bit	Bit Symbol	Type	Reset Value	Description
[31:4]	–	–	Undefined	Read as undefined. Write as zero.
[3]	I2SRx_OVERFLOW_INTMS	R/W	0y1	Rx FIFO overflow setting: 0y0: Enable 0y1: Disable
[2]	I2SRx_UNDERFLOW_INTM	R/W	0y1	Rx FIFO underflow setting: 0y0: Enable 0y1: Disable
[1]	I2STx_OVERFLOW_INTMS	R/W	0y1	Tx FIFO overflow setting: 0y0: Enable 0y1: Disable
[0]	I2STx_UNDERFLOW_INTM	R/W	0y1	Tx FIFO underflow setting: 0y0: Enable 0y1: Disable

[Description]

a. <I2SRx_OVERFLOW_INTMS>
The Rx FIFO overflow setting
0y0: Enable
0y1: Disable

b. <I2SRx_FIFOFULL>
The Rx FIFO underflow setting
0y0: Enable
0y1: Disable

c. <I2SRx_FIFOEMPTY>
The Tx FIFO overflow setting
0y0: Enable
0y1: Disable

d. <I2SRx_FIFOFULL>
The Tx FIFO underflow setting
0y0: Enable
0y1: Disable

17. I2STDAT (Transmit FIFO Window DMA Target Register)

Address = (0xF204_1000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	Left[15:0]	WO	0x0000	I2STx left audio data [15:0]
[15:0]	Right[15:0]	WO	0x0000	I2STx right audio data [15:0]

[Description]

a. <Left[15:0]>, <Right[15:0]>

The set data in this register is written into transmit FIFO, refer to Note 2.

Stereo audio data is set simultaneously with upper data as left channel data and lower data as right channel data. Data can be written to any address in a range of 0xF2041000 to 0xF2041FFF, and is sequentially stored in the FIFO as it is written. This register does not support read operations.

Note1: this register is write-only.

Note2: this register can be accessed from CPU/DMAC as Master.

18. I2SRDAT (Receive FIFO Window Target Register)

Address = (0xF204_2000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:16]	Left[15:0]	RO	0x0000	I2SRx left audio data [15:0]
[15:0]	Right[15:0]	RO	0x0000	I2SRx right audio data [15:0]

[Description]

a. <Left[15:0]>, <Right[15:0]>

To read register data, the data is sequentially read out from receive FIFO, refer to Note 2.

Stereo audio data is input simultaneously with upper data as left channel data and lower data as right channel data. Data can be read from any address in a range of 0xF2042000 to 0xF2042FFF, and is sequentially read out from the FIFO.

Note1: this register is read-only.

Note2: this register can be accessed from CPU/DMAC as Master.

3.17.5 Setting example

- Setting example of Transmission master and Receive slave

```
                I2SCOMMON           ←  0x00000001    ; write 0x00000001 to Register
                I2STCON             ←  0x00000000
                I2SRCON             ←  0x00000000
                I2SRMS              ←  0x00000000    ; Rx is slave

                GPIOMFR1            ←  0x000000FF
                GPIOLFR1            ←  0x000000FF
                0xf8004000          ←  0x0000FFFF    ; Set transfer data
                . . .               ←  . . .
                0xf800403c          ←  0xFFFF0000    ; End of set data
                                                     ; Use DMA scatter gather link
                0xf8004040          ←  0xf8004020    ;  source address
                0xf8004044          ←  I2STDAT       ; destination address
                0xf8004048          ←  0xf8004050    ; next address
                0xf800404c          ←  0x04492008    ; Set DMAC control register
                . . .               ←  . . .
                DMACConfiguration   ←  0x00000001    ; Set Rx  DMAC
                DMACC0SrcAddr       ←  I2SRDAT
                DMACC0DestAddr      ←  0xf8008000
                DMACC0Control       ←  0x08492008
                DMACC0Configuration ←  0x00001017
                I2SRSLVON           ←  0x00000001    ; I2S internal clock on
i2sr_act                                             ; label  i2sr_act
                I2SRST              →  r0            ; read I2SRST register data to r0
                LDR     r1, = 0xc
                AND     r0,r0,r1
                LDR     r2, = 0xc
                CMP     r0,r2                        ; check I2S Active
                BNE     i2sr_act                     ; r0 ≠r2 ,   jump to i2sr_act
                I2SRDMA1            ←  0x00000001    ;I2S DMA Ready

                DMACConfiguration   ←  0x00000001    ; Set Tx DMAC
                DMACC1SrcAddr       ←  0xf8004000
                DMACC1DestAddr      ←  I2STDAT
                DMACC1Control       ←  0x04492008
                DMACC1Configuration ←  0x00000a81
                I2STSLVON           ←  0x00000001    ; I2S internal clock on
i2st_act                                             ; label  i2st_act
                I2SRST              →  r0            ; read I2SRST register data to r0
                LDR     r1, = 0xc
                AND     r0,r0,r1
                LDR     r2, = 0xc
                CMP     r0,r2                        ; check I2S Active
                BNE     i2st_act                     ; r0 ≠r2 ,   jump to i2st_act
                I2STDMA1            ←  0x00000001    ;I2S DMA Ready
finish_DMA                                           ; label
                DMACC0Control       →  r0            ; read DMACC0Control  data to r0
                CMP   r0,#0x0                        ; check the End of Rx DMAC
                BNE   finish_DMA                     ; r0 ≠0x0 , jump to finish_DMA
                I2STDMA1            ←  0x00000000    ; DMA Clear
                I2SRDMA1            ←  0x00000000
                I2STSLVON           ←  0x00000000    ; internal clock off
                I2SRSLVON           ←  0x00000000
i2s_stop_t                                           ; label
                I2STST              →  r0            ; read I2STST register data to r0
                LDR     r1, = 0xc
                AND     r0,r0,r1
                LDR     r1, = 0x0
                CMP     r0,r1
                BNE     i2s_stop_t                   ; check I2S Tx standby
i2s_stop_r                                           ; label
                I2SRST              →  r0            ; read I2SRST register data to r0
                LDR   r1, = 0xc
                AND   r0,r0,r1
                LDR   r1, = 0x0
                CMP   r0,r1                          ; check I2S Rx standby
                BNE   i2s_stop_r                     ; r0 ≠r1 ,   jump to i2st_stop_r
                I2STFCLR            ←  0x00000001    ; clear Tx FIFO
                I2SFRFCLR           ←  0x00000001    ; clear Rx FIFO
```

3.18 LCD Controller (LCDC)

3.18.1 Overview

This LSI incorporates a color-capable LCD controller (LCDC).

The LCDC has the following characteristics:

Table 3.18.1 Characteristics of LCDC

Type of LCD panel		TFT	STN (Dual/Single)
Displayable colors at same (Palette color change available)		~ 256 colors	2,4,16,256 colors
Displayable colors (Palette color change unavailable)		~ 65536 colors	3,375 colors
Bit per Pixel (Data quantity per pixel)		1/2/4/8/16 bit	1/2/4/8/16 bit
Number of available horizontal pixels		16× (PPL + 1)dot : PPL values take integers of 0 to 63 only. (Note)	
Number of available vertical lines		4 to 1,024 (integer)*	
Transfer-destination data bus width (LCD driver)		Max. 16 bit	4/8 bit
FIFO buffer for display data receive		32 bit ×16 word × 2	
Timing adjustment function		Can program the front/back porch timings of vertical/horizontal sync signals.	
Display palette		256 entries, 16-bit palette RAM	
Data type		Little endian support	
Connection terminal	Terminals LD15 to LD0	Data buses for LCD driver	
	LCLAC terminal	Enable data "Enable" signal	AC bias signal (frame signal)
	LCLLP terminal	Horizontal sync signal	Horizontal sync pulse
	LCLFP terminal	Vertical sync signal	Vertical sync pulse
	LCLCP terminal	Clock for LCDD data latch (Panel clock)	
	LCLLE terminal	Line end signal	Not used basically
	LCLPOWER terminal	LCD panel power control signal (Note: Not supported by this LSI)	

Note: In the display size, limitations occur depending on display colors and operation clocks. This is reference as follows.

LCD type	Displayable Maximum dot number
TFT 16bit Color	Approximately 350000dot (around 640×480)
TFT 8bit Color	Approximately 500000dot (around 800×600)
TFT 4bit Color	No particular limitations
TFT 2bit Color	No particular limitations
STN 15bit Color	Approximately 350000dot (around 640×480)
STN 8bit Color	Approximately 700000dot (around 960×640)
STN 4bit Color	No particular limitations
STN 2bit Color	No particular limitations
STN 1bit Color(Monochrome)	No particular limitations

3.18.2 Function

Figure 3.18.1 shows the schematic block diagram of the LCDC.

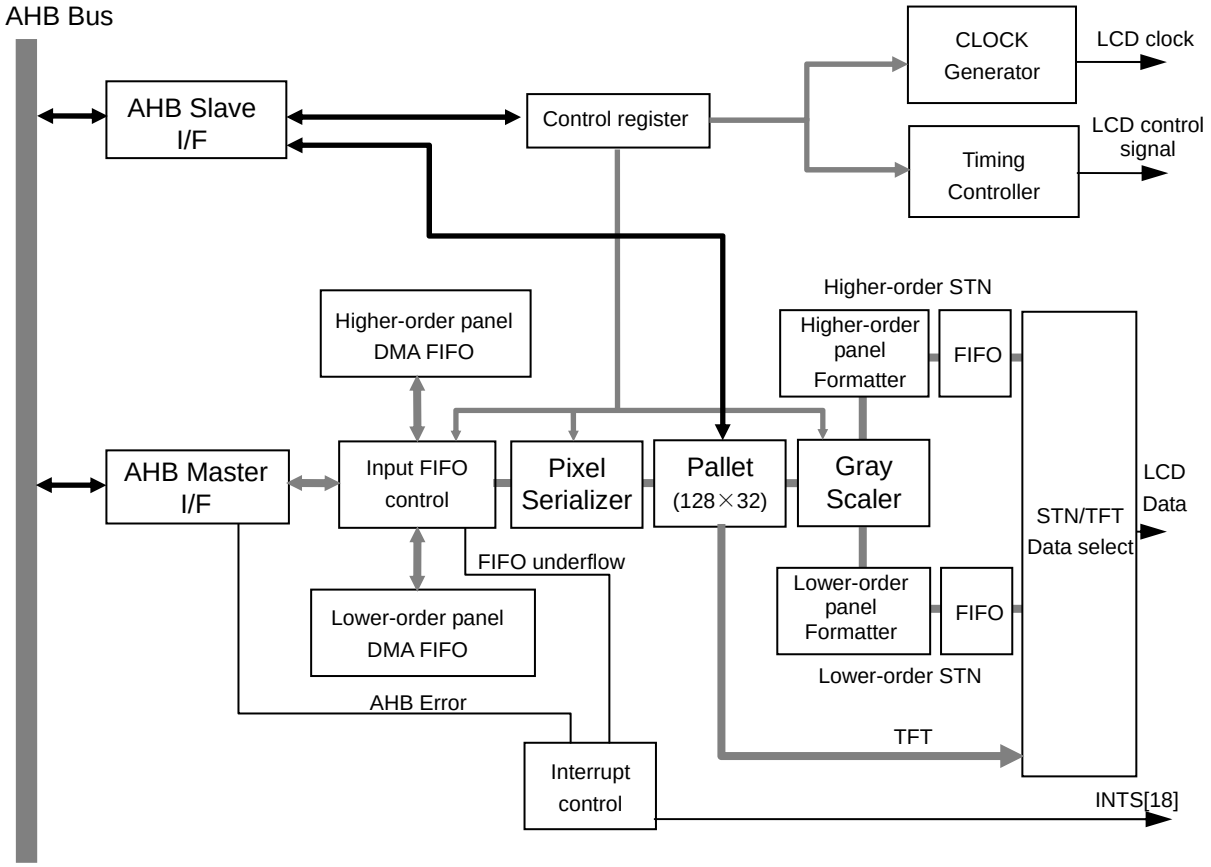


Figure 3.18.1 LCDC Block Diagram

The description of each block is shown in the next and following pages:

3.18.2.1 DMA FIFO and Related Control Logics

The FIFO's input port is connected to the interface; and the output port is connected to the pixel serializer.

In order to match the single/dual panel LCD types, display data read from the display RAM is buffered into the two DMA FIFOs that can control the data individually.

32 words of FIFO can be used. By the WATERMARK register setting, the FIFO requests data at the point when free space of 4 words or more, or 8 words or more occurs.

If LCD data is output with the FIFO empty, an underflow condition results, which asserts an interrupt signal.

3.18.2.2 Pixel Serializer

This block reads LCD data of 32-bit width from the DMA FIFO's output port and converts it into 24-, 16-, 8-, 4-, 2-, or 1-bit data according to the operation mode.

In the dual panel mode, data is divided into the higher DMA FIFO (16 words) and the lower DMA FIFO (16 words) and read alternately.

Data converted into a suitable size is used as color/gray level values in the palette RAM or output directly without bypassing the palette.

Table 3.18.2 LBLP: DMA FIFO output bits 31 to 16

bpp	DMA FIFO output bit															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	P15		P14		P13		P12		P11		P10		P9		P8	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	P7				P6				P5				P4			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	P3								P2							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	P1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24	P0															
	–	–	–	–	–	–	–	–	23	22	21	20	19	18	17	16

Table3.18.3 LBLP: DMA FIFO output bits 15 to 0

bpp	DMA FIFO output bit															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	P7		P6		P5		P4		P3		P2		P1		P0	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	P3				P2				P1				P0			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	P1								P0							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	P0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24	P0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

3.18.2.3 RAM Palette

A palette of 16 bits × 256 entries is incorporated.

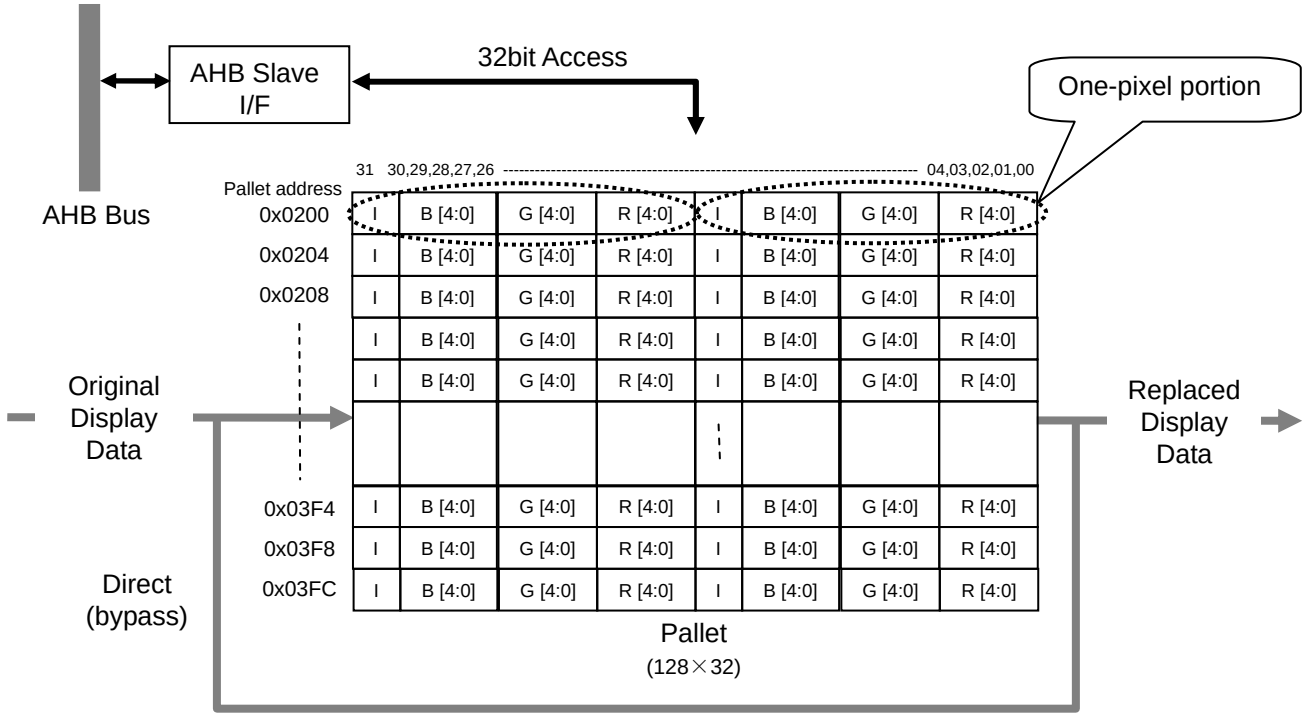


Figure 3.18.2 Palette RAM

Pixel data is replaced into data of in-palette 16 bits and then output.

(Original Display Data is processed as palette addresses, and then this data that into address as Replaced Display data is outputted. Moreover, if it is 16bpp, Original Display Data is outputted as Replaced Display data)

One word (32 bits) of palette RAM data equals to two pixels of data. Therefore, the lowest-order bit of pixel data is used to select either the higher 16 bits or lower 16 bits of the palette RAM.

Example) If 0x00 pixel data is input in 8 bpp, the data is replaced into the lower 16-bit data for an address of 0xF420_0200.
If 0xFF pixel data is input in 8 bpp, the data is replaced into the higher 16-bit data for an address of 0xF420_03FC.

A palette structured with R:5 bits, G:5 bits, and B:5 bits is structured with a dual port RAM of 128×32 bits. Therefore, pixel data of two-pixel can be written in 1 word.

LCD Palette

Address = (0xF420_0000) + ((0x0200) to (0x03FC))

Bit	Bit Symbol	Type	Reset Value	Description
[31]	I	R/W	0y0	Brightness/unused
[30:26]	B[4:0]	R/W	0y00000	Blue palette data setting
[25:21]	G[4:0]	R/W	0y00000	Green palette data setting
[20:16]	R[4:0]	–	0y00000	Red palette data
[15]	I	R/W	0y0	Brightness/unused
[14:10]	B[4:0]	R/W	0y00000	Blue palette data setting
[9:5]	G[4:0]	R/W	0y00000	Green palette data setting
[4:0]	R[4:0]	R/W	0y00000	Red palette data

In the monochrome STN mode (include Gray display), only red palette R [4:1] is used (bit0 not used).

In the STN color mode, red, green, and blue bits [4:1] are used (bit0 not used). To support the BGR data system, this red and blue pixel data can be swapped using the control register bit.

In the 16 / 24 bpp TFT mode, palettes are bypassed so that the pixel serializer's output can be directly used as TFT panel data.

A RAM palette supports 256 entries x 16 bits at maximum. Therefore, TFT and color STN palettes can support up to 8-bpp data.

Note: The LCD data process accelerator (LCDDA) contained in this microcontroller only supports 64K (16 bpp) or 16M (24 bpp) colors. Therefore, the LCDDA does not allow the use of the palette or the swap function.

3.18.2.4 Gray Scaler

The gray algorithm supports monochrome display 15 gray scale levels.

For STN color display, three color components (red, green, and blue) are processed for gray scale level concurrently, allowing 3,375 ($15 \times 15 \times 15$) colors to be usable.

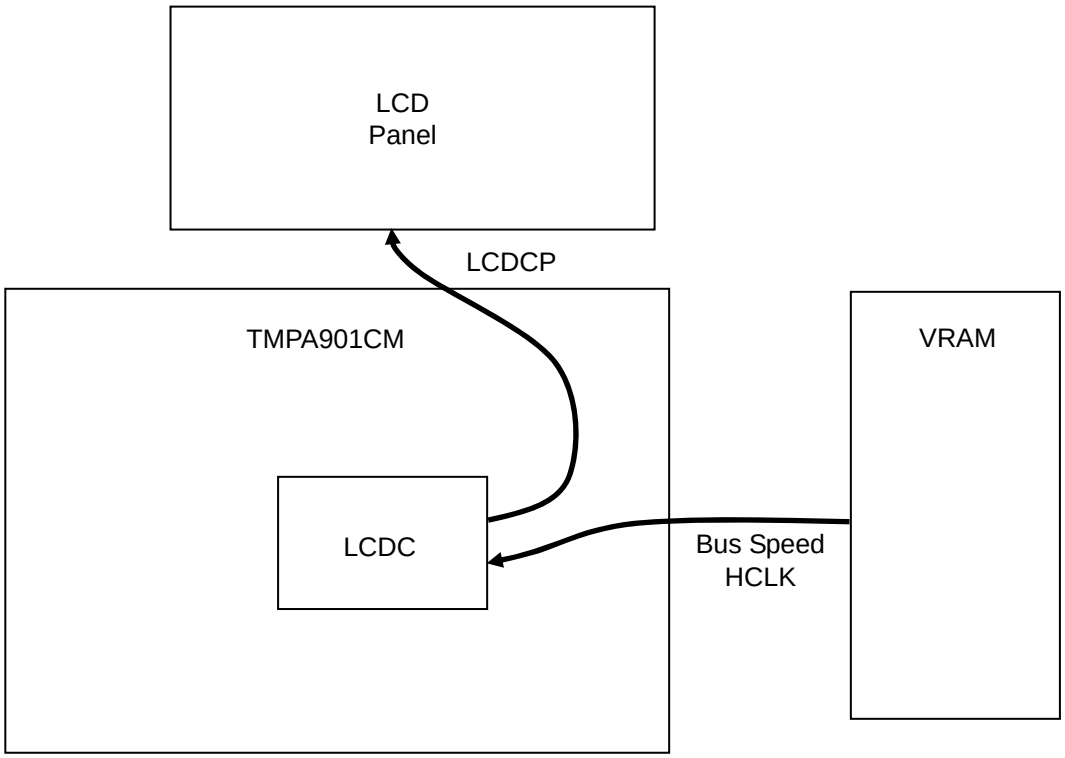
3.18.2.5 Higher-/Lower-Order Panel Formatter

Divides higher and lower pixel data for using Dual Panel.

RGB pixel data is shifted in to each of the unique registers per bit and concurrently to be configured in proper bit positions.

3.18.2.6 Panel Clock Generator

Can set the frequency division rate of data transfer clock (LCLCP signal) used in the internal clock (HCLK) and LCDC. The LCLCP signal can be programmed in the range of HCLK/2 to HCLK/1025 according to the data rate of the LCD panel.



3.18.2.7 Timing Controller

The main function of the timing controller block is to adjust horizontal/vertical timings.

3.18.2.8 Creating Interrupts

The LCDC generates four types of interrupts that are maskable individually and one type of joint interrupt.

3.18.3 Description of Registers

The following lists the SFRs:

Table3.18.4 List of registers

base address = 0xF420_0000

Register Name	Address (base+)	Description
LCDTiming0	0x000	Horizontal Axis Panel Control Register
LCDTiming1	0x004	Vertical Axis Panel Control Register
LCDTiming2	0x008	Clock and Signal Polarity Control Register
LCDTiming3	0x00C	Line End Control Register
LCDUPBASE	0x010	Upper Panel Frame Base Address Register
LCDLPBASE	0x014	Lower Panel Frame Base Address Register
LCDIMSC	0x018	Interrupt Mask Set/Clear Register
LCDControl	0x01C	LCDC Control Register
LCDRIS	0x020	Raw Interrupt Status Register
LCDMIS	0x024	Masked Interrupt Status Register
LCDICR	0x028	Interrupt Clear Register
LCDUPCURR	0x02C	Upper Panel Current Address Value Registers
LCDLPCURR	0x030	Lower Panel Current Address Value Registers
LCDPalette	0x200 to 0x3FC	Color Palette Register

Base address = 0xF00B_0000

Register Name	Address (base+)	Description
STN64CR	0x0000	LCDC Option Control Register for STN 64

1. LCDTiming0 (Horizontal Axis Panel Control Register)

LCDTiming0 is the register to control the following:

- Horizontal sync pulse width (HSW)
- Horizontal front porch (HFP) period
- Horizontal back porch (HBP) period
- Number of pixels per line (PPL)

Address = (0xF420_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	HBP	R/W	0x00	Value set for horizontal back porch width (Setting value + 1) 0x00 to 0xFF
[23:16]	HFP	R/W	0x00	Value set for horizontal front porch width (Setting value + 1) 0x00 to 0xFF
[15:8]	HSW	R/W	0x00	Horizontal sync pulse width (Setting value + 1) 0x00 to 0xFF
[7:2]	PPL	R/W	0y000000	Value set for pixels per line ((Setting value + 1) × 16) 0y000000 to 0y111111
[1:0]	Reserved	–	Undefined	Read as undefined. Write as zero.

[Description]

a. <HBP>

Horizontal back porch refers to the number of LCLCP cycles from the LCLLP rising (or falling) edge to the active data start.

The actual period counted is the value set in this field incremented by one. Therefore, a delay of 1 to 256 LCLCP cycles can be inserted.

b. <HFP>

Horizontal front porch refers to the number of LCLCP cycles from the active data end to the LCLLP falling (or rising) edge.

The actual period counted is the value set in this field incremented by one. Therefore, a delay of 1 to 256 LCLCP cycles can be inserted.

c. <HSW>

Horizontal sync pulse width refers to the active pulse width of LCLLP.

The actual period counted is the value set in this field incremented by one. Therefore, a delay of 1 to 256 LCLCP cycles can be inserted.

d. <PPL>

The PPL field is used to specify the number of pixels per line.

The actual pixel count is calculated by the formula "(PPL value + 1) × 16". Therefore, 16 to 1024 pixels can be specified.

- Limitations on horizontal timing

There is a restriction on the operation mode used.

minimum values are HSW = 2, HBP = 2.

STN single panel mode:

- HSW = 3
- HBP = 5
- HFP = 5
- Panel clock divisor (PCD) = 1(HCLK/3)

STN Dual panel mode:

- HSW = 3
- HBP = 5
- HFP = 5
- PCD = 5(HCLK/7)

Depending on usage conditions, setting enough time before the start point of line (example: HSW = 6, HBP = 10) prevents data from being corrupted even when PCD = 4 (minimum value).

The figure below shows an example of operation mode settings (LCDTiming2<IHS> = 1, LCDTiming2<IPC> = 0, LCDTiming2<IOE> = 0):

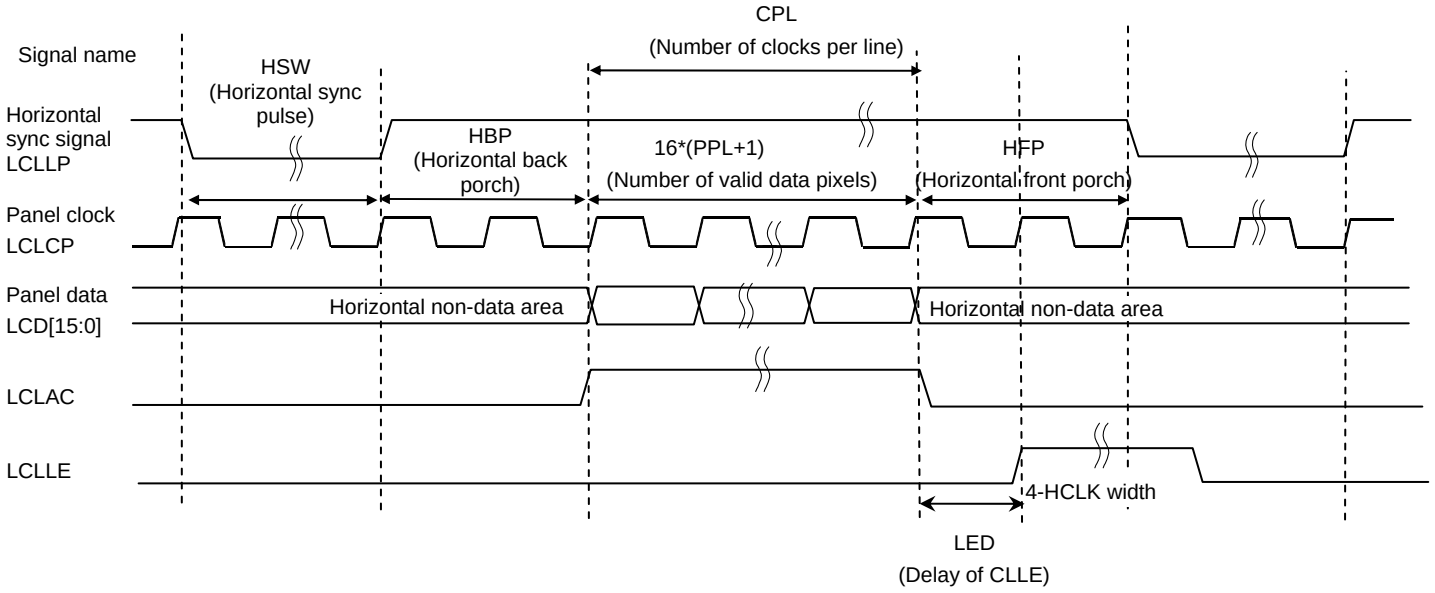


Figure 3.18.3 Basic operation of horizontal control

Note: For CPL, divide PPL by 1 (TFT), 4 or 8 (monochrome STN), or (2 + 2/3) (color STN) to set the division value.

2. LCDTiming1 (Vertical Axis Panel Control Register)

LCDTiming1 is the register to control the following:

- Number of lines per panel (LPP)
- Vertical sync pulse width (VSW)
- Vertical front porch (VFP) period
- Vertical back porch (VBP) period

Address = (0xF420_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	VBP	R/W	0x00	Setting the number of vertical back porch lines 0x00 to 0xFF
[23:16]	VFP	R/W	0x00	Setting the number of vertical front porch lines 0x00 to 0xFF
[15:10]	VSW	R/W	0y000000	Setting the number of vertical sync pulse lines (Setting value + 1) 0y000000 to 0y111111
[9:0]	LPP	R/W	0y0000000000	Setting the number of lines per panel (Setting value + 1) 0y0000000000 to 0y1111111111

[Description]

a. <VBP>

Vertical back porch refers to the number of non-active lines at the start of each frame after the vertical sync signal. This 8-bit VBP field is used to specify the number of line clocks inserted at the start point of each frame. VBP generates 0 to 255 line clock cycles.

b. <VFP>

Vertical front porch refers to the number of non-active lines at the end of each frame before the vertical sync signal. For STN displays, setting a value other than 0 reduces contrast. VFP generates 0 to 255 line clock cycles.

c. <VSW>

Vertical sync pulse width refers to the number of horizontal sync lines. For STN displays, a small value (such as 0) must be programmed. Setting a greater value results in lower contrast.

d. <LPP>

The LPP field specifies the number of active lines per LCD panel.

The actual value counted is the value set in this field incremented by one. Therefore, 1 to 1024 lines can be specified.

For dual-panel displays, this field should be programmed for the upper panel and the lower panel individually.

The figure below shows the operation mode setting (LCDTiming2<IVS> = 1, LCDTiming2<IHS> = 0) as an example:

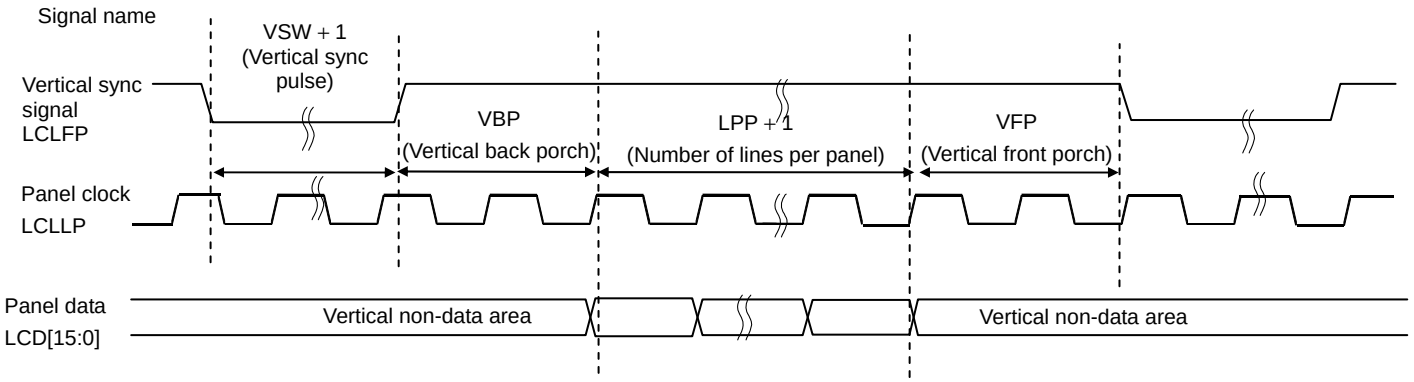


Figure 3.18.4 Basic operation of Vertical control

3. LCDTiming2 (Clock and Signal Polarity Control Register)

LCDTiming2 is the read/write register to control the LCDC timing.

Address = (0xF420_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:27]	PCD_HI	R/W	0y00000	Value set for the higher 5 bits of panel clock frequency division 0y00000 to 0y11111
[26]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[25:16]	CPL	R/W	0y0000000000	Number of clocks per line 0y0000000000 to 0y1111111111
[15]	–	–	Undefined	Read as undefined. Write as zero.
[14]	IOE	R/W	0y0	Data enable signal invert setting (Note1) 0y0: LCLAC output “H” active in TFT mode 0y1: LCLAC output “L” active in TFT mode
[13]	IPC	R/W	0y0	Panel clock signal edge selection 0y0: LCLCP rising edge 0y1: LCLCP falling edge
[12]	IHS	R/W	0y0	Horizontal synchronization signal Invert setting 0y0: LCLLP pin “H” active 0y1: LCLLP pin “L” active
[11]	IVS	R/W	0y0	Vertical synchronization signal Invert setting 0y0: LCLFP pin “H” active 0y1: LCLFP pin “L” active
[10:6]	ACB	R/W	0y00000	Bias invert frequency (Setting + 1) (Note2) 0y00000 to 0y11111
[5]	Reserved	–	Undefined	Read as undefined. Write as zero.
[4:0]	PCD_LO	R/W	0y00000	Value set for the lower 5 bits of panel clock frequency division

Note1: This bit is usable TFT mode only.

Note2: This bit is usable STN mode only.

[Description]

a. <PCD_HI>

The PCD_HI field is used to generate the LCD panel clock frequency by dividing the HCLK frequency. A 10-bit divisor can be specified by combining PCD_HI (upper 5 bits) and PCD_LO (lower 5 bits). LCLCP = HCLK/ (PCD + 2).

b. <CPL>

The CPL field specifies the actual number of LCLCP clocks per line in the LCD panel. This value is obtained by dividing the number of pixels per line by 1, 4, 8 or 8/3, and then subtracting one from the quotient. To allow the LCD controller to function properly, this field needs to be programmed properly in addition to PLL.

Panel Type		Bus Width	CPL Calculation Formula
TFT		—	$CPL = \left\lceil \frac{(\text{Number of pixels per line})}{1} \right\rceil - 1$
STN	Monochrome	4	$CPL = \left\lceil \frac{(\text{Number of pixels per line})}{4} \right\rceil - 1$
		8	$CPL = \left\lceil \frac{(\text{Number of pixels per line})}{8} \right\rceil - 1$
	Color	8	$CPL = \left\lceil \frac{(\text{Number of pixels per line})}{\frac{8}{3}} \right\rceil - 1$ Note: Round up all digits to the right of the decimal point.

c. <IOE>

The IOE bit specifies the polarity of the data enable signal.

The data enable signal is output on the LCLAC pin to notify the LCD panel when valid display data is available. It can be used only for TFT displays.

d. <IPC>

This bit set the panel clock edge.

e. <IHS>

This bit set the polarity of Horizontal sync signal.

f. <IVS>

This bit set the polarity of Vertical sync signal.

g. <ACB>

The ACB field specifies the bias inversion period. For STN displays, the bias polarity needs to be inverted periodically to prevent degradation in the LCD due to the accumulation of DC electrical charge. The bias inversion period, which is specified in lines, is the value set in this field incremented by one. Therefore, it can be set to 1 to 32 lines. This field can be used only for STN displays.

h. <PCD_LO>

The lower 5 bits of the value set for panel clock frequency division setting (10 bits)

Note: There are limitations on the minimum values usable for the panel clock divider in the STN mode.

- Single panel color mode: PCD = 1 (LCLCP = HCLK/3)
- Dual panel color mode: PCD = 4 (LCLCP = HCLK/6)
- Single panel monochrome 4-bit interface mode: PCD = 2 (LCLCP = HCLK/4)
- Dual panel monochrome 4-bit interface mode: PCD = 6 (LCLCP = HCLK/8)
- Single panel monochrome 8-bit interface mode: PCD = 6 (LCLCP = HCLK/8)
- Dual panel monochrome 8-bit interface mode: PCD = 14 (LCLCP = HCLK/16)

4. LCDTiming3 (Line End Control Register)

Address = (0xF420_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:17]	–	–	Undefined	Read as undefined. Write as zero.
[16]	LEE	R/W	0x00	Enable for Line end signal CLLE 0y0: Disable (Fixed to “L”) 0y1: Enable
[15:7]	–	–	Undefined	Read as undefined. Write as zero.
[6:0]	LED	R/W	0y000000	Delay value for CLLE output (Setting value + 1) 0y000000 to 0y111111

[Description]

a. <LEE>

After this signal is enabled, CLLE outputs a positive pulse of 4-HCLK period after the end of each line.

If the line end signal is disabled, this signal is held at “L” at all times.

b. <LED>

Sets the delay value for CLLE output.

5. LCDUPBASE (Upper Panel Frame Base Address Register)

This is the color LCD DMA base address register.

Address = (0xF420_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	LCDUPBASE	R/W	0x00000000	Register to set color LCD DMA base addresses. 0x00000000 to 0x3FFFFFFF
[1:0]	–	–	Undefined	Read as undefined. Write as zero.

6. LCDLPBASE (Lower Panel Frame Base Address Register)

Address = (0xF420_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	LCDLPBASE	R/W	0x00000000	Register to set color LCD DMA base addresses. 0x00000000 to 0x3FFFFFFF
[1:0]	–	–	Undefined	Read as undefined. Write as zero.

LCDUPBASE and LCDLPBASE set the first address of display RAM.

LCDUPBASE is used for the following:

- TFT display
- Single panel STN display
- Higher-order panel of dual panel STN display

LCDLPBASE is used for the lower-order panel of dual panel STN display.

Programmers need to setting LCDUPBASE (and LCDLPBASE of dual panel) before enabling LCDC.

Each address setting set to the full-address of 32-bit ([31:0]). However, its lower 2-bit are ignored, it is set as word unit (4-byte)setting.

7. LCDIMSC (Interrupt Mask Set/Clear Register)

Address = (0xF420_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined. Write as zero.
[4]	MBERRINTRENB	R/W	0y0	AHB master error interrupt enable 0y0: Disable 0y1: Enable
[3]	VCOMPINTRENB	R/W	0y0	Vertical sync. interrupt enable 0y0: Disable 0y1: Enable
[2]	LNBUINTRENB	R/W	0y0	Next base address update interrupt enable 0y0: Disable 0y1: Enable
[1]	FUFINTRENB	R/W	0y0	FIFO underflow interrupt enable 0y0: Disable 0y1: Enable
[0]	–	–	Undefined	Read as undefined. Write as zero.

LCDIMSC is the interrupt enable register. Setting the bits in this register passes the corresponding values in the original interrupt LCDRIS bit to the LCDMIS register.

8. LCDControl (LCDC Control Register)

Address = (0xF420_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:17]	–	–	Undefined	Read as undefined. Write as zero.
[16]	WATERMARK	R/W	0y0	LCD DMA FIFO watermark level 0y0:Requests DMA when space of 4 words or more occurs in either of the two FIFOs. 0y1:Requests DMA when space of 8 words or more occurs in either of the two FIFOs.
[15:14]	–	–	Undefined	Read as undefined. Write as zero.
[13:12]	LcdVComp	R/W	0y00	Interrupt occurrence timing 0y00: At vertical sync start 0y01: At back porch start 0y10: At display data start 0y11: At front porch start
[11]	Reserved	R/W	0y0	Write as 1.
[10]	Reserved	R/W	0y0	Write as 0.
[9]	Reserved	R/W	0y0	Write as 0.
[8]	BGR	R/W	0y0	BGR system selected RGB 0y0: RGB normal output 0y1: BGR red/blue swap
[7]	LcdDual	R/W	0y0	STN panel select 0y0: Single panel LCD 0y1: Dual panel LCD
[6]	LcdMono8	R/W	0y0	Selects the monochrome STN LCD parallel bit. 0y0: 4-bit interface for monochrome LCD 0y1: 8-bit interface for monochrome LCD
[5]	LcdTFT	R/W	0y0	Selects the panel used for LCD. 0y0: STN panel 0y1: TFT panel
[4]	LcdBW	R/W	0y0	Selects monochrome or color for STN LCD. 0y0: Color 0y1: Monochrome
[3:1]	LcdBpp	R/W	0y000	Number of LCD bits per pixel: 0y000 = 1 bpp 0y001 = 2 bpp 0y010 = 4 bpp 0y011 = 8 bpp 0y100 = 16 bpp 0y101 = Reserved 0y110 = Reserved 0y111 = Reserved
[0]	LcdEn	R/W	0y0	LCD controller enable: 0y0: Disable 0y1: Enable

[Description]

a. <WATERMARK>

LCD DMA FIFO watermark level

0y0: Requests DMA when space of 4 words or more occurs in either of the two FIFOs.

0y1: Requests DMA when space of 8 words or more occurs in either of the two FIFOs.

b. <LcdVComp>

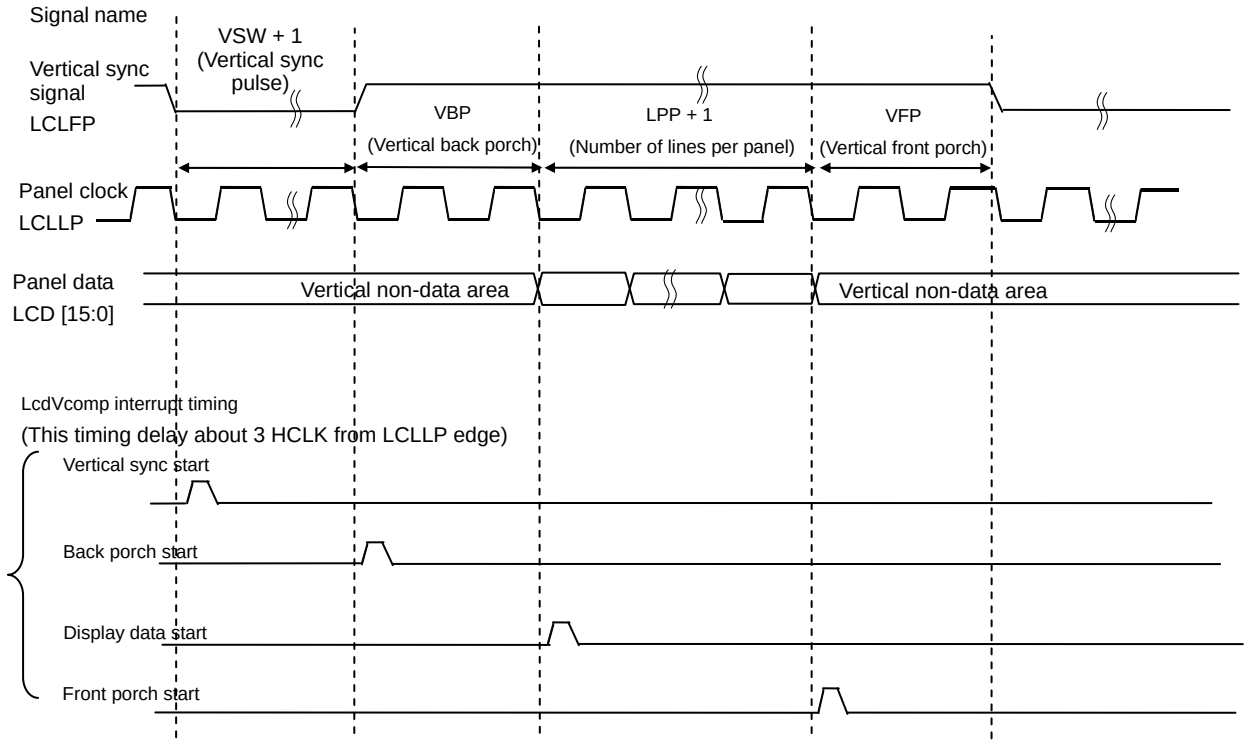
0y00: At vertical sync start

0y01: At back porch start

0y10: At display data start

0y11: At front porch start

Following timing chart shows interrupt generation timing .



c. <BGR>

BGR system selected RGB

0y0: RGB normal output

0y1: BGR red/blue swap

The swap function is to replace blue with red data as shown in following table.

Intended bit	<BGR>=0	<BGR>=1
[15]	Brightness bits	Brightness bits
[14:10]	B[4:0]	R[4:0]
[9:5]	G[4:0]	G[4:0]
[4:0]	R[4:0]	B[4:0]

However, this MCU has a LCDDA function. When using this function, the actual alignment of data is defined as shown in following table.

Intended bit	Data content
[15:11]	B[4:0]
[10:5]	G[5:0]
[4:0]	R[4:0]

Therefore, the LCDDA does not allow the use of the swap function.

d. <LcdDual>

STN panel select

0y0: Single panel LCD

0y1: Dual panel LCD

e. <LcdMono8>

This shows that monochrome LCD uses the 8-bit interface. This bit controls which of 4-bit or 8-bit parallel interface is used for monochrome STN LCD. In other modes, 0 needs to be programmed.

f. <Lcd TFT>

0y0 = Shows that LCD is STN display using the gray scaler.

0y1 = Shows that LCD is TFT using no gray scaler.

g. <LcdBW>

This shows that STN LCD is monochrome (black and white).

0y0 = Shows that STN LCD is color.

0y1 = Shows that STN LCD is monochrome.

This bit has no meaning in the TFT mode.

h. <Lcd Bpp>

This bit set the Number of LCD bits per pixel.

0y000 = 1 bpp

0y001 = 2 bpp

0y010 = 4 bpp

0y011 = 8 bpp

0y100 = 16 bpp

0y101 = Reserved

0y110 = Reserved

0y111 = Reserved

i. <LcdEn>

This bit set operation of the LCD controller.

0y0 = Stop

The LCD signals LCLLP, LCLCP, LCLFP, LCLAC, and LCLLE are disabled (Fixed to “L”).

0y1 = Operate

The LCD signals LCLLP, LCLCP, LCLFP, LCLAC, and LCLLE are enabled (active).

Note1: After each regsiter of LCDC have been completely prepared, set <LcdEn> to 1.

Note2: If you set to the stop state (set to 0), LCD signals (LCLLP, LCLCP, LCLFP, LCLAC and LCLLE) are always fixed to “L”. Please note the signal set by negative-true logic.

9. LCDRIS (Raw Interrupt Status Register)

Address = (0xF420_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined.
[4]	MBERROR	RO	0y0	Request for AMBA AHB master bus error interrupt 0y0: No 0y1: Yes
[3]	Vcomp	RO	0y0	Request for vertical sync. interrupt 0y0: No 0y1: Yes
[2]	LNBU	RO	0y0	Request for LCD next address base update interrupt 0y0: No 0y1: Yes
[1]	FUF	RO	0y0	Request for FIFO underflow interrupt 0y0: No 0y1: Yes
[0]	–	–	Undefined	Read as undefined.

[Description]

a. <MBERROR >

AMBA AHB master bus error status. This is set if the AMBA AHB master detects a bus error response from a slave.

b. <Vcomp>

Vertical sync. This is set if any one of the four vertical areas selected from the LCDControl [13:12] register reaches the timing.

c. <LNBU>

LCD next address base update. This depends on the mode and is set when the current base address register is updated by the net address register properly.

d. <FUF>

FIFO underflow. This is set if either higher- or lower-order DMA FIFO is read and accessed when it is empty, which is the condition of triggering the underflow condition.

10. LCDMIS (Masked Interrupt Status Register)

LCDMIS is a read-only register. This register serves as the logical AND for each bit of the LCDRIS register and the LCDIMSC (Enable) register. The logical ORs of all interrupts are given to the system interrupt controller.

Address = (0xF420_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined. Write as zero.
[4]	MBERRORINTR	RO	0y0	AMBA AHB master bus error status bit 0y0: Clear 0y1: Interrupt requested
[3]	VCOMPINTR	RO	0y0	Vertical sync. interrupt status bit 0y0: Clear 0y1: Interrupt requested
[2]	LNBUINTR	RO	0y0	LCD next address base update status bit 0y0: Clear 0y1: Interrupt requested
[1]	FUFINTR	RO	0y0	FIFO underflow status bit 0y0: Clear 0y1: Interrupt requested
[0]	–	–	Undefined	Read as undefined.

11. LCDICR (Interrupt Clear Register)

Address = (0xF420_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:5]	–	–	Undefined	Read as undefined. Write as zero.
[4]	Clear MBERROR	WO	0y0	Clears AMBA AHB master bus error interrupt request flags 0y0: No change 0y1: Clear
[3]	Clear Vcomp	WO	0y0	Clears vertical sync. interrupt request flags. 0y0: No change 0y1: Clear
[2]	Clear LNBU	WO	0y0	Clears LCD next address base update interrupt request flags. 0y0: No change 0y1: Clear
[1]	Clear FUF	WO	0y0	Clears FIFO underflow interrupt request flags. 0y0: No change 0y1: Clear
[0]	–	–	Undefined	Read as undefined. Write as zero.

12. LCDUPCURR (Upper Panel Current Address Value Registers)

Address = (0xF420_0000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	LCDUPCURR	RO	0x00000000	Approximate values of higher-order panel data DMA addresses

13. LCDLPCURR (Lower Panel Current Address Value Registers)

Address = (0xF420_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	LCDLPCURR	RO	0x00000000	Approximate values of lower-order panel data DMA addresses

The LCDUPCURR register and the LCDLPCURR register retain the approximate values of higher- and lower-order panel data DMA addresses during read. These registers can change all the time. Be careful when using them.

14. LCDPalette (Color Palette Register)

One word (32 bits) of palette RAM data equals to two pixels of data. Therefore, the lowest-order bit of pixel data is used to select either the higher 16 bits or lower 16 bits of the palette RAM.

A palette structured with R:5 bits, G:5 bits, B:5 bits , and brightness bits is structured with a dual port RAM of 128 × 32 bits. Therefore, two-pixel entry into the palette can be written in 1 word.

In the TFT mode, all palette data is used; in the monochrome STN mode, only red palette R[4:1] is used (bit0 not used); in the STN color mode, red, green, and blue [4:1] are used (bit0 not used).

To support the BGR data system, this red and blue pixel data can be swapped using the control register bit.

In the 16 bpp TFT mode, palettes are bypassed so that the pixel serializer's output can be directly used as TFT panel data.

A RAM palette supports 256 entries x 16 bits at maximum. Therefore, TFT and color STN palettes can support up to 8-bpp data.

Address = (0xF420_0000) + ((0x0200) to (0x03FC))

Bit	Bit Symbol	Type	Reset Value	Description
[31]	I	R/W	0y0	Brightness/unused
[30:26]	B[4:0]	R/W	0y00000	Blue palette data setting
[25:21]	G[4:0]	R/W	0y00000	Green palette data setting
[20:16]	R[4:0]	R/W	0y00000	Red palette data setting
[15]	I	R/W	0y0	Brightness/unused
[14:10]	B[4:0]	R/W	0y00000	Blue palette data setting
[9:5]	G[4:0]	R/W	0y00000	Green palette data setting
[4:0]	R[4:0]	R/W	0y00000	Red palette data setting

[Description]

a. <I>

Brightness bit. Using as the LSB of R, B, and B inputs to 6:6:6 TFT display, this bit can set two ways of brightness in each color. Doubling the number of colors, the data becomes 64 K.

b. <B>

Blue palette data.

c. <G>

Green palette data.

d. <R>

For STN display, only four MSB bits (bit 4:1) are used. For monochrome display, only red palette data is used. All palette registers are arranged with the same bits.

3.18.3.1 Multiplexing LCD Panel Signals

While LCLLP, LCLAC, LCLFP, LCLCP, and LCLLE are common, the LCLD [15:0] bus has the eight operation modes supporting the following:

- TFT 16-bit interface
- Color STN single panel
- Color STN dual panel
- 4-bit monochrome STN single panel
- 4-bit monochrome STN dual panel
- 8-bit monochrome STN single panel
- 8-bit monochrome STN dual panel

Note:
CUSTN = Color STN dual higher-order panel data signal / Color STN single panel data signal
CLSTN = Color STN dual lower-order panel data signal
MUSTN = Monochrome STN dual higher-order panel data signal / Monochrome STN single panel data signal
MLSTN = Monochrome STN dual lower-order panel data signal

Table3.18.5 LCD TFT panel signal multiplexing [TFT 16bit Interface]

External pin	Color bit allocation	Pallet & RGB-BGR conversion	VRAM bit allocation			
			32bit bus RAM		16bit bus RAM	
			Address	Data bit	Address	Data bit
CLD[0]	BLUE[4]	Color Pallet Bypass RGB-BGR Not Support	n	D31	n+2	D15
CLD[17]	BLUE[3]			D30		D14
CLD[16]	BLUE[2]			D29		D13
CLD[15]	BLUE[1]			D28		D12
CLD[14]	BLUE[0]			D27		D11
CLD[13]	GREEN[5]			D26		D10
CLD[11]	GREEN[4]			D25		D9
CLD[10]	GREEN[3]			D24		D8
CLD[9]	GREEN[2]			D23		D7
CLD[8]	GREEN[1]			D22		D6
CLD[7]	GREEN[0]			D21		D5
CLD[5]	RED[4]			D20		D4
CLD[4]	RED[3]			D19		D3
CLD[3]	RED[2]			D18		D2
CLD[2]	RED[1]			D17		D1
CLD[1]	RED[0]			D16		D0
CLD[0]	BLUE[4]			D15	n	D15
CLD[17]	BLUE[3]			D14		D14
CLD[16]	BLUE[2]			D13		D13
CLD[15]	BLUE[1]			D12		D12
CLD[14]	BLUE[0]			D11		D11
CLD[13]	GREEN[5]			D10		D10
CLD[11]	GREEN[4]			D9		D9
CLD[10]	GREEN[3]			D8		D8
CLD[9]	GREEN[2]			D7		D7
CLD[8]	GREEN[1]			D6		D6
CLD[7]	GREEN[0]			D5		D5
CLD[5]	RED[4]			D4		D4
CLD[4]	RED[3]			D3		D3
CLD[3]	RED[2]			D2		D2
CLD[2]	RED[1]			D1		D1
CLD[1]	RED[0]			D0		D0

Note: In the case of using 16bitTFT, Intensity bit can't be used. And the swap of RGB and BGR isn't supported.

Table3.18.6 LCD STN panel signal multiplexing

External pin	Color STN single panel	Color STN dual panel	4-bit mono STN single panel	4-bit mono STN dual panel	8-bit mono STN single panel	8-bit mono STN dual panel
CLD[15]	Reserved	CLSTN[7]	Reserved	Reserved	Reserved	MLSTN[7]
CLD[14]	Reserved	CLSTN[6]	Reserved	Reserved	Reserved	MLSTN[6]
CLD[13]	Reserved	CLSTN[5]	Reserved	Reserved	Reserved	MLSTN[5]
CLD[12]	Reserved	CLSTN[4]	Reserved	Reserved	Reserved	MLSTN[4]
CLD[11]	Reserved	CLSTN[3]	Reserved	MLSTN[0]	Reserved	MLSTN[3]
CLD[10]	Reserved	CLSTN[2]	Reserved	MLSTN[1]	Reserved	MLSTN[2]
CLD[9]	Reserved	CLSTN[1]	Reserved	MLSTN[2]	Reserved	MLSTN[1]
CLD[8]	Reserved	CLSTN[0]	Reserved	MLSTN[3]	Reserved	MLSTN[0]
CLD[7]	CUSTN[7]	CUSTN[7]	Reserved	Reserved	MUSTN[7]	MUSTN[7]
CLD[6]	CUSTN[6]	CUSTN[6]	Reserved	Reserved	MUSTN[6]	MUSTN[6]
CLD[5]	CUSTN[5]	CUSTN[5]	Reserved	Reserved	MUSTN[5]	MUSTN[5]
CLD[4]	CUSTN[4]	CUSTN[4]	Reserved	Reserved	MUSTN[4]	MUSTN[4]
CLD[3]	CUSTN[3]	CUSTN[3]	MUSTN[3]	MUSTN[3]	MUSTN[3]	MUSTN[3]
CLD[2]	CUSTN[2]	CUSTN[2]	MUSTN[2]	MUSTN[2]	MUSTN[2]	MUSTN[2]
CLD[1]	CUSTN[1]	CUSTN[1]	MUSTN[1]	MUSTN[1]	MUSTN[1]	MUSTN[1]
CLD[0]	CUSTN[0]	CUSTN[0]	MUSTN[0]	MUSTN[0]	MUSTN[0]	MUSTN[0]

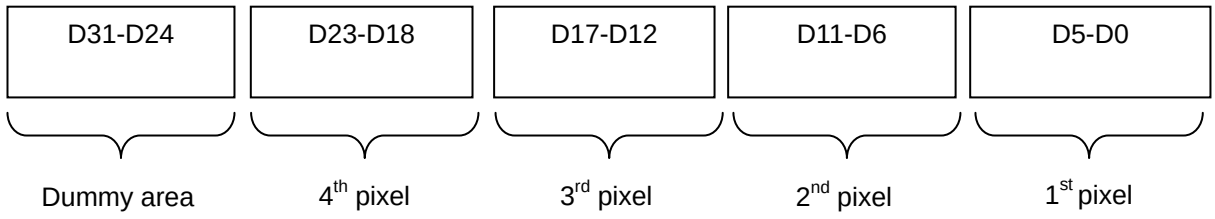
3.18.4 LCD Controller Option Function (LCDCOP)

The LCD controller contained in this LSI supports 64-level grayscale LCD display as an optional feature.

Table 3.18.7 LCDC Optional Feature

Type of LCD panel		STN (Dual/Single)
Number of colors displayable simultaneously		64-level grayscale
Number of available horizontal lines		Max : 640
Number of available vertical lines		Max : 480
Data bus width of transfer destination (LCD driver)		4/8 bits
Input data		LCDC output data in TFT 16-bit mode
Data type		Little-endian supported
Connected pins	LD[7 :0] / LD[3:0] pins	Data bus dedicated to the LCD driver
	LCLAC pin	AC bias signal (frame signal)
	LCLLP pin	Horizontal sync pulse
	LCLFP pin	Vertical sync pulse
	LCLCP pin	Clock for LCDD data latch (panel clock)
	LCLLE pin	Not used basically

To support STN 64-level grayscale mode, VRAM uses the following format. And the external 4-bit and 8-bit LD data buses are supported.



3.18.4.1 Block Diagrams

Figure 3.19.5 shows the schematic block diagram of the LCDC and LCDCOP.

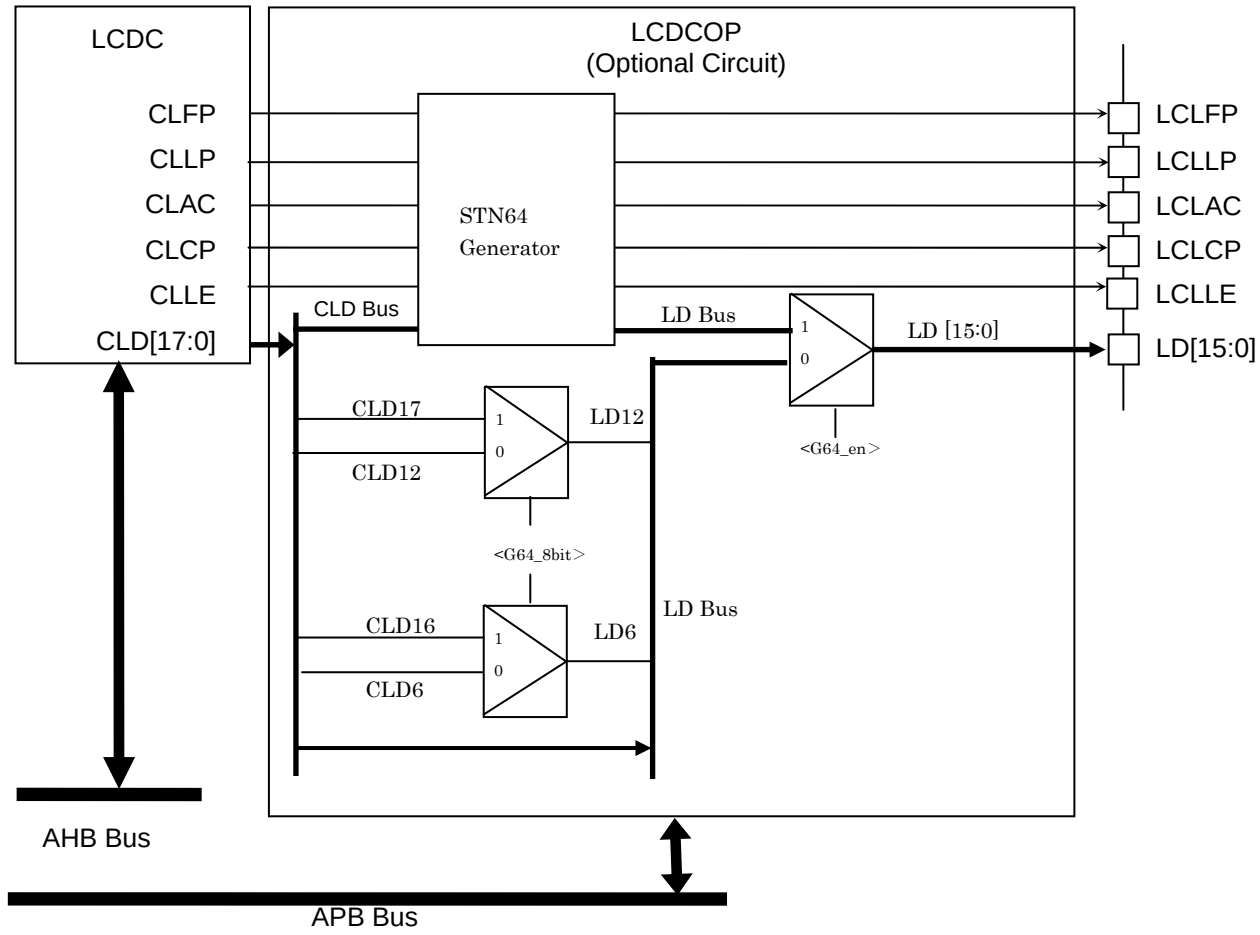


Figure 3.18.5 LCDCOP Block Diagram

3.18.4.2 Description of Registers

The following lists the registers:

Base address = 0xF00B_0000

Register Name	Address (base+)	Description
STN64CR	0x0000	LCDC Option Control Register for STN 64

15. STN64CR (LCDC Option Control Register)

Address = (0xF00B_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	NoSpikeMode	R/W	0y0	Delete noise of CLCP of LCDC 0y0 : invalid 0y1 : valid
[6]	Reserved	R/W	0y0	Read as undefined. Write as zero.
[5]	CLFP_Inv	R/W	0y0	Invert vertical synchronization(=VIS of LCDC) 0y0 : LCLFP pin HIGH active 0y1 : LCLFP pin LOW active
[4]	:CLLP_Inv	R/W	0y0	Invert horizontal synchronization (=HIS of LCDC) 0y0 : LCLLP pin HIGH active 0y1 : LCLLP pin LOW active
[3]	CLAC_Inv	R/W	0y0	Invert output enable (= IOE of LCDC) 0y0 : LCLAC output HIGH active in TFT mode 0y1 : LCLAC output LOW active in TFT mode
[2]	LCP_Inv	R/W	0y0	Invert panel clock (=IPC of LCDC) 0y0 : LCLCP rising edge 0y1 : LCLCP falling edge
[1]	G64_8bit	R/W	0y0	Refer to the table of Table 3.18.8 The setting of STN 64 gray and TFT16 bit .
[0]	G64_en	R/W	0y0	

Note: When using STN 64-level grayscale mode, be sure to set LCLFP, LCLLP, LCLAC, and LCLCP identically with the settings in the LCDC.

Table 3.18.8 The setting of STN 64 gray and TFT16 bit

Mode	Register setting		Note
	<G64_en>	<G64_8bit>	
STN64 gray	0y1 : Use STN 64 gray circuits	0y0 : External 4bit LD Bus 0y1 : External 8bit LD Bus	
TFT16bit	0y0 : Not Use STN 64 gray circuits	0y1 : External LD12=CLD17 External LD6 =CLD16	If LD[15:0] Function of Port P and Port V are setted, <G64_8bit> need be seted to 0y1, to ouput CLD17, CLD16
others	0y0 : Not Use STN 64 gray circuits	0y0 : External LD12=CLD12 External LD6 =CLD6	

Note1: For the LD bus switching mechanism, see Figure 3.18.5 LCDCOP Block Diagram LCDCOP Block Diagram.

Note2: For information about external 16-bit TFT signals, see Table3.18.5 LCD TFT panel signal multiplexing [TFT 16bit Interface]

3.19 LCD Data Process Accelerator (LCDDA)

This microcontroller incorporates the LCD Data Process Accelerator function (LCDDA) as an auxiliary function for display.

The LCDDA supports the scaler function that scales up/down display data including the filter (Bi-Cubic method) processing, and the image rotation function that rotates and mirror-inverts display data function, as well as the function of superimposing two images (αBlend, Picture in Picture, Superimposing font).

The following lists the functions:

Table 3.19.1 LCDDA functions

Function	Description
Scaler function	Scale up: Can scale up to the magnification of 256/n: (n = 1 to 255). Can scale up independently in horizontal/vertical directions. Filtering by Bi-Cubic method is possible in scaled up images.
	Scale down: Can scale down to the magnification of 256/(n × m): (n = 1 to 255, m = 1 to 16). Can scale down independently in horizontal/vertical directions. Filtering by Bi-Cubic method is possible in scaled down images.
Image rotation function	90° / 180° / 270° / horizontal mirror reversal / vertical mirror reversal possible
Image Blend function	Function of superimposing two images (Picture in Picture)
	Superimposing (α-Blend) possible adjusting the gray level of two images
	Font Draw function for Font Data represented in binary (monochrome)

These circuits, which operate as other circuits completely separate from the LCD controller, all use the Copy Back (write back) method. Image data is processed and the data is written into the display Ram of the LCDC. Then the LCDC displays the processed data.

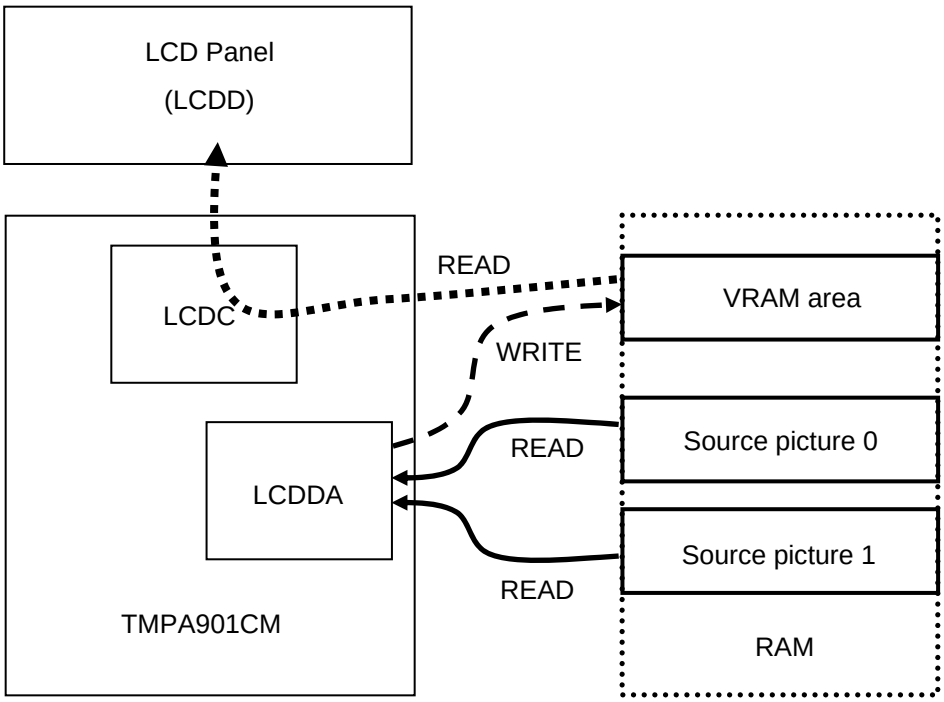


Figure 3.19.1 Image of LCDDA operation (Example of Blend function)

3.19.1 Block Diagrams

The block diagram of LCDDA is shown below:

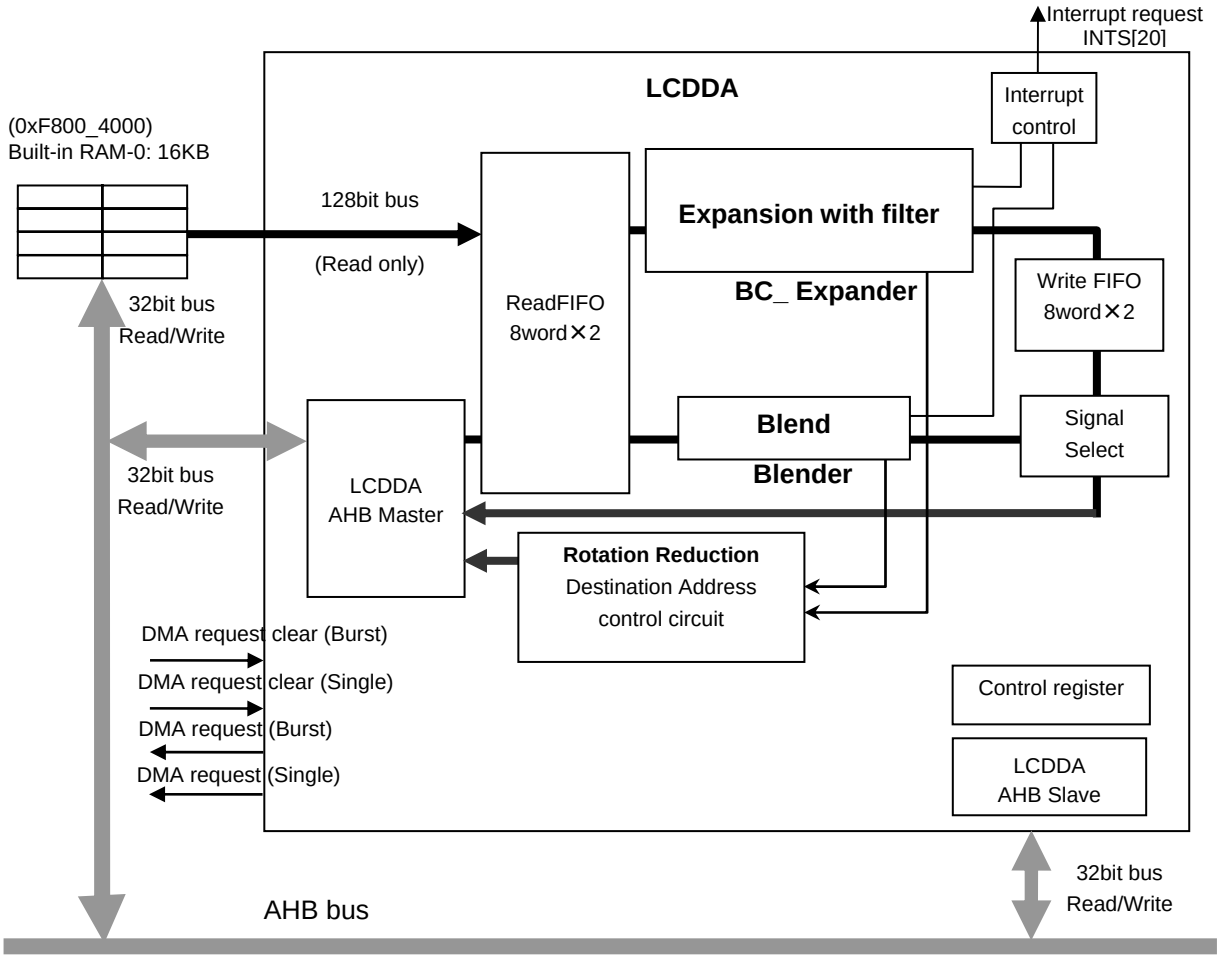


Figure 3.19.2 LCDDA block diagram

The LCDDA is mainly broken down into eight blocks:

- "BC_Expander" where scaling up/down is performed using the Bi-Cubic method
- "Blender" where Blend processing is performed
- "Read FIFO buffer" where source data is accumulated
- "Write FIFO buffer" where destination data is accumulated
- "Transfer address control circuit" where rotation / simple scaling down processing is performed
- "AHB slave block" where the access from the AHB bus to control registers is controlled
- "AHB master block" where the data access to the AHB bus is controlled
- "Interrupt control block" where interrupts are generated by monitoring processing completion and error occurrence

3.19.2 Description of Operation

This section describes the functions that the LCDDA has:

3.19.2.1 Scaler Function

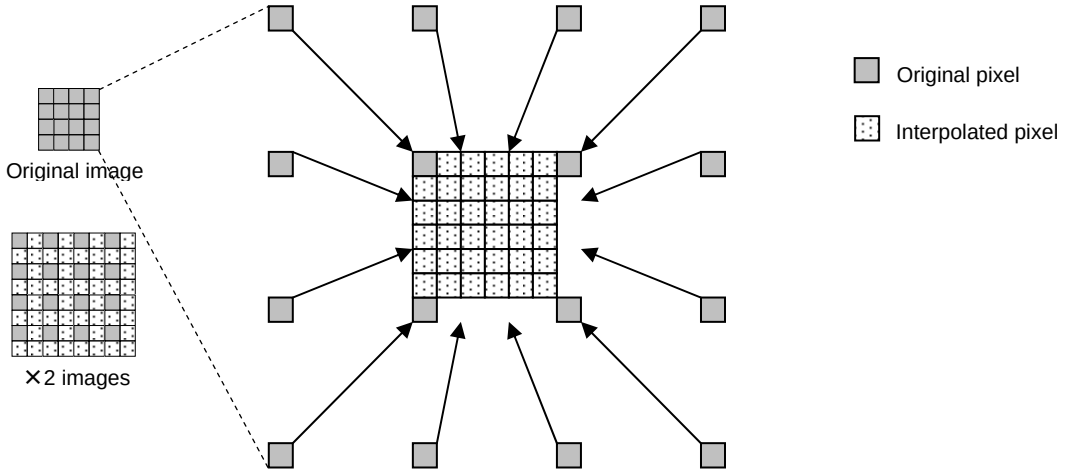
Table 3.19.2 Scaler function

Function	Details of function	Description / Standard
Scale-up function	Scale-up rate/Interpolation data quantity	256/n: (Can set to the magnification of 256/N: N = 1 to 255: Setting to 0 is disabled.) Can set independently in horizontal/vertical directions
	Supportable data format	Digital RGB Note: YUV-format data is not supported.
	Number of supportable image colors	64-K color (16 bpp) Note: Monochrome, monochrome gray-level, and the color of other color numbers are not supported.
	Supportable image size	・ORG image Horizontal: Max. 510 pixels Vertical: No particular limitations ・Scale-up image Horizontal: Max. 1024 pixels Vertical: No particular limitations Note: The maximum display size supported by this microcontroller's LCDC differs with the display panel. Please refer to section 3.19 LCD controller.
	Correction function	Period point correction function, termination point correction function provided
Scale-down function	Scale-down rate/Interpolation data quantity	Can interpolate 255 points of data between original pixels. (Can set independently in horizontal/vertical directions) Can output 255 points of data between original pixels. (Can set to the magnification of 255/N/M: N = 1 to 255, M = 1 to 16)
	Supportable data format	Digital RGB Note: YUV-format data is not supported.
	Number of supportable image colors	64-K color (16 bpp) Monochrome, monochrome gray-level, and the color of other color numbers are not supported.
	Supportable image size	・ORG image Horizontal: Max. 511 pixels Vertical: No particular limitations ・Scale-up image Horizontal: Max. 1024 pixels Vertical: No particular limitations Note: The maximum display size supported by this microcontroller's LCDC differs with the display panel. Please refer to section 3.19 LCD controller.
	Correction function	Period point correction function, termination point correction function provided

3.19.2.2 Mechanism of Scaler Processing

1) Basic Configuration

The scaler function of the LCDDA can insert interpolation data of 255 points at maximum between original pixels using the Bi-Cubic method.



Interpolation data is generated using the original pixels of 4×4 around the area to be interpolated.

Figure 3.19.3 Data interpolation

In this method, image data of 256-magnification (8 bits) at maximum is calculated automatically by the H/W, from the image data of the "4×4 point" pixels around the area to be interpolated.

The pixels of 256 points (including the original pixels) generated from the original pixels can be output at every n_Step.

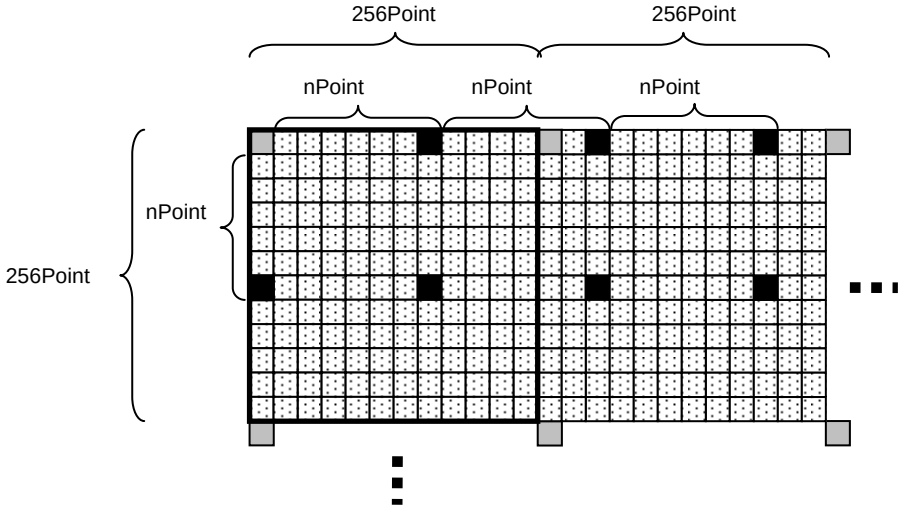


Figure 3.19.4 Scaling method

(Scaling up if the Step number is 1 to 255; scaling down if 257 or more)

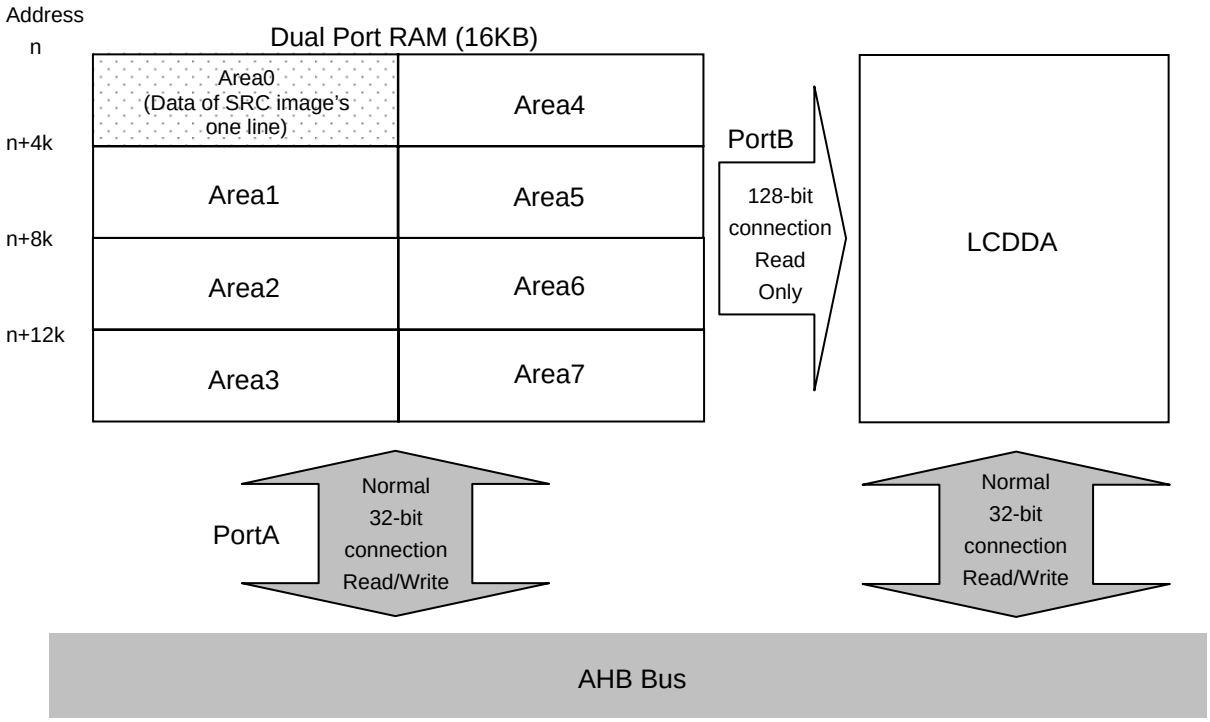


Figure 3.19.5 Connection with memory, and basic operation

To use the scaler function, original image data (RGB) needs to have been written into the dedicated DualPortRAM.

As described earlier, to generate interpolation data, original pixels of 4 × 4 are required. Therefore, the BC_Expander circuit can start generating interpolation data at the point when four lines of original image data become available.

The BC_Expander circuit of the LCDDA is connected with the 16-K-byte Dual Port RAM and the 128-bit-width bus (PortB).

This 16-K-byte Dual Port RAM, which can be used as a normal RAM, is connected to the AHB bus with the 32-bit-width bus (PortA).

In addition, this 16-K-byte Dual Port RAM is divided into 2-Kbyte × 8 areas in which one line of original image data is prepared. (Max 510 pixel: Dummy+510+Dummy).

The BC_Expander circuit can start calculation at the point when four lines of data (For example, area 0 to area 3) become available. After that, every time one line of data is added (area 4), four lines of data are prepared again (area 1 to area 4) to start next calculation.

In this manner, the area is looped to perform calculation.

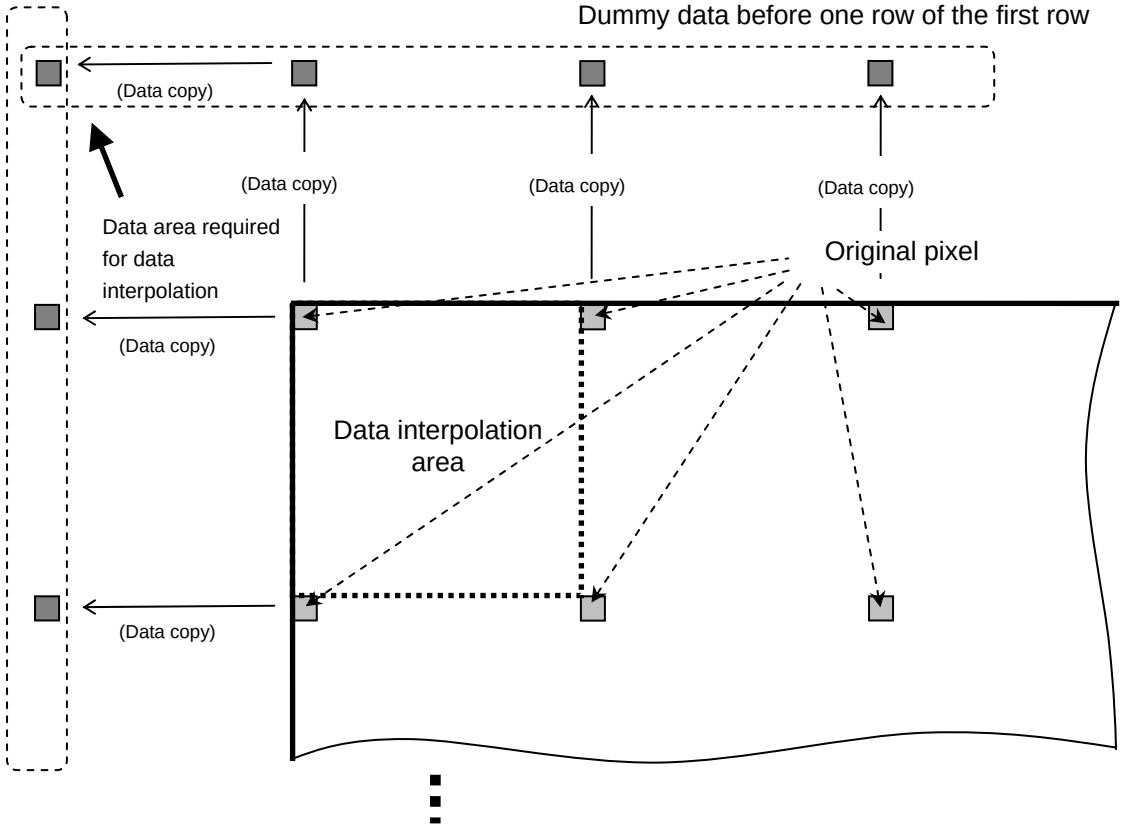
3.19.2.3 Correction Processing

The scaler function supports the function of correcting sampling points for scaling up/down processing. Using this function can express more natural images.

The correction function, if classified broadly, has two functions: the “edge data automatic addition” function and the “sampling correction” function.

1) Edge data automatic addition function

As described in the previous section, the original pixels of 4×4 around an area to be interpolated are required in the scaler function. Therefore, original pixels cannot be prepared in the pixels’ edge area. Dummy data of one line before the first line and one line after the last line or of one row before the first row and one row after the last row needs to be prepared.



This function can prepare this dummy data automatically.

The following shows examples of how to set original image data to the dual-port RAM connected to the LCDDA. The dual-port RAM also needs to have dummy data areas for edge processing.

Let us set original image data for 16-bit color displays with 510 pixels per line by using the edge data automatic addition function.

- The first line is handled as a dummy data area. It is therefore not necessary to set image data at addresses 0xF800_4000 through 0xF800_47FF.
- The first and last pixels in the second and subsequent lines are also handled as a dummy data area.

Original data: 510 pixels, 16-bit color

```
; Internal RAM area 0
0xF800_4800          ; Dummy data (no setting is required)
0xF800_4802          ; Valid data (set the first pixel data)
        .
        .
        .
0xF800_4BFC          ; Valid data (set the last pixel data)
0xF800_4BFE
 ~0xF800_4FFC        ; Dummy data (no setting is required)
```

Note: The maximum number of pixels per line is 510 pixels for 16-bit color displays.
To scale up or down an image larger than 510 pixels, it is necessary to divide the image.

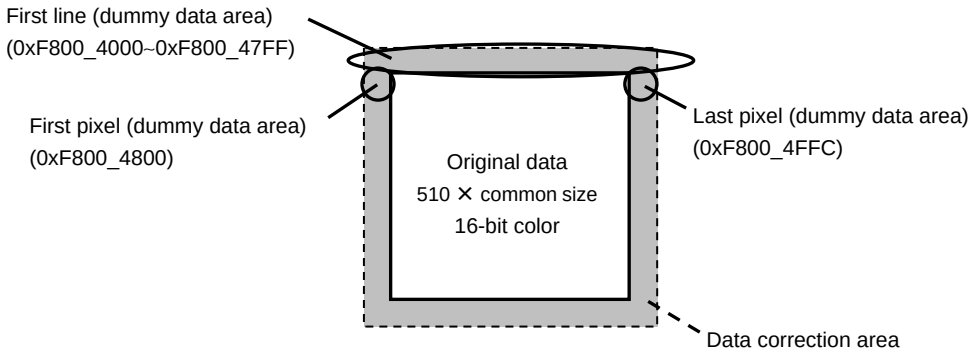


Figure 3.19.6 Image representation of original data (510 x common size, 16-bit color)

2) Sampling correction function

In the scaler function, scaling up/down at a magnification of 256/(n × m) is possible to the number of original pixels (n = 1 to 255, m = 1 to 16).

The scaling up method, set with the equation above, creates fractions of decimal places, which creates an error because the actual sampling point is set with integer.

Therefore, the accumulated errors in the whole image need to be corrected at an appropriate point.

This circuit supports the two correction functions: The "offset function" adds an offset to the first sampling point in the X direction; and the "period correction function," when a set sampling point comes to a certain point, the point is corrected to the original image point.

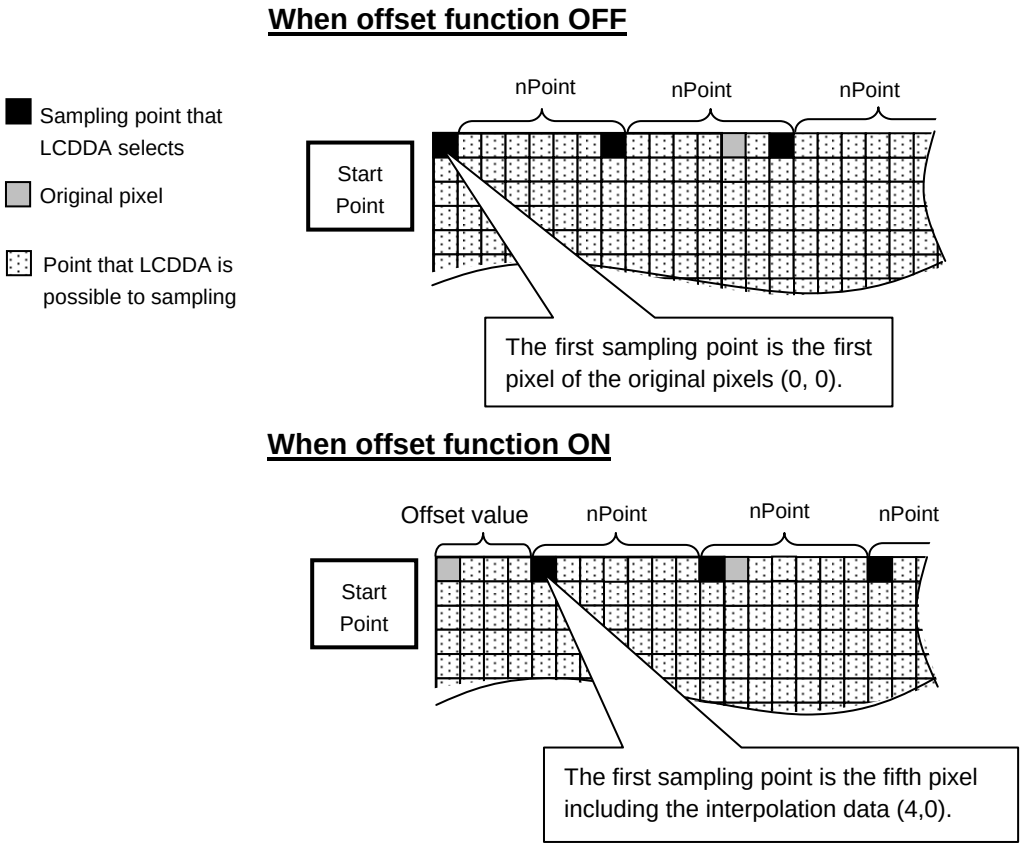
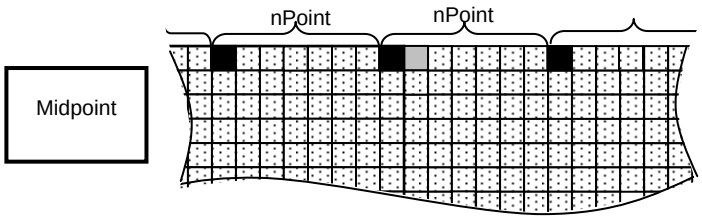


Figure 3.19.7 Offset function

When period correction function OFF



When period correction function ON

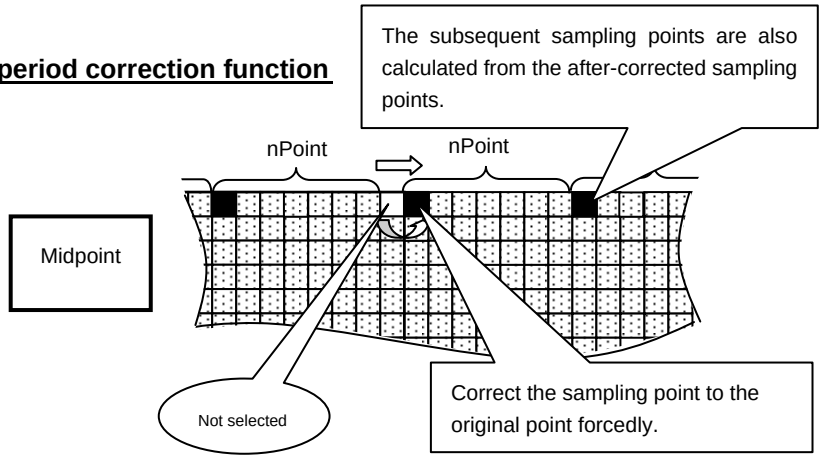


Figure 3.19.8 Period correction function

Scaling up processing examples

The following describes the examples of how to establish the setting for scaling up processing including correction processing:

- Number of original pixels in the X direction: OX pixel
- Number of after-scaled-up pixels in the X direction: GX pixel

(Sampling cycle)

= (Maximum number of interpolatable pixels) / (Number of after-scaled-up pixels)
= {(OX–1) × 256} ÷ GX

Example 1: For scaling up 128 pixels to 256 pixels

(127 × 256) ÷ 256
= (32512) ÷ 256
= 127

Example 2: For scaling up 128 pixels to 255 pixels

(127 × 256) ÷ 255
= (32512) ÷ 255
= 127.49...

Sampling cycles are specified in integers by dropping the fractional portion. This results in the same sampling cycle (127 = 0x7F) for Examples 1 and 2.

When data is sampled at every 127 pixels, sampling points occur as shown below.

Example 1 sampling point

First point: 0
Second point: 127
Third point: 254
| |
254th point: 32258
255th point: 32385

256th point: 32512

In Example 1, the 256th sampling point is 32512, which is the last pixel of the original image.

Since Example 2 uses the same sampling cycle, the 255th sampling point is 32385. Therefore, the pixels after 32385 are not used and discarded.

Example 2 Sampling points

First point:	0
Second point:	127
Third point:	254
｜	｜
254th point:	32258
255th point:	32385

* Pixels 32386 to 32512 are not used.

This produces a slightly off-center scaled-up image shifted to the left.
To realize more natural-looking scaling up, the correction function is utilized.

[Offset correction example]

The offset correction function adds an offset to the first sampling point so that unused pixel points are evened out from the center.

Move the first sampling point to move all the sampling points to the center.

Offset half of the difference between the last sampling point (255th point) and the last generable sampling point.

Example of simple offset correction (using Example 2 described earlier)

(Offset correction value) = { (Maximum number of interpolatable pixels) − (Sampling cycle after dropping the fractional portion) × GX} ÷ 2

= (32512–32385) ÷ 2 = 63.5

Since the offset value in the X direction must be an integer, it is set to 63 (oct) = 3F (hex).
Likewise, the offset value in the Y direction is set to 63 (oct) = 3F (hex).

3.19.2.4 Blend Processing Function

Table 3.19.3 Blend function

Function	Details of function	Description / Standard
BLEND function	α BLEND	Settable in 256 levels (0 to 255) for each image and specifically for RGB
	Color + monochrome BLEND	Two-valued (monochrome) data, FONT (1) data, and data other than FONT (0) can be each converted into 16-bit color RGB on the palette.
	Font	Superimposing monochrome font onto image
	Supportable data format	Digital RGB Note: YUV-format data is not supported.
	Number of supportable image colors	64-K color (16 bpp) Note: Monochrome, monochrome gray-level, and the color of other color numbers are not supported.
	Supportable image size	Horizontal: Max. 1024 pixels Vertical: Max. 1024 pixels Note: The maximum display size supported by this microcontroller's LCDC differs with the display panel. Please refer to section 3.18 LCD controller.

3.19.2.5 Mechanism of BLEND Processing

The BLEND processing of the LCDDA first breaks each data of two images into the basis of pixels and then breaks them into the basis of RGB (8 bits each: 24 bits in total).

This broken down RGB data is each weighted on a scale of 256 (0x00/0x100 to 0xFF/0x100) to output the addition result data.

Source 0 pixel: $R_{S0:}$, $G_{S0:}$, $B_{S0:}$

Source 1 pixel: $R_{S1:}$, $G_{S1:}$, $B_{S1:}$

Weight assigned for Source 0

LDADRSRC0<RDRSRC0[7:0]>: $R_{S0RATIO:}$,

LDADRSRC0<GDRSRC0[7:0]>: $G_{S0RATIO:}$,

LDADRSRC0<BDRSRC0[7:0]>: $B_{S0RATIO:}$

Weight assigned for Source 1

LDADRSRC1<RDRSRC1[7:0]>: $R_{S1RATIO:}$,

LDADRSRC1<GDRSRC1[7:0]>: $G_{S1RATIO:}$,

LDADRSRC1<BDRSRC1[7:0]>: $B_{S1RATIO:}$

Calculation method:

$$R_{DST:} = (R_{S0:} \times R_{S0RATIO}) + (R_{S1:} \times R_{S1RATIO})$$

$$G_{DST:} = (G_{S0:} \times G_{S0RATIO}) + (G_{S1:} \times G_{S1RATIO})$$

$$B_{DST:} = (B_{S0:} \times B_{S0RATIO}) + (B_{S1:} \times B_{S1RATIO})$$

Thus, setting the sum of two images' weights (Case of R setting: <RDRSRC0[7:0]>+ <RDRSRC1[7:0]>) so as not to exceed 0x100 is required.

Note: If added RGB data exceeds 0x100, correct BLEND cannot be achieved.

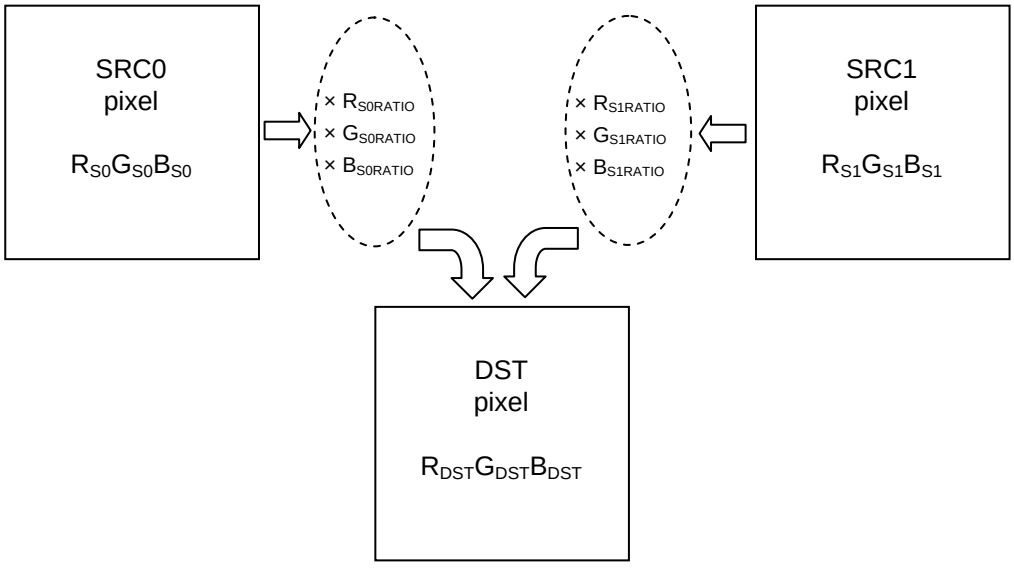
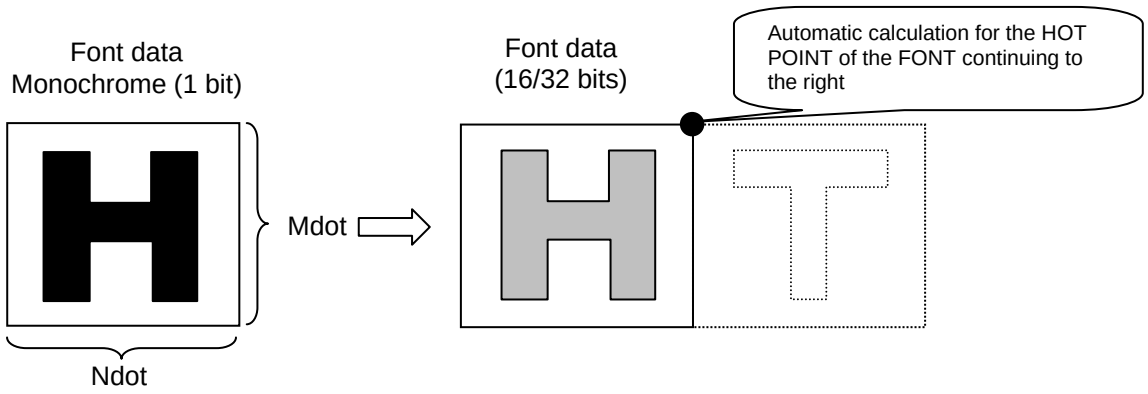


Figure 3.19.9 BLEND processing

3.19.2.6 Monochrome Blend Function

The monochrome blend function is the function of overwriting the converted data to color data that data defined in monochrome (two-valued). The BLEND of rectangular area and the overwriting font (FONT DRAW) can be possible. An excellent automatic address calculation in continuous writing of font is supported.

- Drawable "N × M" FONT defined with serialized addresses continuously in the direction of left → right (Calculating the next FONT HOT_POINT (the address in the upper left top position) automatically)



3.19.2.7 Rotation Function

Table 3.19.4 Rotation function

Function	Details of function	Description / Standard
Rotation function	Rotation angle	90° / 180° / 270° / horizontal mirror reversal / vertical mirror reversal
	Supportable data format	Digital RGB Note: YUV-format data is not supported.
	Number of supportable image colors	64-K color (16 bpp) Note: Monochrome, monochrome gray-level, and the color of other color numbers are not supported.
	Supportable image size	Horizontal: Max. 1024 pixels Vertical: Max. 1024 pixels Note: The maximum display size supported by this microcontroller's LCDC differs with the display panel. Please refer to section 3.19 LCD controller.

3.19.2.8 Rotation Processing

To perform rotation, the rotation process calculates addresses when the LCDDA reads and copies back original images. A rotation shape is controlled by either incrementing (INCREMENT) or decrementing (DECREMENT) each of the transfer-destination address's start point, the X direction, and the Y direction.

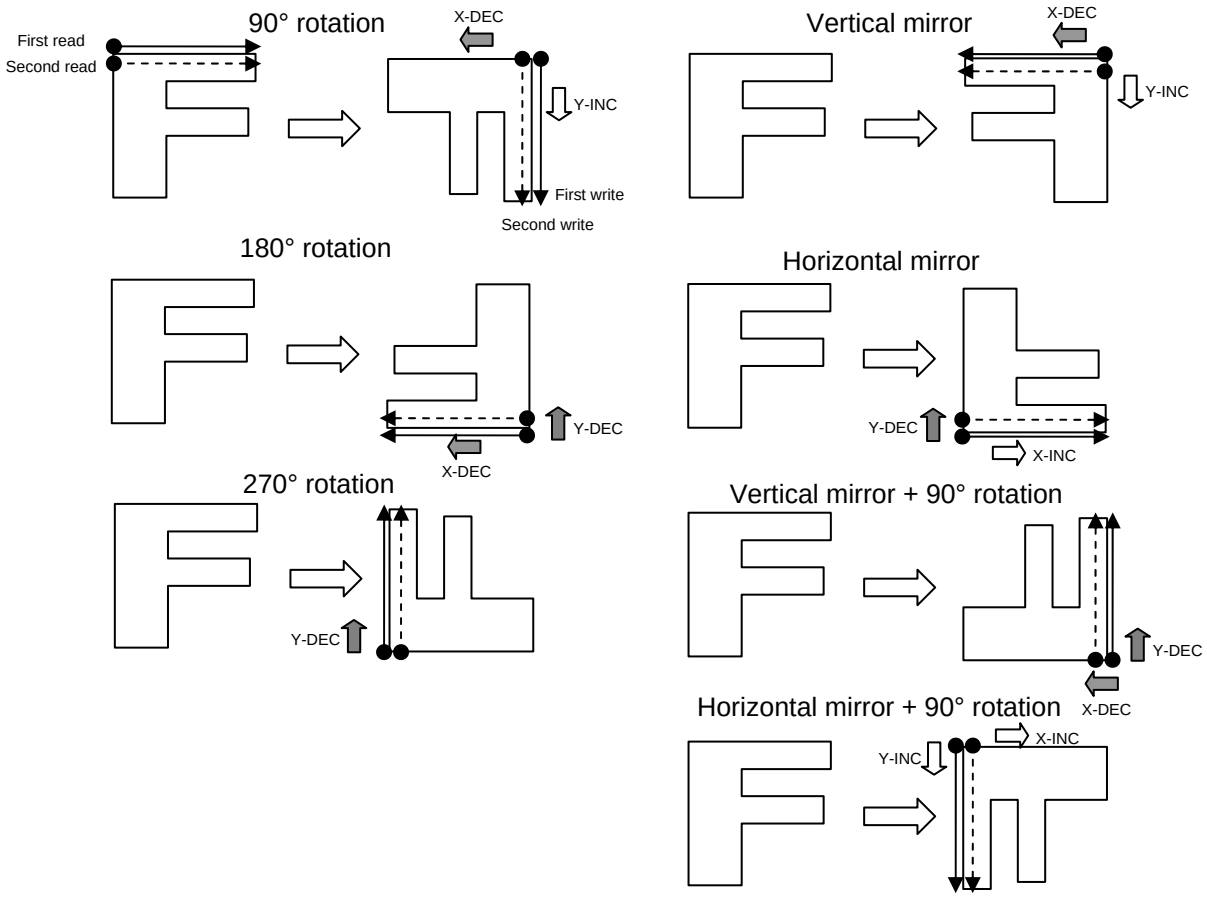


Figure 3.19.10 Rotation processing

Note: In Rotation function, all specificated rectangular area are rotated. Threfore, when using Font draw function and Rotation function together, all Fonts and backgrounds is rotated. When using this function, please be careful.

3.19.3 Operation Description of Each Mode

This section describes each operation mode of the image synthesis.

The LCDDA realizes each mode by combining appropriate original image data and circuits to be used from data sources and circuits shown in the table below.

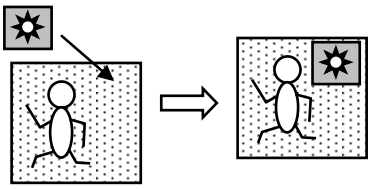
Mode	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing (Note 2)	Image data used for superimposing and simple transfer (Note 2)	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.

Note 1: The BC_Expander and Blender cannot be used simultaneously.

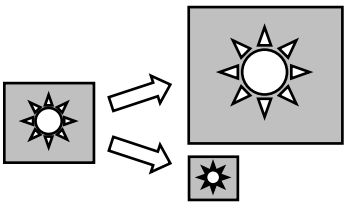
Note 2: Source0 and Source1 are not used in the scaler mode.

The following operation modes are available, which can be selected through LDACR1<OPMODE[4:0]>.

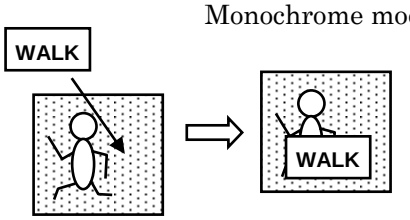
Overwriting image to image
Normal mode



Scale up or down image
Scaler mode

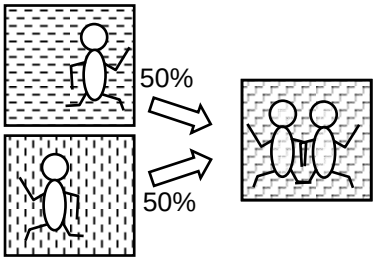


Overwriting monochrome data to image
Monochrome mode

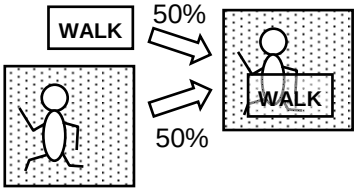


※ Monochrome reversing is supposed

Blend images
Blend mode

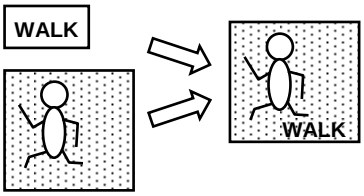


Blend monochrome data to image
Monochrome Blend mode



※ Monochrome reversing is supposed

Blend font (binary data) to image
Font draw mode



Operation Mode	Description
NORMAL mode	Performs simple data transfer of image data without color conversion or blended. And the cutout image (rectangular area) is overwritten to the background image.
Scaler mode	Controls scale up or down image data.
Monochrome mode	The monochrome cutout image (rectangular area) is overwritten to the background image after expanded color data. Transfer mode that the monochrome (1bit) data is expanded to color data of 16bit
Monochrome invert mode	After inverting the monochrome cutout image (rectangular area) data, this data is expanded to the color data and overwriting to the background image. Transfer mode that the inverted monochrome (1bit) data is expanded to color data of 16bit
BLEND mode	Blends two color images (16bit/24bit) to αBlend
Monochrome BLEND mode	αBlends a monochrome data source and color data. Specified by the monochrome image is converted to color data then the color image is αBlended to be used.
Monochrome invert BLEND mode	Inverting monochrome data, and blends two color images to αBlend. This is same as the monochrome blend mode but when converting the monochrome data to color data, the monochrome data is inverted first then being processing.
Font draw mode	Two images of monochrome source and color data are blended onto color image which is converted only the data of “1” in monochrome data to gray-level This mode is convenient for processing the superimposing and so on.

Each operation mode is described in detail on the pages that follow.

1. NORMAL mode

This is a simple data transfer mode.

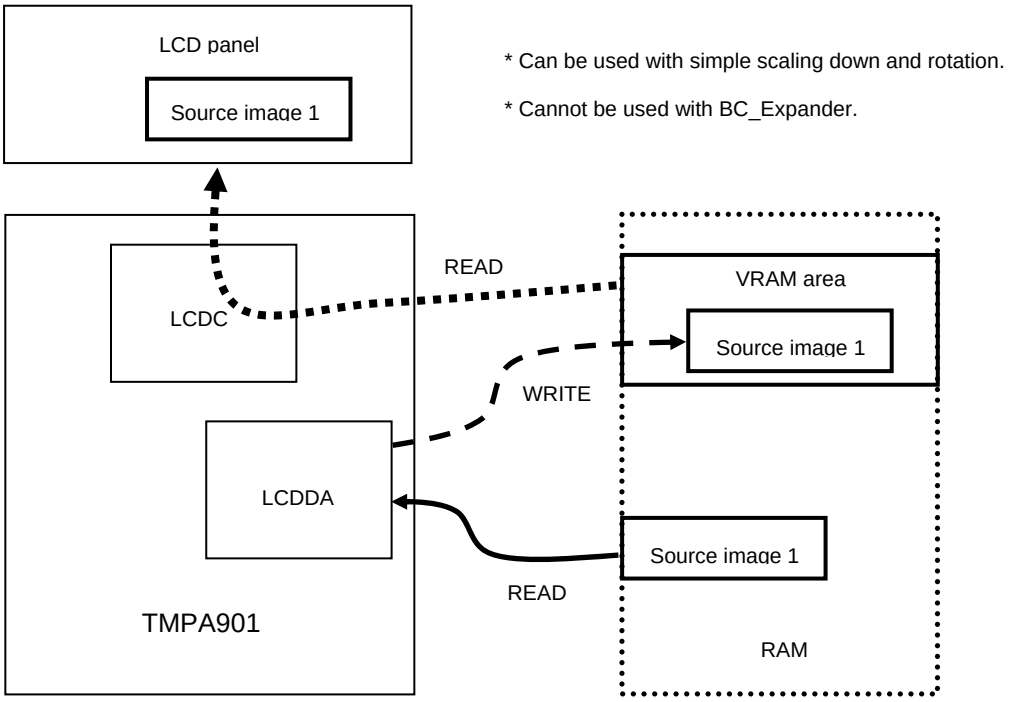
This mode is used for transfers including:

- Simple image transfer from S1 (source image 1) color original images
- Scaled-down transfer by simple thinning-out from S1 color original images
- Rotation transfer of image data from S1 color original images

NORMAL	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Not used	Used (Internal or external RAM can be used.)	Not used	Not used	Not used	Not used	Used

Application example:

The NORMAL mode can be used to display a color image in another color image (Picture In Picture). It can also be used for simple DMA transfer.



2. Scaler mode

This mode scales up or down images by controlling the operation of the BC_Expander circuit.

This mode is used such as for:

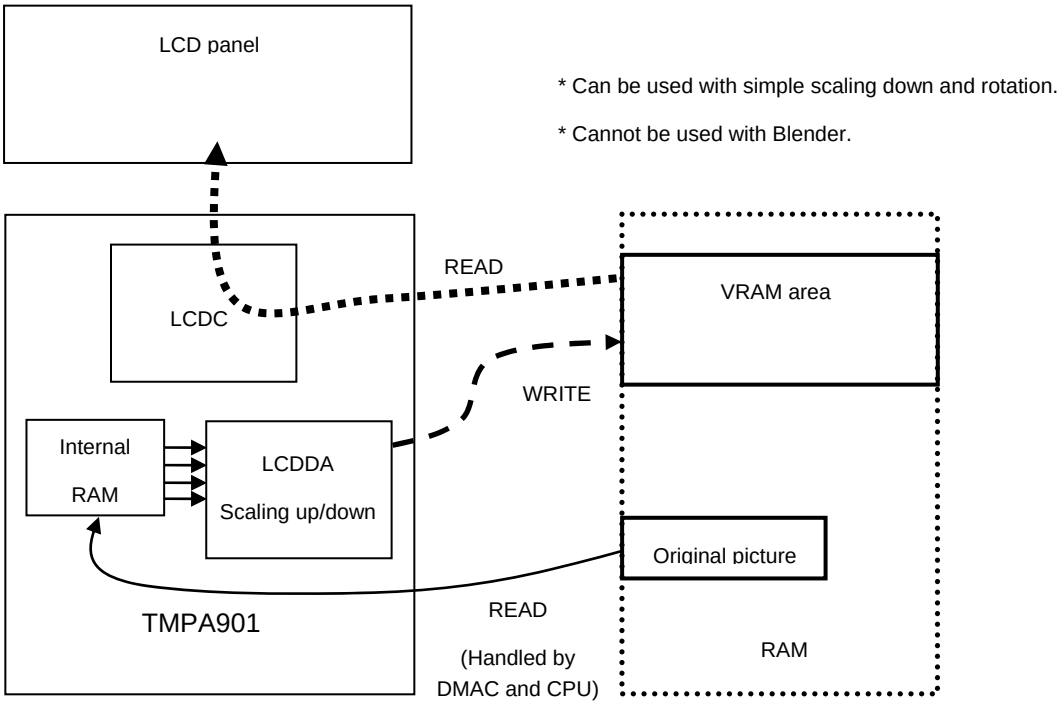
- Scaled-up image generation and transfer using Bi-Cubic from S1 color original images
- Scaled-down image generation and transfer using Bi-Cubic from S1 color original images
- Scaled-up image generation and rotation transfer using Bi-Cubic from S1 color original images
- Scaled-down image generation and rotation transfer using Bi-Cubic from S1 color original images

Scaler	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Not used	Not used	Used	Cannot be used	Cannot be used	Cannot be used	Used

Note: In the scaler mode, Source0 and Source1 are not used. This mode assumes that original image data is stored in the dual port RAM (0xF800_4000 in internal RAM-0: 16 KB).

Application example:

The scaler mode can be used to scale up small images or scale down large images. The Picture In Picture function can also be used in this mode.



3. Monochrome mode

This mode transfers monochrome source data.

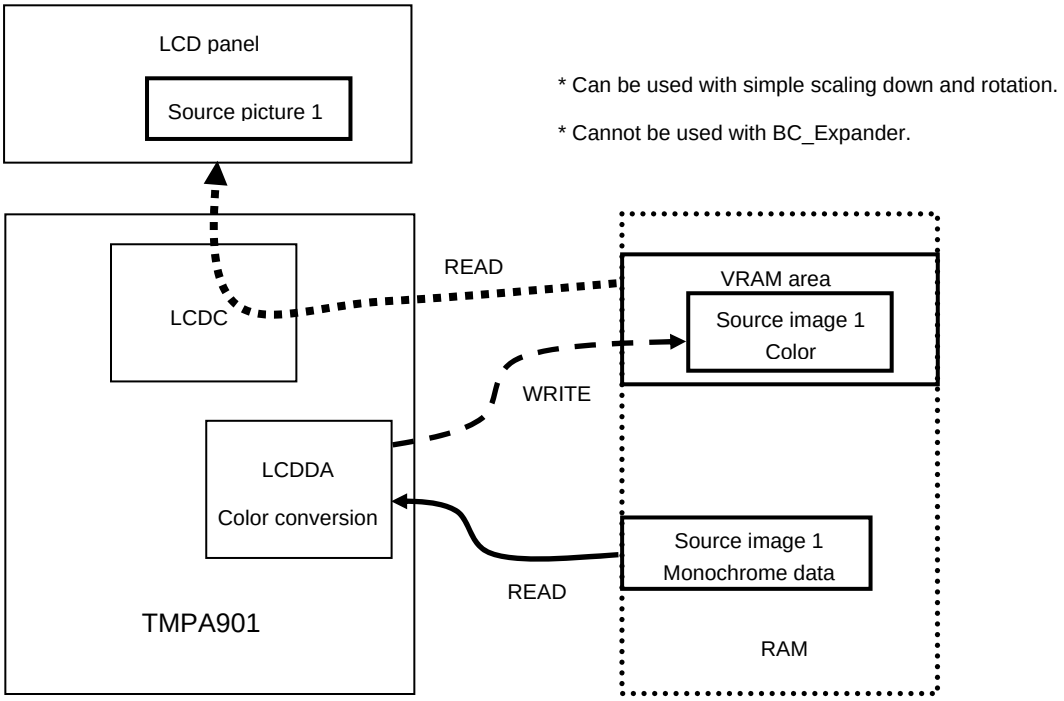
This mode is used such as for:

- Image transfer after converting S1 monochrome data into color
- Scaled-down transfer by simple thinning-out after converting S1 monochrome data into color
- Rotation transfer of image data after converting S1 monochrome data into color

Monochrome	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Not used	Used (Internal or external RAM can be used.)	Cannot be used	Used	Not used	Not used	Used

Application example:

The monochrome mode can be used to overwrite monochrome data onto a color image (Picture In Picture).



4. Monochrome invert mode

This mode inverts and transfers monochrome source data. It is the same as the monochrome mode except that monochrome data is inverted before being converted into color and transferred.

This mode is used such as for:

・Image transfer after converting S1 monochrome data into color

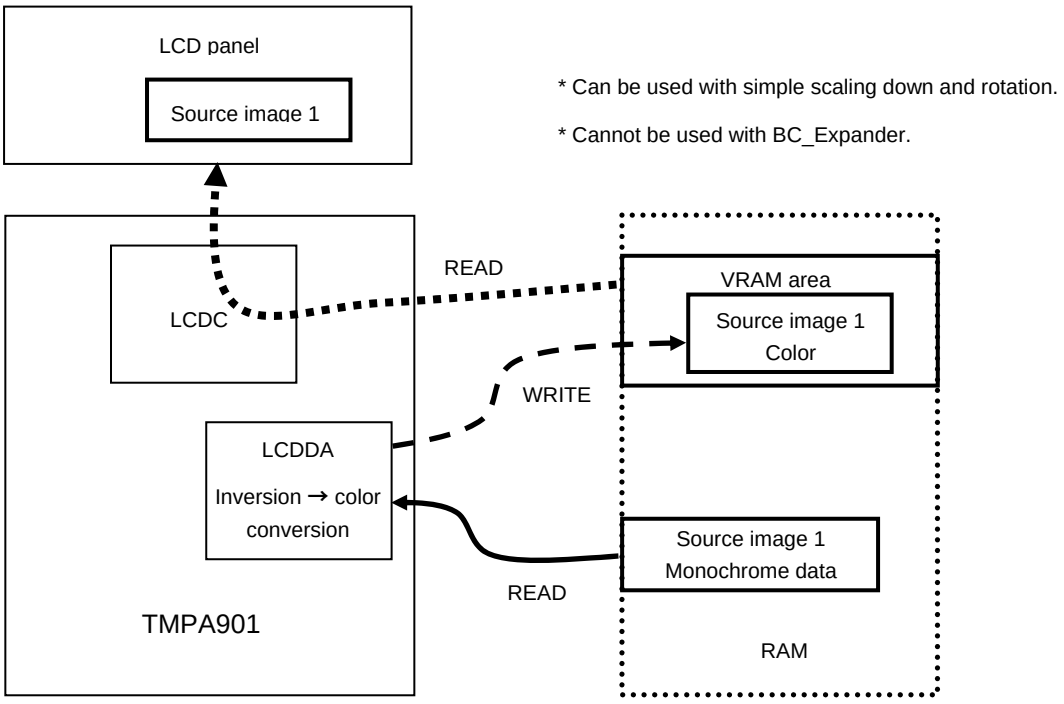
・Scaled-down transfer by simple thinning-out after converting S1 monochrome data into color

・Rotation transfer of image data after converting S1 monochrome data into color

Monochrome Invert	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Not used	Used (Internal or external RAM can be used.)	Cannot be used	Used	Not used	Not used	Used

Application example:

The monochrome invert mode can be used to overwrite inverted two-valued data onto a color image (Picture In Picture).



5. BLEND mode

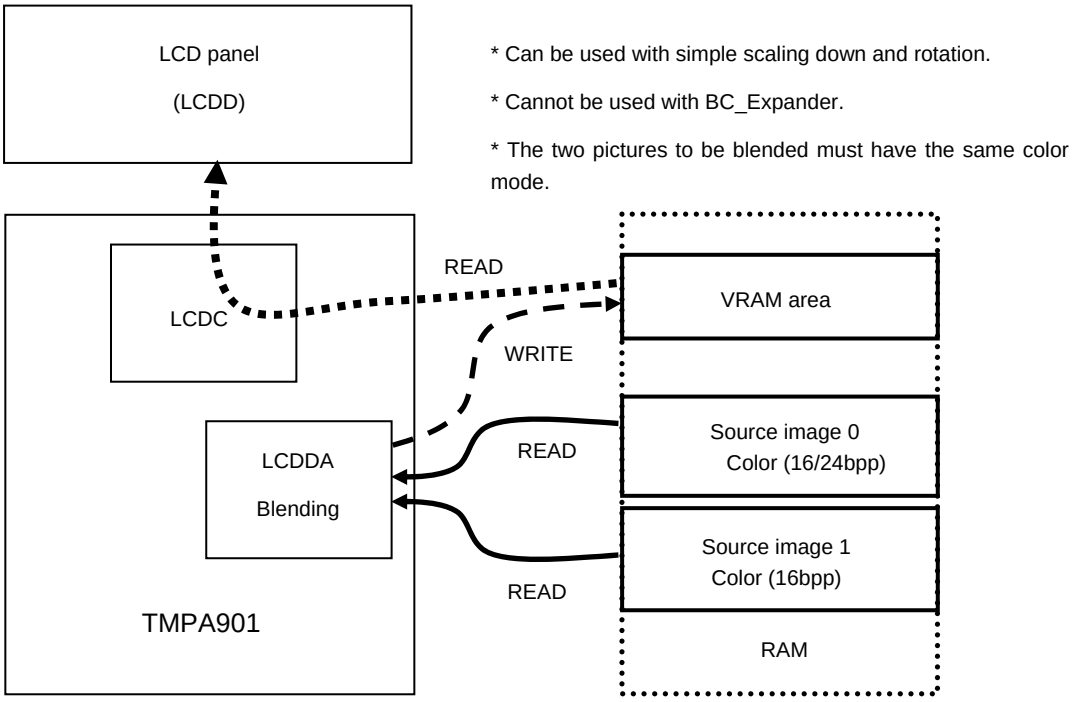
This mode blends two color (16 bpp) images.

This mode is used such as for:
- Image transfer after blending two color data of S0 and S1
- Scaled-down transfer by simple thinning-out after blending two color data of S0 and S1
- Rotation transfer of image data after blending two color data of S0 and S1

BLEND	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Used (Internal or external RAM can be used.)	Used (Internal or external RAM can be used.)	Cannot be used	Not used	Not used	Used	Used

Application example:

The BLEND mode enables gradual switching between two color images by gradually changing the blend ratio.



6. Monochrome BLEND mode

This mode blends two images of monochrome source and color (16 bpp).

This mode is used such as for:

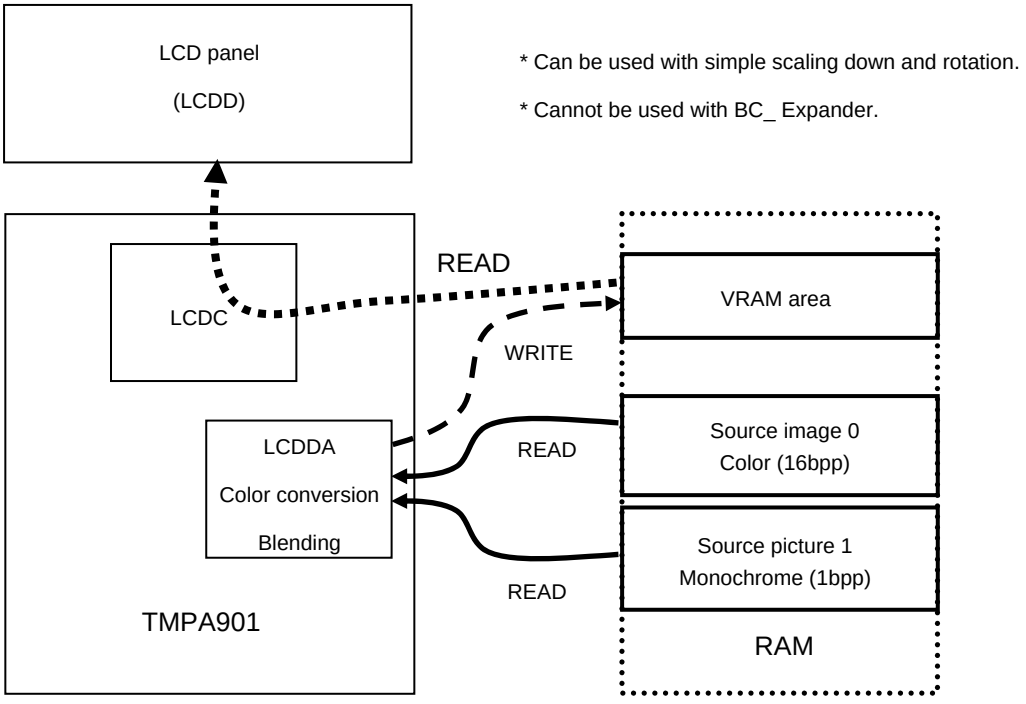
- Image transfer after blending S0 color data and color-converted S1 monochrome data
- Scaled-down transfer by simple thinning-out after blending S0 color data and color-converted S1 monochrome data
- Rotation transfer after blending S0 color data and color-converted SI monochrome data

Monochrome BLEND	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not Used	Used (Internal or external RAM can be used.)	Used (Internal or external RAM can be used.)	Cannot be used	Used	Not used	Used	Used

Note: Monochrome data from Source1 must be used.

Application example:

The monochrome BLEND mode enables gradual switching between color and monochrome pictures by gradually changing the blend ratio. When converting two-valued data, this mode can be possible to support the superimposing of FONT data but white color only, to set the color of part of data "1" to R:0xFF, G:0xFF and B:0xFF (white color) then 100% blend ratio and 0% blend ratio of converting "0" data. (When background is white color, this method is not corresponded.)



7. Monochrome invert BLEND mode

This mode blends two images of monochrome source inverted data and color (16bpp). This mode is the same as the monochrome BLEND mode except that monochrome data is inverted when converted into color.

This mode is used such as for:

・Image transfer after blending S0 color data and color-converted S1 monochrome data

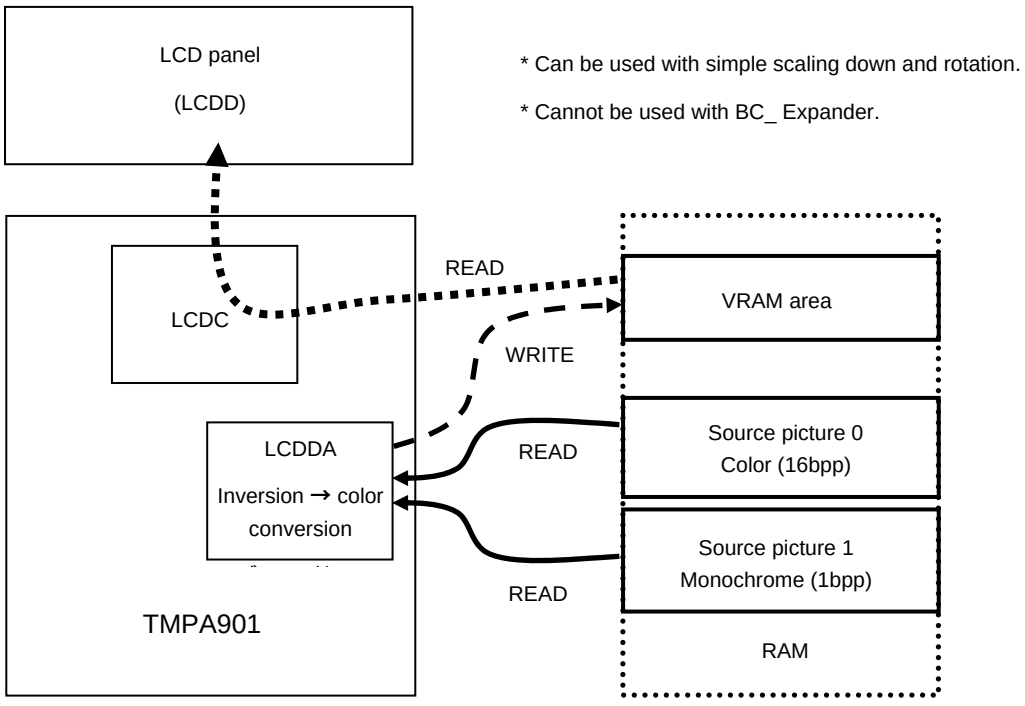
・Scaled-down transfer by simple thinning-out after blending S0 color data and color-converted S1 monochrome data

・Rotation transfer after blending S0 color data and color-converted S1 monochrome data

Monochrome Invert BLEND	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Used (Internal or external RAM can be used.)	Used (Internal or external RAM can be used.)	Cannot be used	Used	Not used	Used	Used

Application example:

The monochrome invert BLEND mode enables gradual switching between color and monochrome images by gradually changing the blend ratio.



8. Font Draw mode

This mode blends two images of monochrome source data and color (16bpp/24bpp). This mode blends that bit is expanded only data "1" which is specified by monochrome image data (S1).

This mode is used such as for:

・Image transfer after blending S0 color data and expanded only data "1" is monochrome data of S1.

・Scaled-down transfer by simple thinning-out after blending S0 color data and expanded only data "1" is monochrome data of S1.

・Rotation transfer after blending S0 color data and expanded only data "1" is monochrome data of S1.

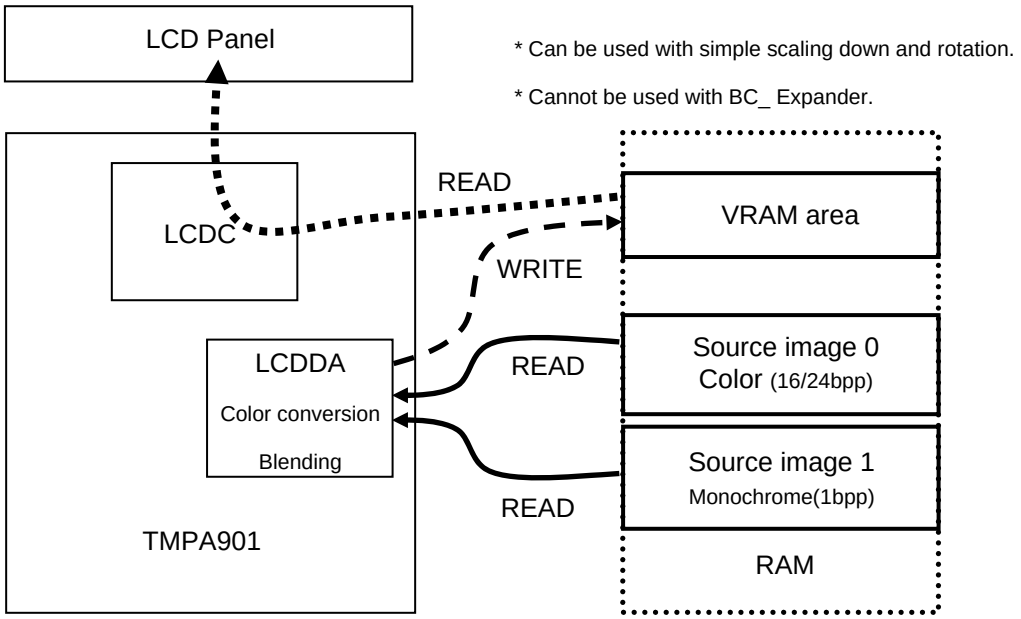
Note: The expanded bit of color font is used to set the LDAFCPSRC1<RFONT[7:0]>, <GFONT[7:0]> and <BFONT[7:0]>, in this case, the setting must be set to <RFONT[7:0]>=<BFONT[7:0]>.

Font Draw	Original Image Data		Circuits Used				
	Source0	Source1	BC Expander	Blender			Transfer Address Control Circuit
				Color Converter	Area Filter	Superimposer	
Description	Image data only used for superimposing	Image data used for superimposing and simple transfer	Scales up or down display data.	Performs color conversion.	Filters areas to be superimposed.	Performs superimposing.	Performs rotation and simple scaling down.
Used/Not used	Used (Internal or external RAM can be used.)	Used (Internal or external RAM can be used.)	Cannot be used	Used	Not used	Used	Used

Application example:

The Font Draw mode can be used to overwrite only Font data onto a color image.

Note: As this transfer is executed after blending, so please pay attention that the background Font is processed rotation but also the background image is rotated.



3.19.4 Description of Registers

The following lists the SFRs:

Base address = 0xF205_0000

Register Name	Address (base+)	Description
LDACR0	0x0000	LCDDA Control Register 0
LDADRSRC1	0x0004	LCDDA Density Ratio of Source 1 Image
LDADRSRC0	0x0008	LCDDA Density Ratio of Source 0 Image
LDAFCPSRC1	0x000C	LCDDA Replaced Font Area Color pallet of Source1
LDAEFCPSRC1	0x0010	LCDDA Replaced Except Font Area Color pallet of Source1
LDADVSRC1	0x0014	LCDDA Delta Value (Read Step) address Register of Source 1
LDACR2	0x0018	LCDDA Control Register 2
LDADXDST	0x001C	LCDDA X-Delta Value (Write Step) address Register of Destination
LDADYDST	0x0020	LCDDA Y-Delta Value (Write Step) address Register of Destination
LDASSIZE	0x0024	LCDDA Source Image Size
LDADSIZE	0x0028	LCDDA Destination Image Size
LDAS0AD	0x002C	LCDDA Source 0 Start Address
LDADAD	0x0030	LCDDA Destination Start Address
LDACR1	0x0034	LCDDA Control Register1
LDADVSRC0	0x0038	LCDDA Delta Value (Read Step) address Register of Source 0

The LCDDA has 14 types of registers. They are connected to the CPU with the 32-bit bus.

1. LDACR0 (LCDDA Control Register 0)

Address = (0xF205_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:22]	–	–	Undefined	Read as undefined. Write as zero.
[21]	ERRINTF	RW	0y0	LCDDA processing error flag During READ During Write 0y0: No interrupt 0y0: Flag clear 0y1: With interrupt 0y1: Invalid
[20]	EINTF	RW	0y0	Scaler 1-line processing end interrupt enable (Enabled by Scaler/Rotation function) During READ During Write 0y0: No interrupt 0y0: Flag clear 0y1: With interrupt 0y1: Invalid
[19:18]	–	–	Undefined	Read as undefined. Write as zero.
[17]	ERRINTM	RW	0y0	LCDDA processing error interrupt MASK 0y0: Interrupt mask 0y1: Interrupt enabled
[16]	EINTM	RW	0y0	LCDDA 1-image processing end interrupt MASK (Enabled by scaler/Rotation function) 0y0: Interrupt mask 0y1: Interrupt enabled
[15]	BCENYB	RW	0y0	Y-direction last LINE data correction (Last dummy line addition) 0y0: OFF 0y1: ON
[14]	AUTOHP	RW	0y0	Automatic calculation of HOT point 0y0: OFF 0y1: ON
[13]	DMAMD	RW	0y0	DMA select 0y0: Single transfer 0y1: Burst transfer
[12]	DMAEN	RW	0y0	DMA enable. 0y0: OFF 0y1: Enable
[11]	BCENYT	RW	0y0	Y-direction front LINE data correction (Front dummy line addition) 0y0: OFF 0y1: ON
[10]	DTFMT	RW	0y0	Display color select 0y0: Reserved 0y1: 64-K color (16 bits)
[9]	BCENX	RW	0y0	X-direction edge data correction (Right-left dummy row addition) 0y0: OFF 0y1: ON
[8]	PCEN	RW	0y1	Period correction 0y0: OFF 0y1: ON
[7:0]	S1ADR[31:24]	RW	0x00	SRC1 image's front address (Higher 8 bits of 32 bits)

[Description]

a. <ERRINTF >

Shows the status of an interrupt that shows the occurrence of processing errors in the LCDDA circuit.

Note that the meanings differ between during READ and during WRITE.

During READ	During WRITE
0y0: No interrupt	0y0: Flag clear
0y1: With interrupt	0y1: Invalid (No status change)

b. <EINTF>

Shows the status of an interrupt that shows the end of processing for one line in the LCDDA's scaler/filter circuit.

This interrupt shows the end of internal processing of the LCDDA's scaler, BLEND and Rotation functions. This, however, requires a caution because this interrupt does not show the end of transfer of the data accumulated in the write buffer.

Scaler is processed every original image's 1-line. Therefore, interrupt is generated more than once until the scale-up/scale-down of the one image is terminated.

Note that the meanings differ between during READ and during WRITE.

During READ	During WRITE
0y0: No interrupt	0y0: Flag clear
0y1: With interrupt	0y1: Invalid (No status change)

c. <ERRINTM>

Sets the mask for a processing error occurrence interrupt in the LCDDA circuit.

0y0: Error interrupt mask

0y1: Error interrupt enabled

d. <EINTM>

Sets the mask for an interrupt that shows the end of processing for one line in the LCDDA's scaler circuit.

0y0: Mask for one-line processing interrupt

0y1: One-line processing interrupt enabled

e. <BCENYB>

Controls the function of automatically adding dummy sampling points in Y-direction last LINEs for using the interpolation scaler function.

0y0: OFF

0y1: ON

f. <AUTOHP>

Automatic calculation of HOT point

0y0: OFF

0y1: ON

g. <DMAMD>

Sets the DMA transfer mode of the LCDDA.

0y0: Single transfer

0y1: Burst transfer

h. <DMAEN>

This is the signal that shows the end of processing for one image in the LCDDA and controls the enable of DMA transfer.

0y0: DMA disabled

0y1: DMA enabled

i. <BCENYT>

Controls the function of automatically adding dummy sampling points in Y-direction front LINEs for using the interpolation scaler function.

0y0: OFF

0y1: ON

j. <DTFMT>

Defines the format of RGB data handled by the LCDDA.

0y0: Reserved

0y1: 64-K color (16-bit data)

k. <BCENX>

Controls the function of automatically adding dummy sampling points in X-direction left/right rows for using the interpolation scaler function.

0y0: OFF

0y1: ON

l. <PCEN>

Controls the function of correcting periodical sampling points for using the scaler function.

0y0: OFF

0y1: ON

m. <S1ADR [31:24]>

Shows the front address of the memory in which original images (Source image 1) processed on the LCDDA are stored. The higher 8 bits of the 32-bit address area are set.

2. LDADRSRC1 (LCDDA Density Ratio of Source 1 Image)

Address = (0xF205_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	–	–	Undefined	Read as undefined. Write as zero.
[23:16]	BDRSRC1[7:0]	R/W	0x00	Blue data gray level adjustment in SRC1 image (256 levels) 0x00: Light to 0xFF: Dark
[15:8]	GDRSRC1[7:0]	R/W	0x00	Green data gray level adjustment in SRC1 image (256 levels) 0x00: Light to 0xFF: Dark
[7:0]	RDRSRC1[7:0]	R/W	0x00	Red data gray level adjustment in SRC1 image (256 levels) 0x00: Light to 0xFF: Dark

Adjust the gray level of Source 1 (foreground) image independently for R, G, B

3. LDADRSRC0 (LCDDA Density Ratio of Source 0 Image)

Address = (0xF205_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	–	–	Undefined	Read as undefined. Write as zero.
[23:16]	BDRSRC0[7:0]	R/W	0x00	Blue data gray level adjustment in SRC0 image (256 levels) 0x00: Light to 0xFF: Dark
[15:8]	GDRSRC0[7:0]	R/W	0x00	Green data gray level adjustment in SRC0 image (256 levels) 0x00: Light to 0xFF: Dark
[7:0]	RDRSRC0[7:0]	R/W	0x00	Red data gray level adjustment in SRC0 image (256 levels) 0x00: Light to 0xFF: Dark

Adjust the gray level of Source 0 (background) image independently for R, G, B.

4. LDAFCPSRC1 (LCDDA Replaced Font Area Color pallet of Source1)

Address= (0xF205_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	–	–	Undefined	Read as undefined. Write as zero.
[23:16]	BFONT[7:0]	R/W	0x00	FONT area color in SRC1 image (Blue data)
[15:8]	GFONT[7:0]	R/W	0x00	FONT area color in SRC1 image (Green data)
[7:0]	RFONT[7:0]	R/W	0x00	FONT area color in SRC1 image (Red data)

Set the FONT color of Source 1 (foreground) image independently for R, G, B.

During in font draw mode, set to <RFONT[7:0]>=<BFONT[7:0]>.

5. LDAEFCPSRC1 (LCDDA Replaced Except Font Area Color pallet of Source1)

Address= (0xF205_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	–	–	Undefined	Read as undefined. Write as zero.
[23:16]	BFONT[7:0]	R/W	0x00	Area color other than FONT in SRC1 image (Blue data)
[15:8]	GFONT[7:0]	R/W	0x00	Area color other than FONT in SRC1 image (Green data)
[7:0]	RFONT[7:0]	R/W	0x00	Area color other than FONT in SRC1 image (Red data)

Set the color of places other than FONT in Source 1 (foreground) independently for R, G, B.

6. LDADVSRC0 (LCDDA Delta Value (Read Step) address Register of Source 0)

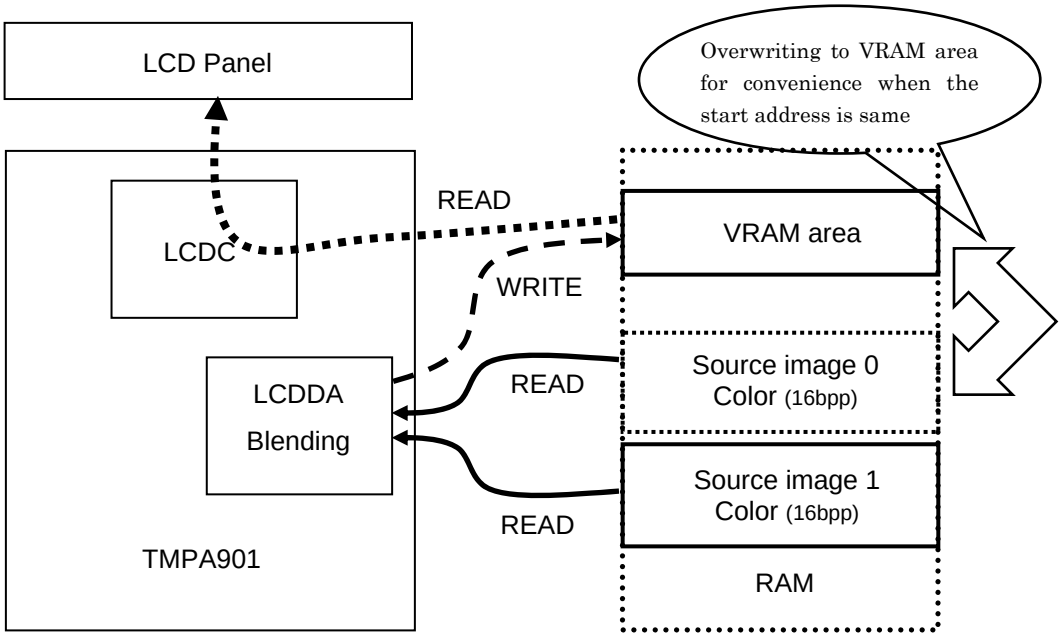
Address = (0xF205_0000) + (0x0038)

Bit	Bit Symbol	Type	Reset Value	Description
[31]	OVWEN	RW	0y0	0y0: SRC0 start address ≠ DST start address 0y1: SRC0 start address = DST start address
[30]	INDSAEN	RW	0y0	0y0: SRC0 DX, DY = SRC1 DX, DY 0y1: SRC0 DX, DY ≠ SRC1 DX, DY
[29:19]	–	–	Undefined	Read as undefined. Write as zero.
[18:6]	DYS0[12:0]	R/W	0x000	Read Step address until the next line of SRC0 data
[5:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	DXS0[2:0]	R/W	0y000	Horizontal Read Step address of SRC0 data

[Description]

a. <OVWEN>

This bit is used when Source 0 image and the destination image are the same in an image processed on the LCDDA. Setting 1 uses the front address setting of the destination image into the front address of Source 0 image too. In circuitry, the blend is executed and used to change the display which is blended part of current displaying image.



b. <INDSAEN>

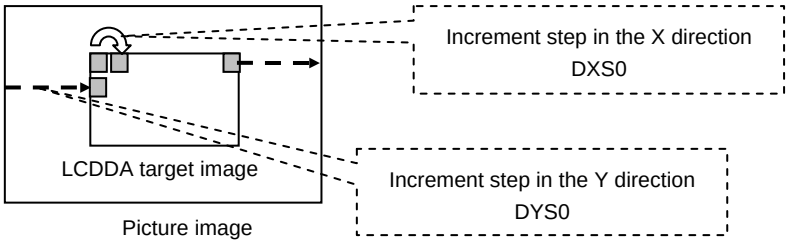
This bit is used when increment steps differ between Source 0 image and Source 1 image in an image processed for BLEND on the LCDDA. Setting 1 can set the number of steps individually for each Source 0 image and Source 1 image.

When 0 is set, the increment step set for Source 1 image is used for the increment step in Source 0 image.

c. <DYS0>

Used to have increment step settings in Source 0 image differing from Source 1 image. Not setting 1 in the INDSAEN bit disables this setting, applying the increment step settings of Source 1 image to Source 0 image too.

Set the Step for vertical increment addresses (during line feed) for reading data from the original image (Source image 0) processed on the LCDDA.



Set address step values after calculating them for each display color used.

d. <DXS0[2:0]>

Used to have increment step settings in Source 0 image differing from Source 1 image. Not setting 1 in the INDSAEN bit disables this setting, applying the increment step settings of Source 1 image to Source 0 image too.

Set the Step for horizontal increment addresses for reading data from the original image (Source image 0) processed on the LCDDA. For 16-bpp (64-K color) data, set 0y010 because the step used is 2-byte step.

7. LDADVSRC1 (LCDDA Delta Value (Read Step) address Register of Source 1)

Address = (0xF205_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	OFSETX[7:0]	R/W	0x0000	Offset value for horizontal sampling point during scaler use
[23:18]	–	–	Undefined	Read as undefined. Write as zero.
[17:6]	DYS1[11:0]	R/W	0x000	Read Step address until the next line of SRC1 data
[5:3]	–	–	Undefined	Read as undefined. Write as zero.
[2:0]	DXS1[2:0]	R/W	0y000	Horizontal Read Step address of SRC1 data

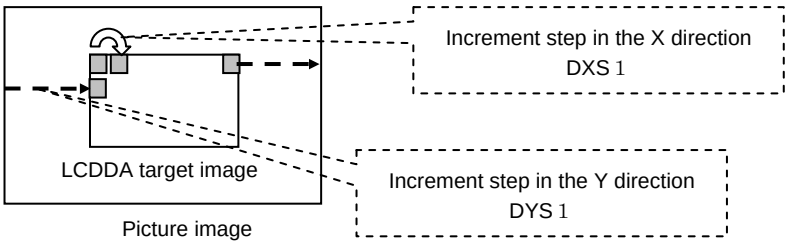
[Description]

a. <OFSETX[7:0]>

Set the offset value for the horizontal pixel sampling steps in the scaler circuit. By setting a value of 0x01 to 0xFF, the set pixel point is sampled first.

b. <DYS1[11:0]>

Set the Step for vertical increment addresses (during line feed) for reading data from the original image (Source image 1) processed on the LCDDA.



c. <DXS1[2:0]>

Set the Step for horizontal increment addresses for reading data from the original image (Source image 1) processed on the LCDDA. For 16-bpp (64-K color) data, set 0y010 because the step used is 2-byte step.

8. LDACR2 (LCDDA Control Register 2)

Address = (0xF205_0000) + (0x0018)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	OFSETY[7:0]	R/W	0x0000	Offset value for vertical sampling point during scaler use
[23:16]	–	–	Undefined	Read as undefined. Write as zero.
[15:8]	HCRCT[7:0]	R/W	0x00	Horizontal period correction value
[7:0]	VCRCT[7:0]	R/W	0x00	Vertical period correction value

[Description]

a. <OFSETY[7:0]>

Sets the offset value for the vertical pixel sampling steps in the scaler circuit. By setting a value of 0x01 to 0xFF, the set pixel point is sampled first.

b. <HCRCT[7:0]>

Sets the horizontal period correction values in the scaler circuit.
Setting the LDACR0<PCEN> to 0y1 enables this bit. By setting a value of 0x01 to 0xFF, the set pixel point is corrected to the point one point right-side to the original pixel.

c. <VCRCT[7:0]>

Sets the vertical period correction values in the scaler circuit.
Setting the LDACR0<PCEN> to 0y1 enables this bit. By setting a value of 0x01 to 0xFF, the set pixel point is corrected to the point one line below the original pixel.

9. LDADXDST (LCDDA X-Delta Value (Write Step) address Register of Destination)

Address = (0xF205_0000) + (0x001C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:28]	XRDRATE[3:0]	R/W	0y0000	Horizontal scale-down rate
[27:25]	–	–	Undefined	Read as undefined. Write as zero.
[24]	DXDSIGN	R/W	0y0	Horizontal heading direction in DST data
[23:0]	DXDST[23:0]	R/W	0x000000	Number of horizontal steps in DST data

[Description]

a. <XRDRATE[3:0]>

Sets horizontal scale-down values.

By setting a value of 0x0 to 0xf, perform a scale down having a value of "set value + 1" as the denominator.

Example: When set to 0x2:

Results as 1/ (2 + 1) = 1/3, scaling down to 1/3 in the horizontal direction.

b. <DXDSIGN>

Sets the direction of horizontal destination step.

Set this by deciding the address's heading direction according to the Rotation function's rotation angle and horizontal/vertical mirror.

0y0: Plus (Increment)

0y1: Minus (Decrement)

c. <DXDST[23:0]>

Sets horizontal destination step addresses.

In order to control addresses and accomplish the Rotation function, 24 bits are provided for step addresses. Set step addresses according to the Rotation function's rotation angle and horizontal/vertical mirror.

10. LDADYDST (LCDDA Y-Delta Value (Write Step) address Register of Destination)

Address = (0xF205_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:28]	YRDRATE[3:0]	R/W	0y0000	Vertical scale-down rate
[27:25]	–	–	Undefined	Read as undefined. Write as zero.
[24]	DYDSIGN	R/W	0y0	Vertical write direction in DST data
[23:0]	DYDST[23:0]	R/W	0x000000	Number of vertical steps in DST data

[Description]

a. <YRDRATE[3:0]>

Sets vertical scale-down rate.
By setting a value of 0x0 to 0xf, perform a scale down having a value of "set value + 1" as the denominator.

Example: When set to 0xf,
Results as 1/(15 + 1) = 1/16, scaling down to 1/16 in the vertical direction.

b. <DYDSIGN>

Sets the direction of vertical destination step.
Set this by deciding the address's heading direction according to the Rotation function's rotation angle and horizontal/vertical mirror.
0y0: Plus (Increment)
0y1: Minus (Decrement)

c. <DYDST[23:0]>

Sets vertical destination step addresses.
In order to control addresses and accomplish the Rotation function, 24 bits are provided for step addresses. Set step addresses according to the Rotation function's rotation angle and horizontal/vertical mirror.

11. LDASSIZE (LCDDA Source Image Size)

Address = (0xF205_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	XEXRATE[7:0]	R/W	0x00	Horizontal scale-up rate during scaler use
[23:22]	–	–	Undefined	Read as undefined. Write as zero.
[21:12]	SYSIZE[9:0]	R/W	0x000	Vertical SRC image size (Dot-basis setting)
[11:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:0]	SXSIZE[9:0]	R/W	0x000	Horizontal SRC image size (Dot-basis setting)

Note1: Setting required in all functions

Note2: The horizontal image max size is 511 dot when used in the scaler function.

[Description]

a. <XEXRATE[7:0]>

Sets the horizontal scale-up rate.
By setting a value of 0x01 to 0xff (Setting to 0x00 disabled), perform a scale up.

Input values according to the equation below:

- Number of original pixels in the X direction: m pixel(s)
- Number of after-scaled-up pixels in the X direction: n pixel(s)

(Maximum number of interpolatable pixels) / (Number of after-scaled-up pixels)
$= \{(m - 1) \times 256\} \div n$

b. <SYSIZE[9:0]>

Sets vertical source image sizes.
Sets an image size on a dot basis. A dot of "input size + 1" is specified for size.
Example: For 200 dots, set as 199 (oct) = C7 (hex).

c. <SXSIZE[9:0]>

Sets horizontal source image sizes.
Sets an image size on a dot basis. A dot of "input size + 1" is specified for size.
Example: For 200 dots, set as 199 (oct) = C7 (hex).

12. LDADSIZE (LCDDA Destination Image Size)

Address = (0xF205_0000) + (0x0028)

Bit	Bit Symbol	Type	Reset Value	Description
[31:24]	YEXRATE[7:0]	R/W	0x00	Vertical scale-up rate during scaler use
[23:10]	–	–	Undefined	Read as undefined. Write as zero.
[9:0]	DXSIZE[9:0]	R/W	0x000	Horizontal DST image size (Dot-basis setting)

Note: When used in the scaler function, vertical image size settings are invalid. Because the scaler function is processed on the basis of one line, the number of processes will result as the image size in the vertical direction directly.

[Description]

a. <YEXRATE[7:0]>

Sets the vertical scale-up rate.
By setting a value of 0x01 to 0xff (Setting to 0x00 disabled), perform a scale up.

Input values according to the equation below:

- Number of original pixels in the Y direction: k pixel(s)
- Number of after-scaled-up pixels in the Y direction: l pixel(s)

(Maximum number of interpolatable pixels) / (Number of after-scaled-up pixels)
$= \{(k - 1) \times 256\} \div l$

b. <DXSIZE[9:0]>

Sets horizontal destination image sizes.
Sets an image size on a dot basis. A dot of "input size + 1" is specified for size.

Example: For 200 dots, set as 199(oct) = C7(hex).

13. LDAS0AD (LCDDA Source 0 Start Address)

Address = (0xF205_0000) + (0x002C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	S0ADR[31:0]	R/W	0x00000000	Start address of SRC0 image

Note: While 32-bit addresses are set, the addresses in the higher 8 bits have no internal counter. Note that the setting of start address allowing the addresses in the higher 8 bits to change in the SRC image data area is unavailable.
(Example: If set to 0x00FFFFFF, the LCDDA accesses in the order of 0x00FFFFFF → 0x00000000 → 0x00000001: The addresses in the higher 8 bits cannot be incremented.)

14. LDADAD (LCDDA Destination Start Address)

Address = (0xF205_0000) + (0x0030)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	DSADR[31:0]	R/W	0x00000000	Start address of DST image

Note: While 32-bit addresses are set, the addresses in the higher 8 bits have no internal counter. Note that the setting of start address allowing the addresses in the higher 8 bits to change in the DST image data area is unavailable.
(Example: If set to 0x00FFFFFF, the LCDDA accesses in the order of 0x00FFFFFF → 0x00000000 → 0x00000001: The addresses in the higher 8 bits cannot be incremented.)

15. LDACR1 (LCDDA Control Register1)

Address = (0xF205_0000) + (0x0034)

Bit	Bit Symbol	Type	Reset Value	Description
[31]	SYNRST	WO	0y0	S/W reset control
[30]	LDASTART	WO	0y0	LCDDA START control
[29]	–	–	Undefined	Read as undefined. Write as zero.
[28:24]	OPMODE[4:0]	R/W	0y00000	LCDDA mode setting 0y00000 : NORMAL mode 0y00001 : Scaler mode 0y00010 : Monochrome mode 0y00110 : Monochrome invert mode 0y10000 : BLEND mode 0y10010 : Monochrome BLEND mode 0y10110 : Monochrome invert BLEND mode 0y11010 : Font Draw mode * Other combination of bits (functions) than the above cannot be set.
[23:0]	S1ADR[23:0]	R/W	0x000000	SRC1 image's front address (Lower 24 bits of 32 bits)

[Description]

a. <SYNRST>

Controls the S/W reset.
0y1: Reset
0y0: Ignored

b. <LDASTART>

Controls the start of the LCDDA.
0y1: LCDDA start
0y0: Ignored

c. <OPMODE[4:0]>

Selects the LCDDA operation modes.

0y00000: NORMAL mode
This is a simple data transfer mode. Used to transfer (rotation, scaling down) rectangular images with only Source 1 (S1) selected for source data (S0 settings disabled).

0y00001: Scaler mode
The scaler mode is accomplished by controlling the operation of the BC-Expander circuit.
The concurrent use with the BLEND function and the FONT function is unavailable.

0y00010: Monochrome mode
This mode transfers monochrome sources. Used to transfer (rotation, scaling down) images after converting monochrome data into color data with only Source 1 (S1) selected for source data (S0 settings disabled).

0y00110: Monochrome invert mode

This mode inverts and transfers monochrome source data. Used to transfer (rotation, scaling down) images after inverting monochrome data to convert it into color data with only Source 1 (S1) selected for source data (S0 settings disabled).

0y10000: BLEND mode

This mode blends two color (16bpp) images. Used to transfer (rotation, scaling down) images after blending them, with Source 0 (S0) and Source 1 (S1) selected for source data.

0y10010: Monochrome BLEND mode

This mode blends two images of monochrome source and color (16bpp). Used to transfer (rotation, scaling down) images after blending two images of color data converted from a Source 1 (S1) image specified in monochrome, and Source 0 (S0) specified in color.

0y10110: Monochrome invert BLEND mode

This mode blends two images of data inverted from monochrome-source data and color (16bpp). Used to transfer (rotation, scaling down) images after blending two images of color data converted from a Source 1 (S1) image specified in monochrome, and Source 0 (S0) specified in color.

0y11010: Font Draw mode

This mode blends two images of monochrome source data and color data (16bpp). And blends two images which is converting the specified in monochrome source 1 (S1) image to color data and is specified by color source 0 (S0). Only “1” of monochrome data is valid and transfers (rotation/scaled-down) the expanded image.

d. <S1ADR[23:0]>

Sets the lower 24 bits for the start addresses of Source 1 picture.

While 32-bit setting is required for the addresses of Source 1 picture, set the addresses in the higher 8 bits by LDACR0<S1ADR[31:24]>.

3.20 Touch Screen Interface (TSI)

An interface for 4-terminal resistor network touch-screen is built in. The TSI easily supports two procedures: touch detection and X/Y position measurement. Each procedure is performed by setting the TSI control register (TSICR0 and TSICR1) and using an internal AD converter.

3.20.1 TSI External Connection Diagram and Internal Block Diagram

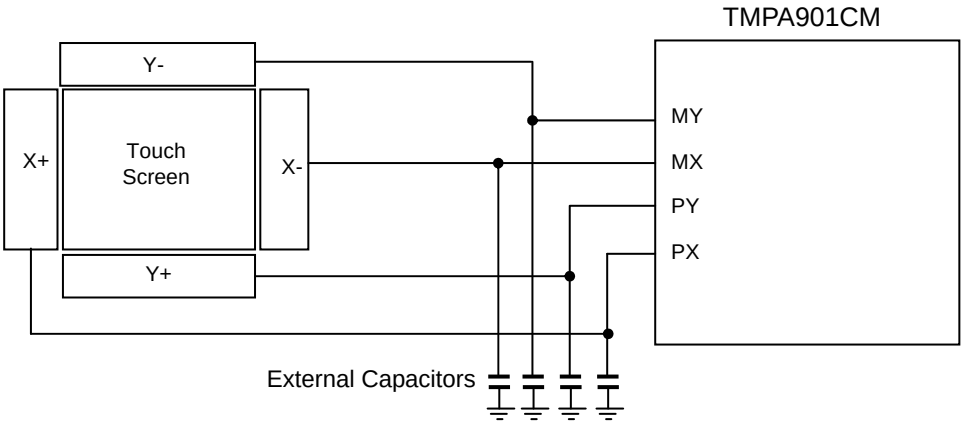


Figure 3.20.1 External connection of TSI

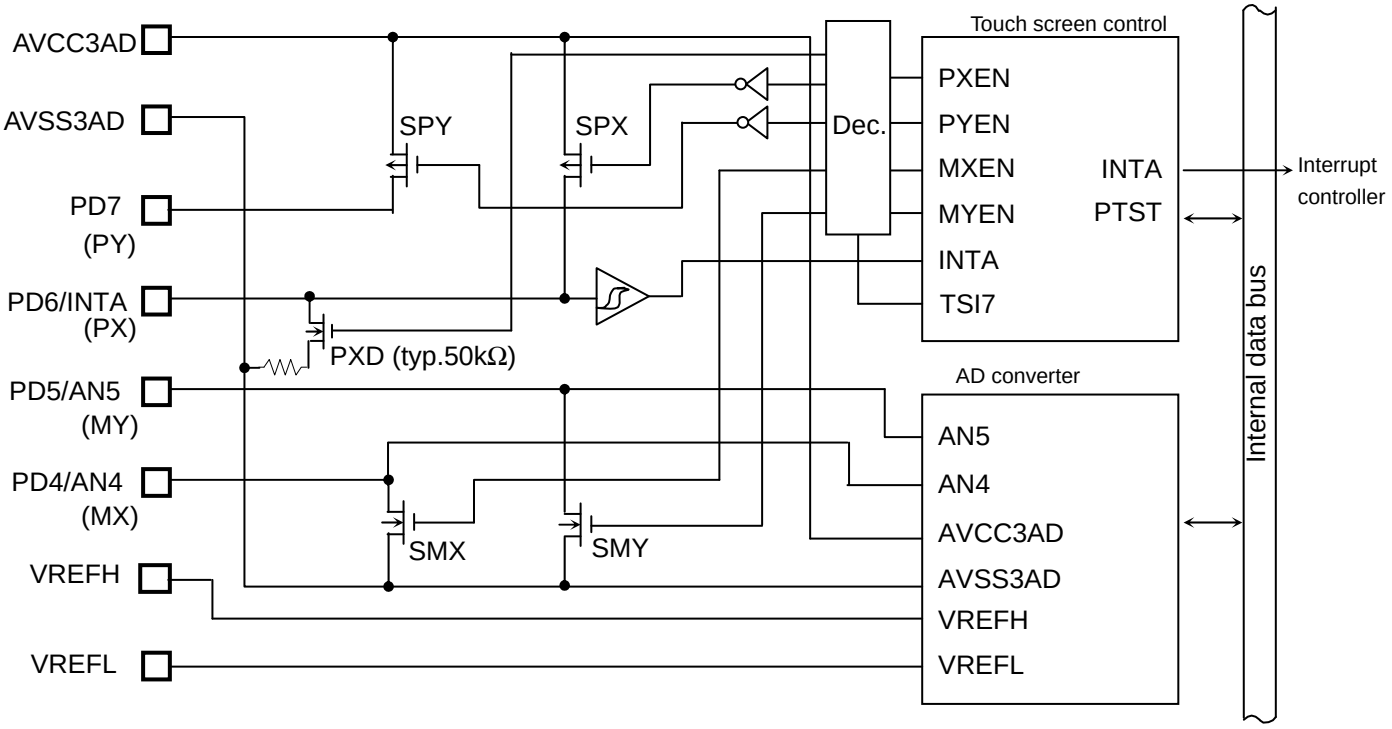


Figure 3.20.2 Internal block diagram of TSI

3.20.2 SFR

The following lists the SFRs:

Base address = 0xF006_0000

Register Name	Address (base+)	Description
TSICR0	0x01F0	TSI Control Register0
TSICR1	0x01F4	TSI Control Register1

1. TSICR0 (TSI Control Register0)

Address = (0xF006_0000) + (0x01F0)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	TSI7	R/W	0y0	pull-down resistor(refer to Description) 0y0: Disable 0y1: Enable
[6]	INGE	R/W	0y0	Input gate control of Port PD6, PD7 0y0: Enable 0y1: Disable
[5]	PTST	RO	0y0	Detection condition Read: 0y0: No touch 0y1: Touch Write: Invalid
[4]	TWIEN	R/W	0y0	INTA interrupt control 0y0: Disable 0y1: Enable
[3]	PYEN	R/W	0y0	SPY 0y0: OFF 0y1: ON
[2]	PXEN	R/W	0y0	SPX 0y0: OFF 0y1: ON
[1]	MYEN	R/W	0y0	SMY 0y0: OFF 0y1: ON
[0]	MXEN	R/W	0y0	SMX 0y0: OFF 0y1: ON

Note: To avoid a flow-through current to the normal C-MOS input gate when converting analog input data by using the AD converter, TSICR0<INGE> can be controlled. If the intermediate voltage is input, cut the input signal to the C-MOS logic (PD6, PD7) by setting this bit. TSICR0<PTST> is to confirm the initial pen-touch. Note that, when the input to the C-MOS logic is blocked by TSICR0<INGE>, this bit is always 1.

[Description]

a. <TSI7>

PXD (Internal pull-down resistor) ON/OFF

<TSI7> \ <PXEN>	0	1
0	OFF	OFF
1	ON	OFF

2. TSICR1 (TSI Control Register1)

Address = (0xF006_0000) + (0x01F4)

Bit	Bit Symbol	Type	Reset Value	Description	
[31:8]	–	–	Undefined	Read as undefined. Write as zero.	
[7]	DBC7	R/W	0y0	0y0: Disable 0y1: Enable	
[6]	DB1024	R/W	0y0	1024	De-bounce time is set by the "(N × 64-16)/f_{PCLK}" formula. "N" is the sum of the numbers obtained when 1 is set in bit 6 to bit 0. (Note 2)
[5]	DB256	R/W	0y0	256	
[4]	DB64	R/W	0y0	64	
[3]	DB8	R/W	0y0	8	
[2]	DB4	R/W	0y0	4	
[1]	DB2	R/W	0y0	2	
[0]	DB1	R/W	0y0	1	

Note 1: Since several pulses of f_{PCLK} are used to synchronize the PD6/INTA signal before it is input to the counter circuit for counting the debounce time, the actual debounce time is the above period plus an extra synchronization period.

Note 2: For example, when TSICR1 = 0x95, N = 64 + 4 + 1 =69. In this case, the debouce time is 44 μs plus an extra synchronization period when f_{PCLK} = 100 MHz.

3.20.3 Touch Detection Procedure

The touch detection procedure includes the procedures starting from when the pen is touched onto the touch screen and until the pen-touch is detected.

Touching the screen generates the interrupt INTA and terminates this procedure. After an X/Y position measuring at INTA interrupt routine, and an X/Y position measuring procedure is terminated, return to this procedure to wait for the next touch.

When waiting for a touch with no contact, set only the SPY switch to ON and set all other three switches (SMY, SPX, SMX) to OFF. At this time, the pull-down resistor built in the PD6/INTA/PX pin is set to ON.

In this state, because the internal X- and Y-direction resistors in the touch screen are not connected, the PD6/INTA/PX pin is set to Low by the internal pull-down resistor (PXD), generating no INTA interrupt.

When a next pen-touch is given, the X- and Y-direction internal resistors in the touch screen are connected, which sets the PD6/INTA/PX pin to High and generates an INTA interrupt.

To avoid generating more than one INTA interrupt by one pen-touch, the de-bounce circuit as shown below is provided. Setting de-bounce time in the TSICR1 register ignores pulses whose time equals to or is below the set time.

The de-bounce circuit detects a rising of signal to count up a set de-bounce counter time and then captures the signal into the inside after counting. When the signal turns to "L" during counting, the counter is cleared, starting to wait for a rising edge again.

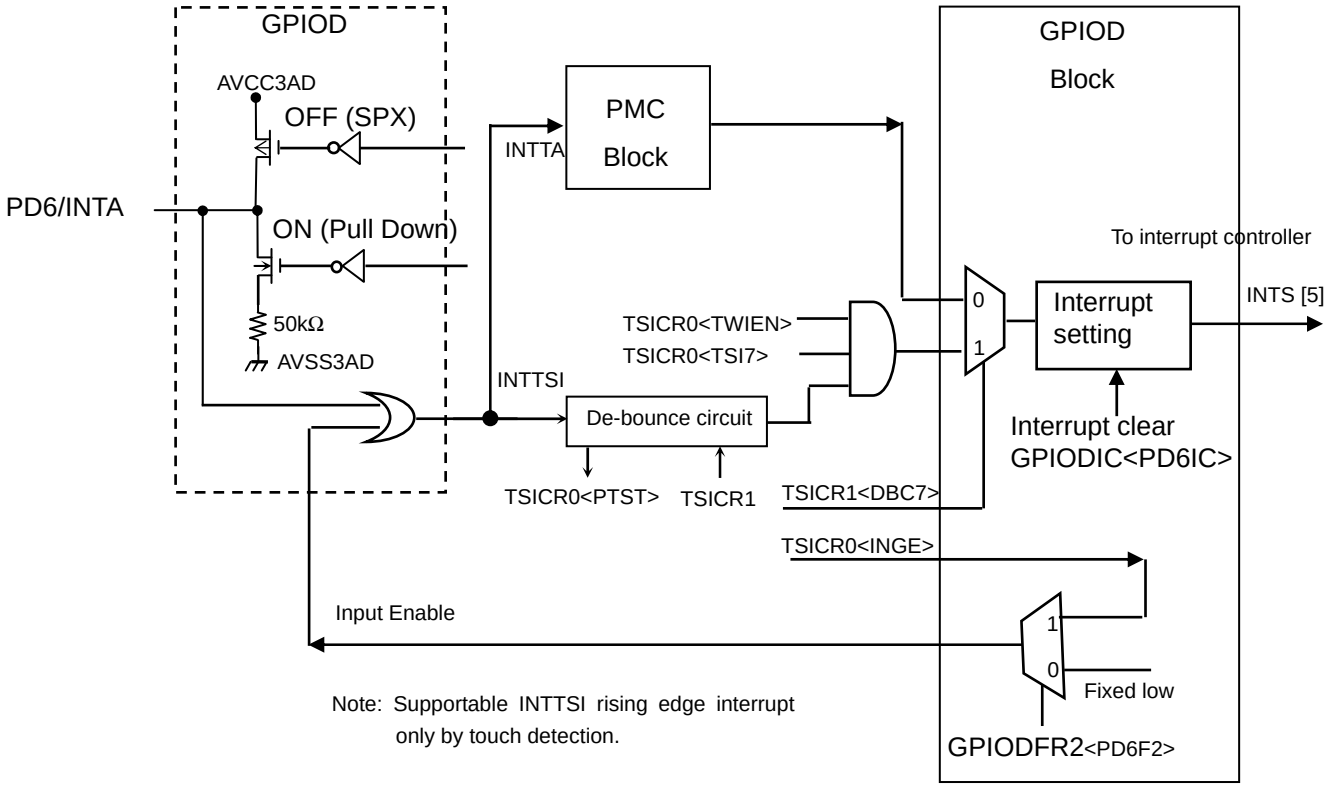


Figure 3.20.3 Block diagram of touch detection

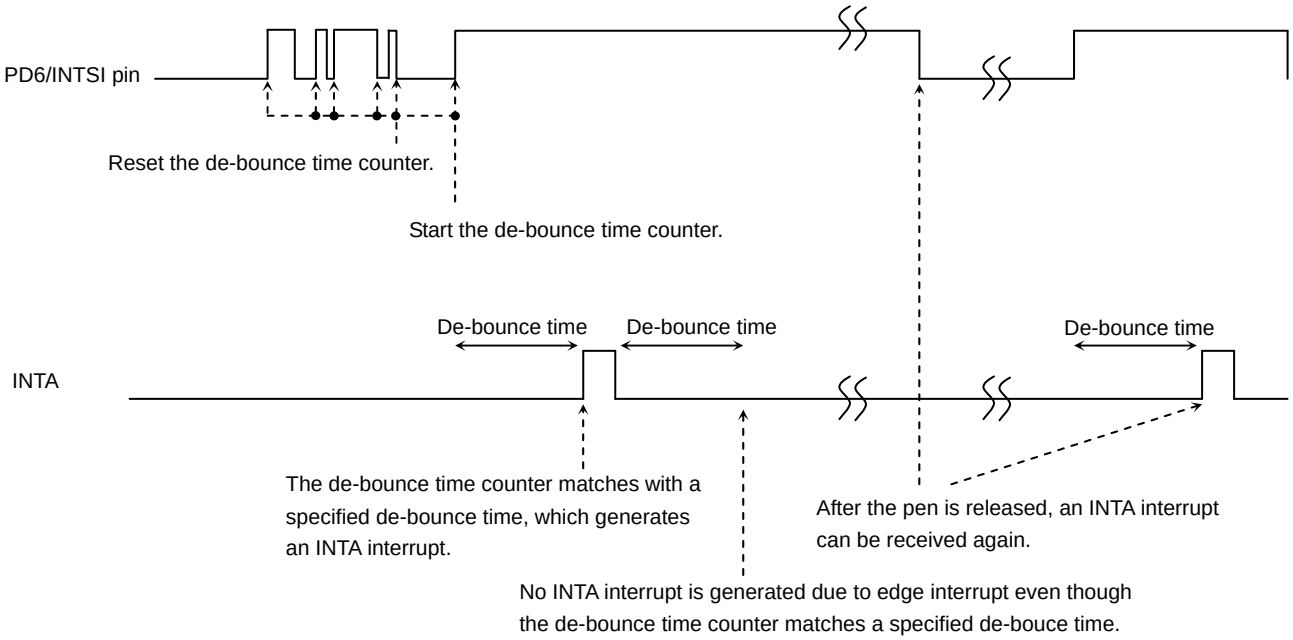


Figure 3.20.4 Timing diagram of de-bounce circuit

3.20.4 X/Y Position Measuring Procedure

During the routine of pen-touch and INTA interrupt generation, execute a pen position measuring following the procedure below:

<X-position coordinate measurement>

Make the SPX and SMX switches ON, and the SPY and SMY switches OFF. With this setting, an analog voltage that shows the X position will be input to the PD5/MY/AN5 pin. The X-position coordinate can be measured by converting this voltage into digital code using the AD converter.

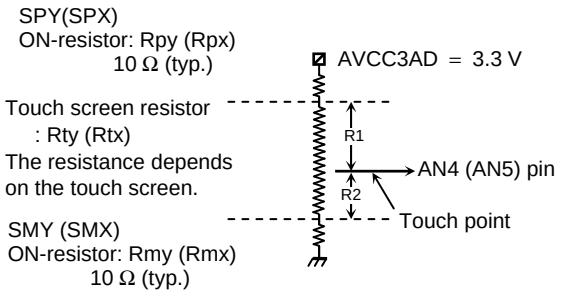
<Y-position coordinate measurement>

Make the SPY and SMY switches ON, and the SPX and SMX switches OFF. With this setting, an analog voltage that shows the Y position will be input to the PD4/MX/AN4 pin. The Y-position coordinate can be measured by converting this voltage into digital code using the AD converter.

The analog voltage which is input to the AN5 and AN4 pins during the X and Y position measurement above can be determined with the ratio between the ON resistance value of the switch in the TMPA900CM and the resistance value in the touch screen as shown in Figure3.20.5

Therefore, even when touching an end area on the touch screen, the analog input voltage will be neither 3.3 V nor 0 V.

Note that the rate of each resistance varies. Remember to take this into consideration during designing. It is also recommended that an average taken from several AD conversions performed if required be adopted as the final correct value.



[Analog input voltage to the AN4 and AN5 pins: Formula to calculate E1]

$$E1 = ((R2 + Rmy) / (Rpy + Rty + Rmy)) \times AVCC3AD[V]$$

Example) Where

AVCC3AD = 3.3 V, Rpy = Rmy = 10 Ω R1 = 400 Ω and R2 = 100 Ω

$E1 = ((100 + 10) / (10 + 400 + 100 + 10) \times 3.3$

$= 0.698$ V

Note 1: An X-coordinate position can be calculated in the same way though above formula is for Y-coordinate position.

Note 2: Rty = R1 + R2

Figure3.20.5 Analog input voltage calculation values

3.20.5 Flow Chart of Touch Screen Interface (TSI)

(1) Touch detection procedure (2) X/Y position measuring procedure

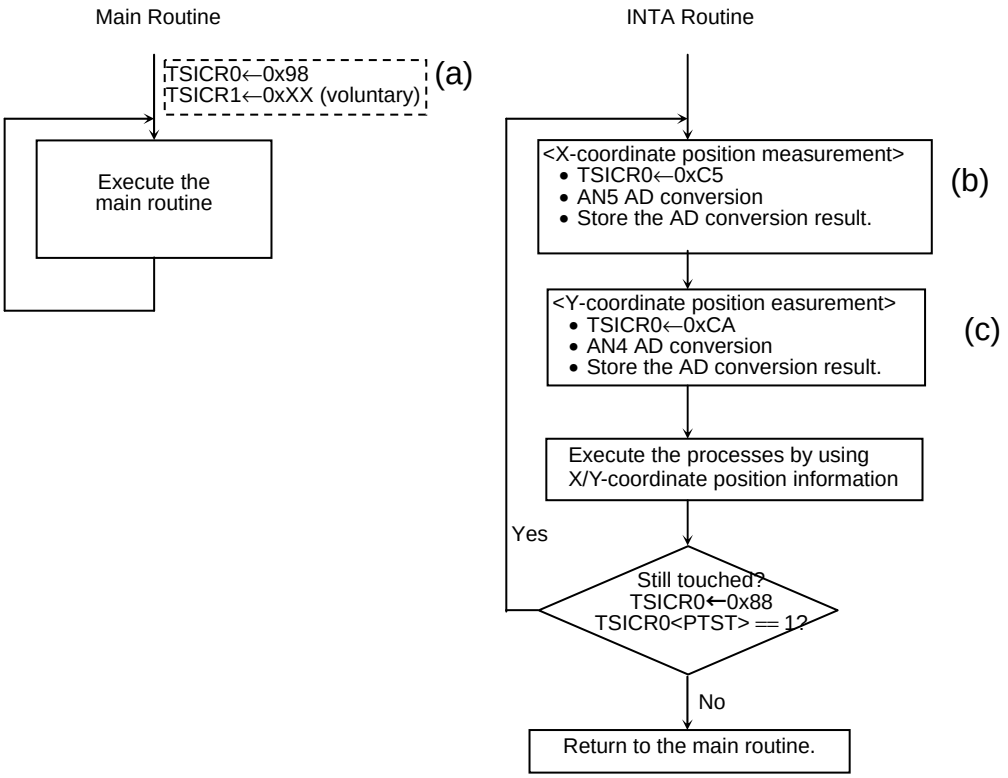
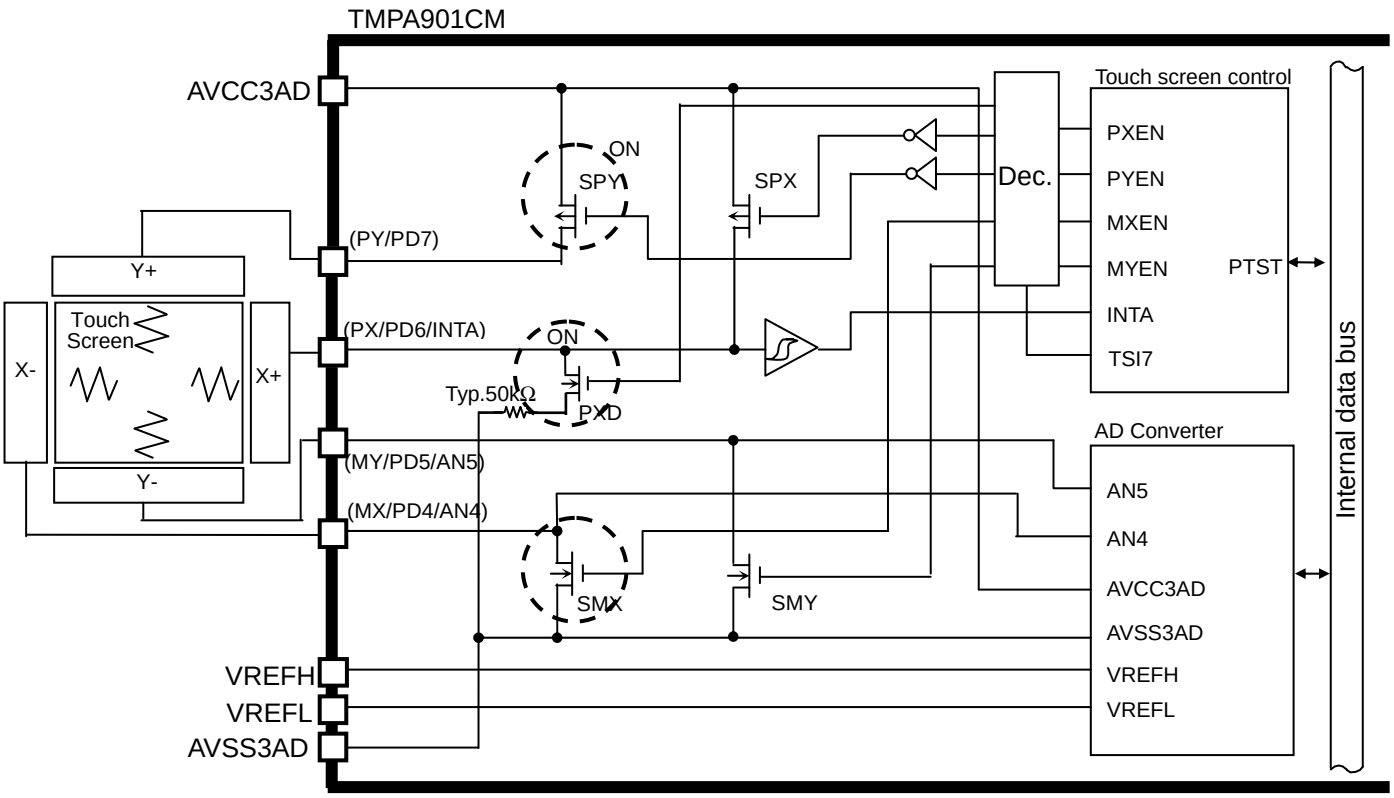


Figure 3.20.6 Flow example for TSI

The following pages explain each circuit condition (a), (b) and (c) in the flow chart above:

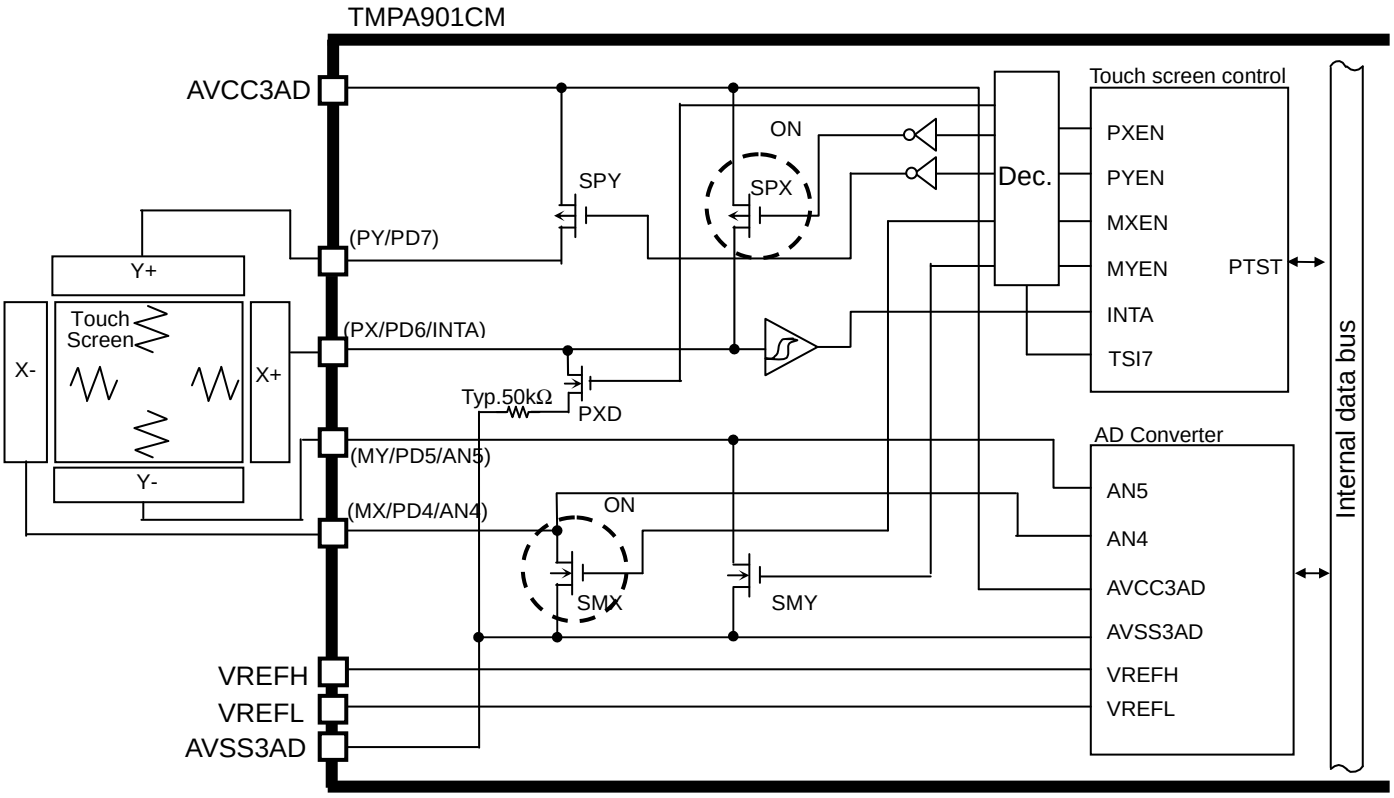
(a) Main routine: condition of waiting for INTA interrupt

```
GPIODFR1    ← 0x00000030    ;PD[6] INTA,PD[5]AN5 , PD[4]AN4
GPIODFR2    ← 0x000000C0    ;PD[7]PY , PD[6] PX
GPIODIS     ← 0x00000000    ;set INTA edge interrupt
GPIODIBE    ← 0x00000000    ;set INTA single edge
GPIODIE     ← 0x00000040    ;enable INTA
GPIODIEV    ← 0x00000040    ;set INTA positive edge
TSICR0      ← 0x00000098    ;[7] enable TSI / PXD ON
                            ;[4] INTA enable
                            ;[3] SPY:ON
```



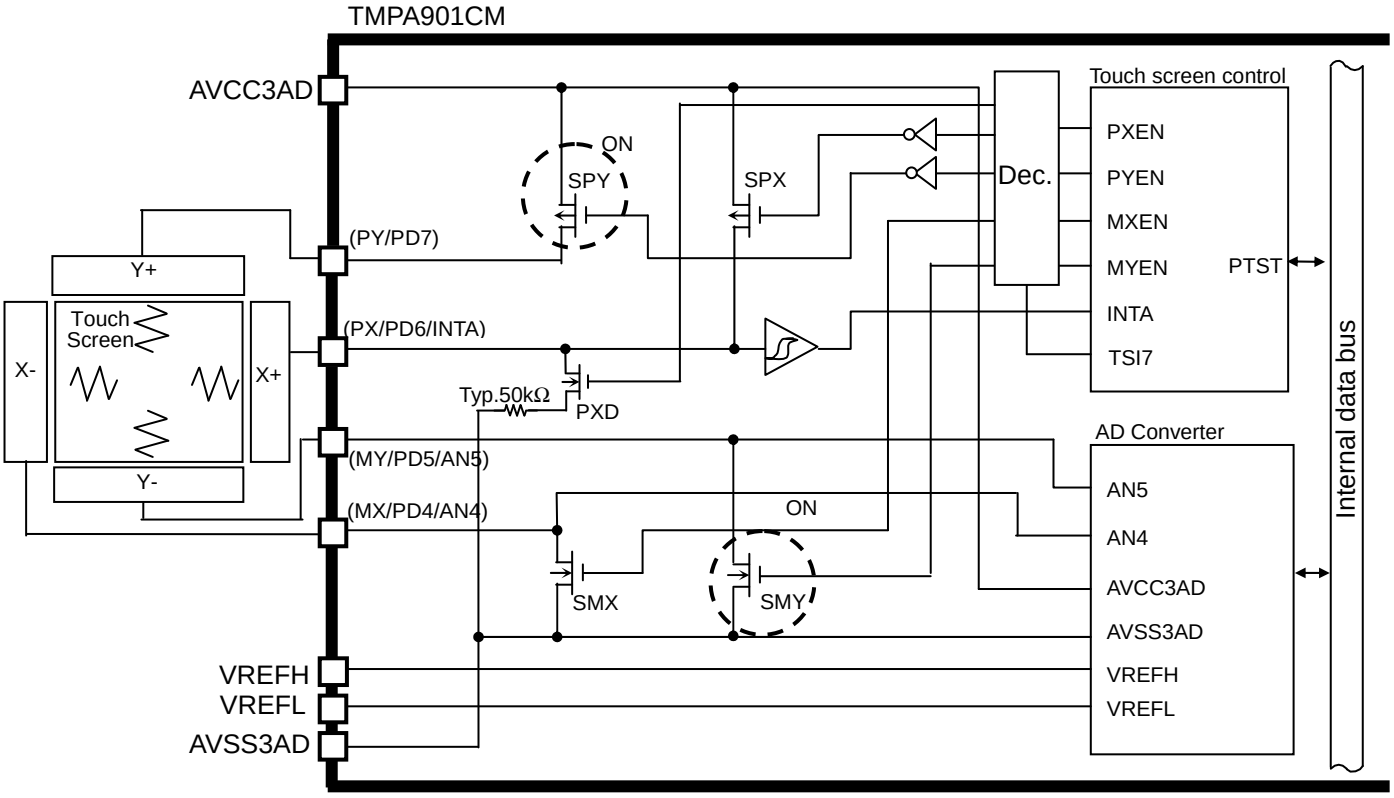
(b) INTA routine: X-position coordinate measurement (AD conversion start)

```
GPIODIC   ← 0x00000040   ;INT request clear
TSICR0    ← 0x00000085   ;Disable INTA
TSICR0    ← 0x000000C5   ;[7] enable TSI
                         ;[6] PD6/PD7 : disable input
                         ;[2] SPX:ON , [0] SMX:ON
ADMOD1    ← 0x00000085   ;Set AN5
ADMOD0    ← 0x00000001   ;Start AD conversion
```



(c) INT4 routine: Y-position coordinate measurement (AD conversion start)

```
TSICR0      ← 0x000000CA    ;[7] enable TSI
                            ;[6] PD6/PD7 : disable input
                            ;[3] SPY:ON , [1] SMY:ON
ADMOD1      ← 0x00000084    ;set AN4
ADMOD0      ← 0x00000001    ;start AD conversion
```



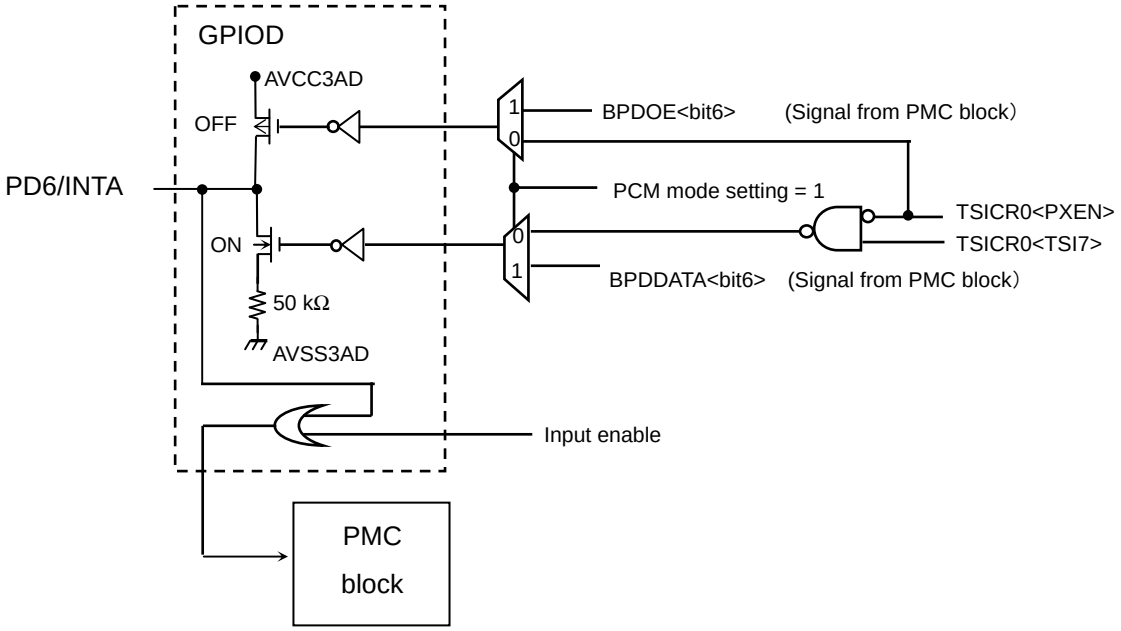
3.20.6 Considerations for Using the TSI

1. Recovery from PCM state

In PCM state, Power supply of TSI circuit is turned off. However, it can recover from PCM state because of the touch detection is inputted to PMC from PD6/INTA input pin directly.

Setting example:

```
BPDRINT   ←  0x00000000   ; write 0x00000000 to Register
                          ; INT status initial
BPARINT   ←  0x00000000   ; INT status initial
BPPRINT   ←  0x00000000   ; INT status initial
BSMRINT   ←  0x00000000   ; INT status initial
BRTRINT   ←  0x00000000   ; INT status initial
BPDEDGE   ←  0x00000000   ; Rising edge
BPDOE     ←  0x000000B0   ; PD7(SPY):ON ,PD6(SPX):OFF
                          ; PD5(SMY):OFF , PD4(SMX):OFF
BPDDATA   ←  0x00000000   ; PD6(PXD):ON (pull down)
BPDRELE   ←  0x00000040   ; PCM release Enable by INTA
…         ←  …            ; and other setting before enter PCM
                          ; refer to PMC chapter
```



Note: If it is waked up from PCM state by INTA, the interrupt edge is usable rising /falling both edges. However, if using with TSI, it recommends using rising edge.

2. Port setting

When an intermediate voltage between 0V and AVCC3AD is converted by the AD converter, the intermediate voltage is also applied to normal C-MOS input gates due to the circuit structure.

Take measures against the floating current to PD6 and PD7 by setting TSICR0<INGE> = 1. When the input to the C-MOS logic is cut off, TSICR0<PTST> that indicates whether or not a pen touch is detected is always set to 1.

3.21 Real-Time Clock/Melody Alarm Generator (RTCMLD)

3.21.1 Functional Overview

The circuits include Real-Time Clock, Melody and Alarm generator block.

The base clock is 32 kHz low frequency.

Each circuit function is shown below.

1) Melody:

- Can generate melody waveforms at any frequency from 4 Hz to 5461 Hz.

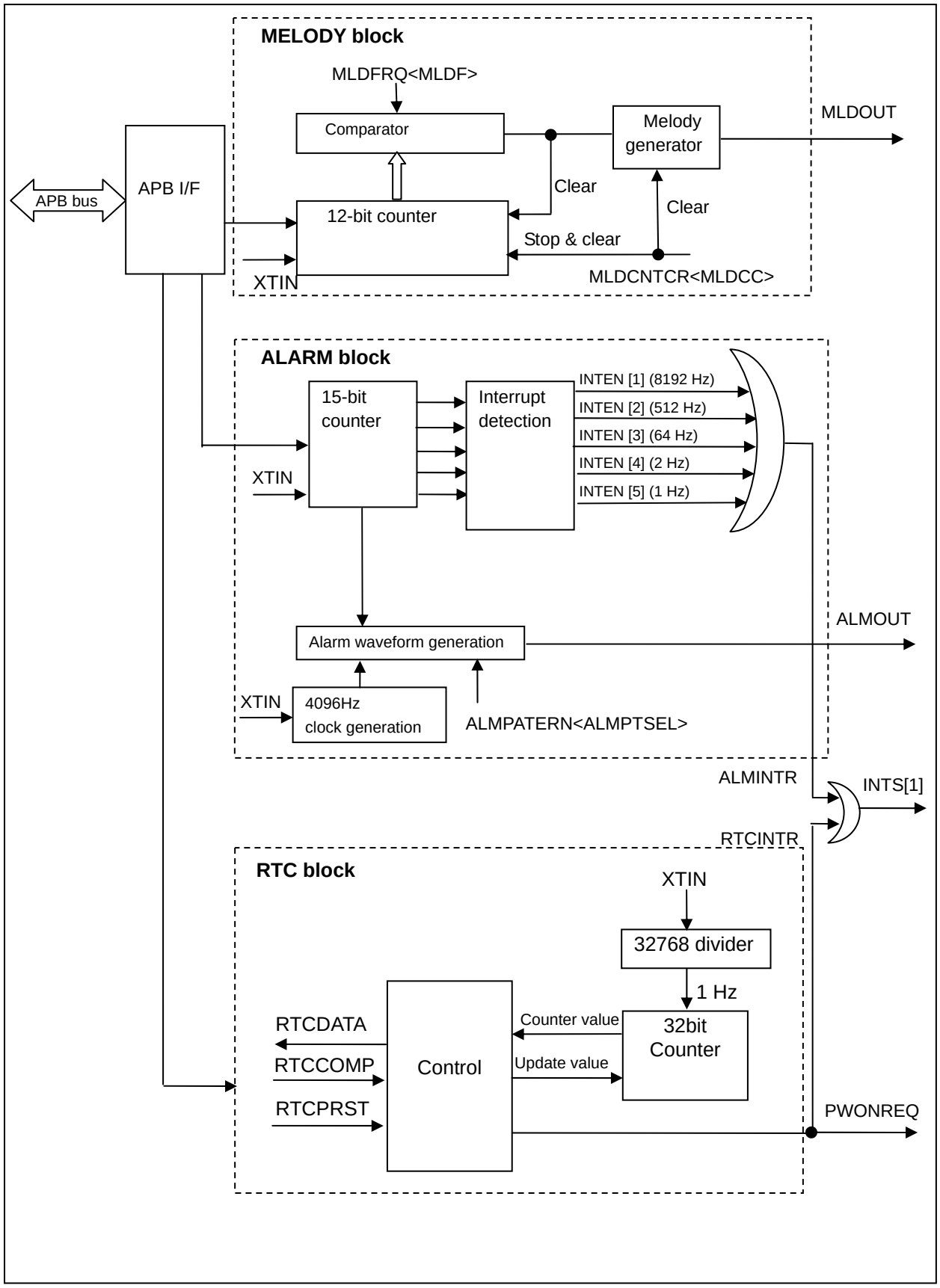
2) Alarm:

- Can generate eight patterns of alarm output.
- Can generate five types of fixed-interval interrupts (1 Hz, 2 Hz, 64 Hz, 512 Hz and 8192 Hz).

3) RTC:

- 32-bit counter that counts up every second
- Compare 32-bits counter value, and generate interrupt requests (Resume request to the PMC is also corresponded)

3.21.2 Block Diagram



3.21.3 Operational Description

3.23.3.1 Melody Generator

(1) Operational overview

Based on the low-speed clock (32.768 kHz), clock waveforms at any frequency from 4 Hz to 5461 Hz can be generated and output from the MLDALM pin.

By connecting buzzer etc outside, melody sounds can easily be played.

The melody frequency is calculated as below.

XTIN = 32.768 [kHz]

Melody output waveform $f_{MLD}[Hz] = 32768 / (2 \times N + 4)$

Melody setting value $N = (16384 / f_{MLD}) - 2$

Note: The value N is set through the MLDFRQ<MLDF> register:
N = 1 to 4095 (0x001 to 0xFFF)
Setting N = 0 is prohibited.

*For the above equation, see the waveform diagram below.

(For reference: Basic tone scale setting table)

Tone scale	Frequency [Hz]	MLDFRQ<MLDF> Register value: N
C	264	0x03C
D	297	0x035
E	330	0x030
F	352	0x02D
G	396	0x027
A	440	0x023
B	495	0x01F
C	528	0x01D

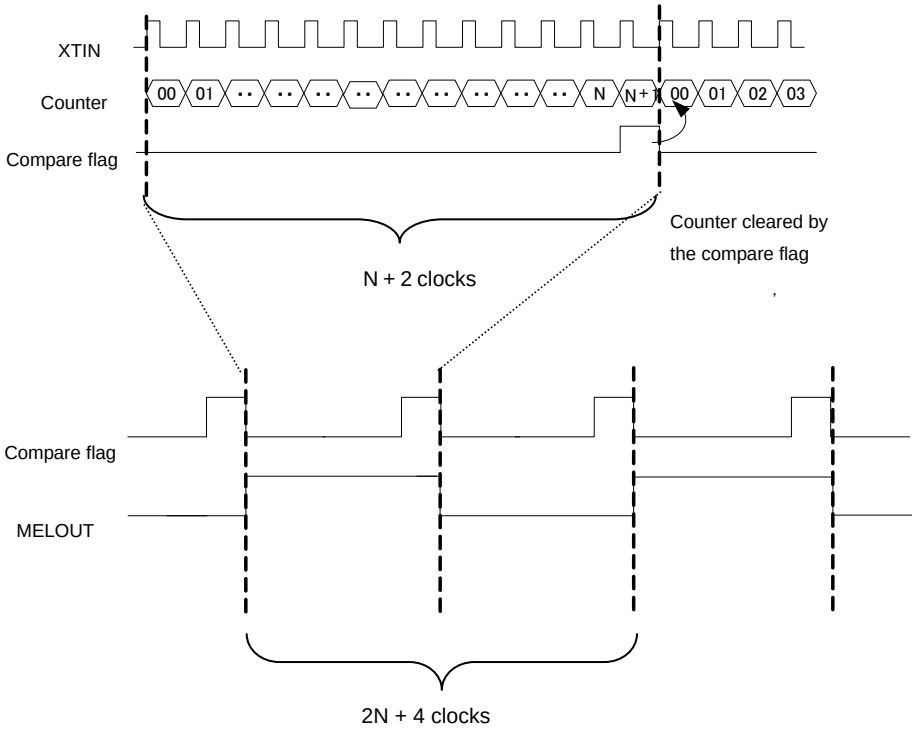


Figure 3.21.1 Melody waveform

(2) Flowchart for melody setting

Note: The MLDFRQ in the flowchart can not read. Therefore, switch to the next step after the data was written and the 3-clocks of 32kHz (approx. 93 μs) passed. In the other registers, switch to the next step after the data was written and confirmed the updated data by data polling.

3.23.3.2 Alarm Generator

(1) Operational overview

At the frequency (4096 Hz) divided based on the low-speed clock (32.768 kHz), eight types of alarm waveforms can be generated and output from the MLDALM pin.

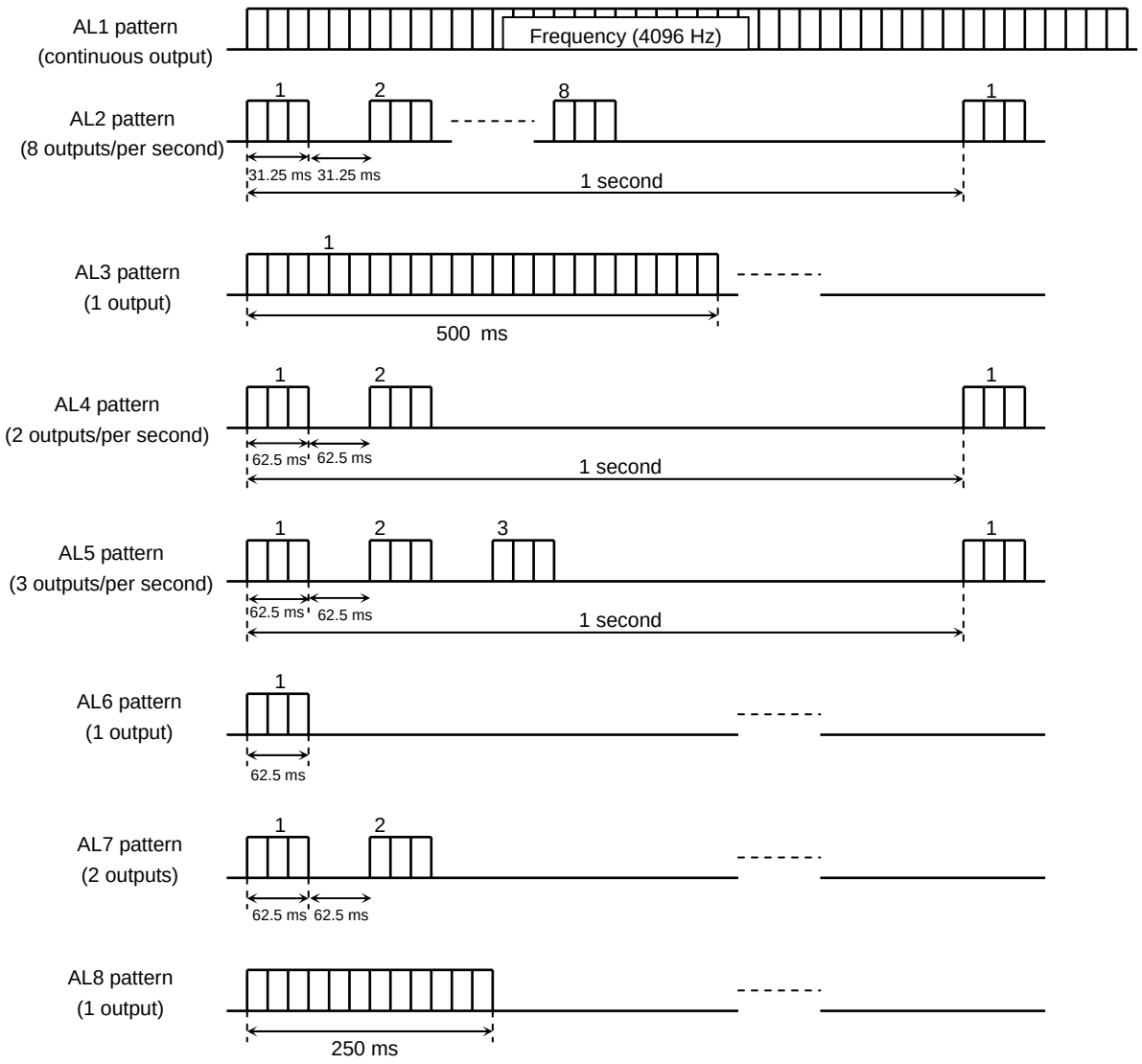
By connecting buzzer etc outside, alarm sounds can easily be played.

The free-running counter in the alarm generator can be used to generate five types of interval interrupts (1 Hz, 2 Hz, 64 Hz, 512 Hz and 8192 Hz).

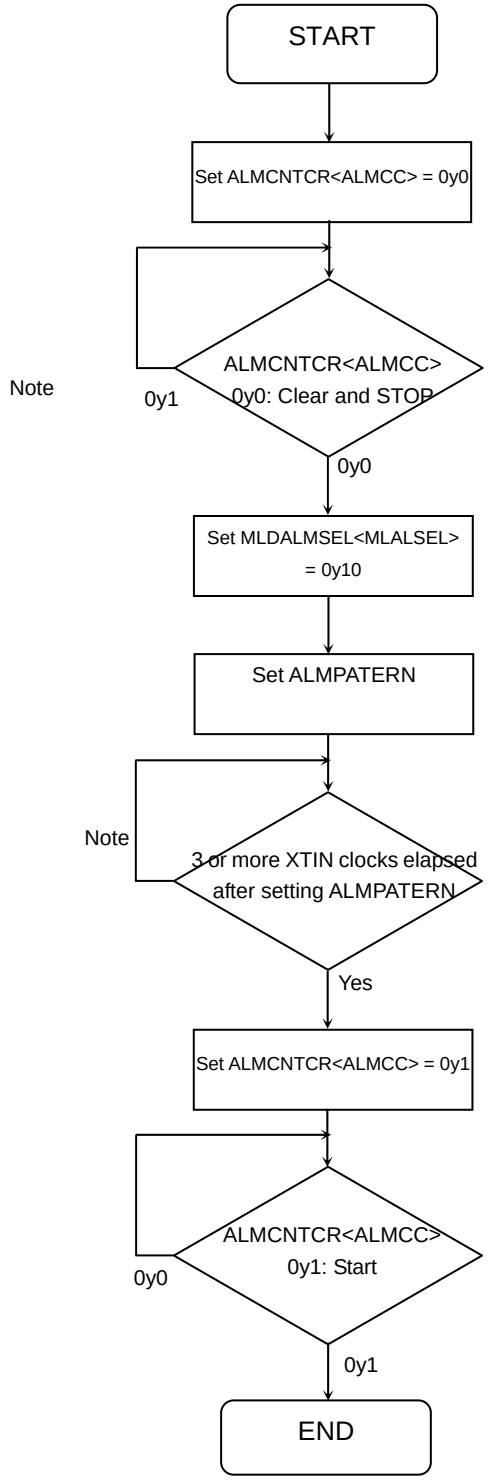
(Alarm pattern setting table)

ALM register setting	Alarm waveform
0x00	Fixed at 0
0x01	AL1 pattern
0x02	AL2 pattern
0x04	AL3 pattern
0x08	AL4 pattern
0x10	AL5 pattern
0x20	AL6 pattern
0x40	AL7 pattern
0x80	AL8 pattern
Others	Undefined (Setting prohibited)

Example: Various alarm waveform patterns



(2) Flowchart for alarm generator setting



Note: The ALMPATERN register in the flowchart can not read. Therefore, switch to the next step after the data was written and the 3-clocks of 32kHz (approx. 93 μs) passed. In the other registers, switch to the next step after the data was written and confirmed the updated data by data polling.

3.21.4 Real-Time Clock

(1) Operational overview

The real-time clock (32 bit counter) can count every second based on the frequency (1 Hz) divided from the low-speed clock (32.768 kHz).

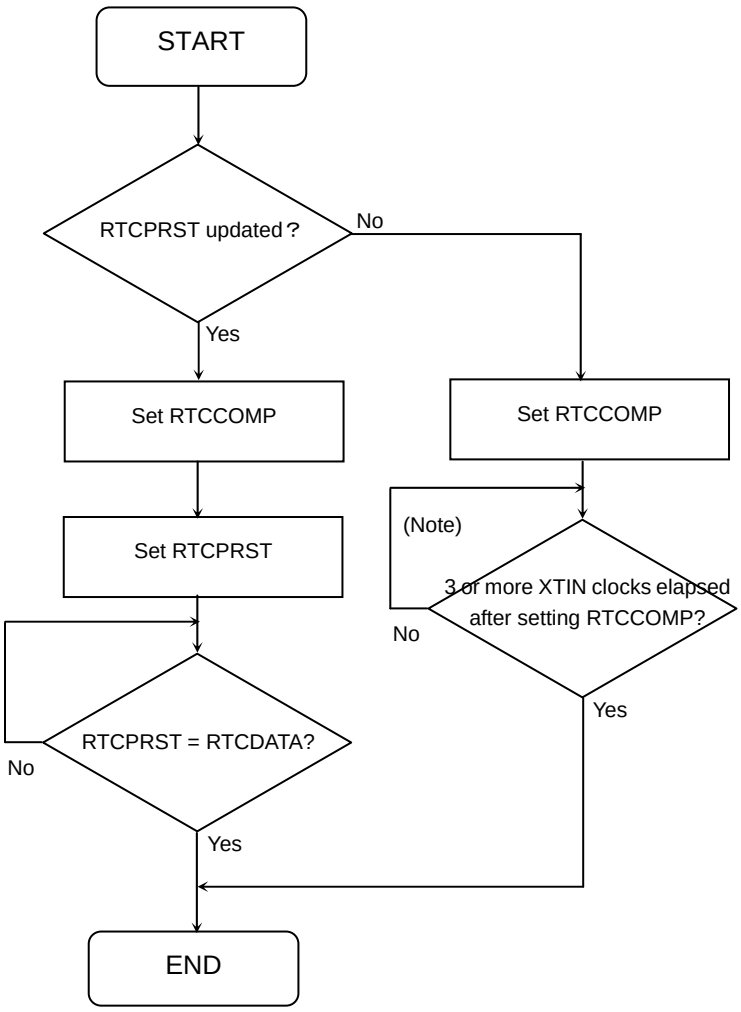
The current count value can be read from the RTCDATA register. Clock function can be easily realized

By comparing the count value with the value set in the RTCCOMP register, an interrupt can be generated (can also be used as the resume signal from PCM (Power Cut Mode) state).

The counter value can be changed arbitrarily by setting a value in the RTCPRST register.

To keep counting time, power supply is always turned on. Even if there is no external factor to Wakeup from PCM mode, RTC can issue a power resume request to the PMC, the operation of the 1A power supply circuit can be resumed.

(2) Flowchart for real-time clock setting



Note: Switch to the next step after the data was written and the 3-clocks of 32 kHz (approx. 93 μs) passed.

3.21.5 Register descriptions

The following lists the SFRs:

Base address = 0xF003_0000

Register Name	Address (base+)	Description
RTCDATA	0x0000	RTC Data Register
RTCCOMP	0x0004	RTC Compare Register
RTCPRST	0x0008	RTC Preset Register
MLDALMINV	0x0100	Melody Alarm Invert Register
MLDALMSEL	0x0104	Melody Alarm signal Select Register
ALMCNTCR	0x0108	Alarm Counter Control Register
ALMPATERN	0x010C	Alarm Pattern Register
MLDCNTCR	0x0110	Melody Counter Control Register
MLDFRQ	0x0114	Melody Frequency Register
RTCALMINTCTR	0x0200	RTC ALM Interrupt Control Register
RTCALMMIS	0x0204	RTC ALM Interrupt Status Register

1. RTCDATA (RTC Data Register)

Address = (0xF003_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	RTCCOUNT	RO	Undefined	32-bit counter value

[Description]

a. <RTCCOUNT>
Returns the RTC count value (32 bits) on read.

2. RTCCOMP (RTC Compare Register)

Address = (0xF003_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	RTCCP	WO	Undefined	Value to be compared with the RTC counter

[Description]

a. <RTCCP>
A match between the counter value (in steps of 1 HZ) and the value in <RTCCP> generates an interrupt and power supply resume request.

3. RTCPRST (RTC Preset Register)

Address = (0xF003_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	RTCPR	WO	Undefined	RTC counter preset value

[Description]

a. <RTCPR>

Specifies the RTC counter preset value.

4. MLDALMINV (Melody Alarm signal Invert Register)

Address = (0xF003_0000) + (0x0100)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	MLALINV	WO	0y0	MLDALM output signal inversion 0y0: Do not invert 0y1: Invert

[Description]

a. <MLALINV>

Selects whether or not to invert the melody/alarm output.

5. MLDALMSEL (Melody Alarm Select Register)

Address = (0xF003_0000) + (0x0104)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Write as zero.
[1:0]	MLALSEL	WO	0y00	Output signal select 0y00: Stop output 0y01: Melody 0y10: Alarm 0y11: Setting prohibited

[Description]

a. <MLALSEL>

Selects the melody or alarm output from MLDALM pin.

6. ALMCNTCR (Alarm Counter Control Register)

Address = (0xF003_0000) + (0x0108)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	ALMCC	R/W	0y0	Free-running counter control 0y0: Clear and stop 0y1: Start

[Description]

a. <ALMCC>

Controls the 15-bit counter for alarm generation.

7. ALMPATERN (Alarm Pattern Register)

Address = (0xF003_0000) + (0x010C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	ALMPTSEL	WO	0x00	Alarm pattern setting (See the table below.)

[Description]

a. <ALMPTSEL>

Selects the alarm pattern to be output.

[Alarm pattern setting values]

ALMPTSEL setting	Alarm waveform
0x00	Fixed at 0
0x01	AL1 pattern
0x02	AL2 pattern
0x04	AL3 pattern
0x08	AL4 pattern
0x10	AL5 pattern
0x20	AL6 pattern
0x40	AL7 pattern
0x80	AL8 pattern
Others	Undefined (Setting prohibited)

8. MLDCNTCR (Melody Counter Control Register)

Address = (0xF003_0000) + (0x0110)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined. Write as zero.
[0]	MLDCC	R/W	0y0	Free-running counter control 0y0: Clear and stop 0y1: Start

[Description]

a. <MLDCC>
Controls the 12-bit counter for melody generation.

9. MLDFRQ (Melody Frequency Register)

Address = (0xF003_0000) + (0x0114)

Bit	Bit Symbol	Type	Reset Value	Description
[31:12]	–	–	Undefined	Read as undefined. Write as zero.
[11:0]	MLDF	WO	0x000	Melody output frequency setting value N: 0x001 to 0xFFF

[Description]

a. <MLDF >
Specifies the counter value (N) for Melody output frequency setting.
Formula: f_{MLD}[Hz] = 32768/ (2N + 4)

10. RTCALMINTCTR (RTC ALM Interrupt Control Register)

Address = (0xF003_0000) + (0x0200)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	ALMINTCLR	WO	0y0	Alarm interrupt flag clear 0y0: Invalid 0y1: Clear
[6]	RTCINTCLR	WO	0y0	RTC interrupt flag clear 0y0: Invalid 0y1: Clear
[5]	AINTEN1	R/W	0y0	Alarm (1 Hz) interrupt enable 0y0: Disable 0y1: Enable
[4]	AINTEN2	R/W	0y0	Alarm (2 Hz) interrupt enable 0y0: Disable 0y1: Enable
[3]	AINTEN64	R/W	0y0	Alarm (64 Hz) interrupt enable 0y0: Disable 0y1: Enable
[2]	AINTEN512	R/W	0y0	Alarm (512 Hz) interrupt enable 0y0: Disable 0y1: Enable
[1]	AINTEN8192	R/W	0y0	Alarm (8192 Hz) interrupt enable 0y0: Disable 0y1: Enable
[0]	RTCINTEN	R/W	0y0	RTC interrupt enable 0y0: Disable 0y1: Enable

[Description]

a. <ALMINTCLR >
Clears the enabled alarm interrupt flag.

b. <RTCINTCLR>
Clears the enabled RTC interrupt flag.

c. <AINTEN1>
Enables or disables the alarm (1 Hz) interrupt.

d. <AINTEN2>
Enables or disables the alarm (2 Hz) interrupt.

e. <AINTEN64>
Enables or disables the alarm (64 Hz) interrupt.

f. <AINTEN512>
Enables or disables the alarm (512 Hz) interrupt.

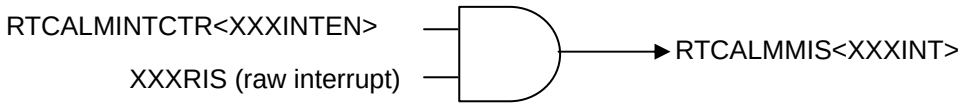
g. <AINTEN8192>
Enables or disables the alarm (8192 Hz) interrupt.

h. <RTCINTEN>
Enables or disables the RTC interrupt.

11. RTCALMMIS (RTC ALM Masked Interrupt Status Register)

Address = (0xF003_0000) + (0x0204)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined.
[1]	ALMINT	RO	0y0	Alarm interrupt enabled status 0y0: Interrupt not requested 0y1: Interrupt requested
[0]	RTCINT	RO	0y0	RTC interrupt enabled status 0y0: Interrupt not requested 0y1: Interrupt requested



- Notes:
  (1) The RTC count-up control register always counts up. The 32 bit counter of RTC can't be stopped.
  (2) After power-on, The RTCCOMP and RTCPRST registers are undefined. So The RTCCOMP and RTCPRST registers must be confined, and confirm the right data, can be read from the RTCDATA register, then can set interrupt to Enable state.

  If the RTCCOMP, RTCPRST and other RTC relation registers are undefined state, don't set MCU to the PCM mode. Unexpected action resume from PCM maybe happen. Please pay attention to it.
  (3) The RTCDATA value is updated after the DVCC1A power supply is resumed by PCM releasing and the approximately one second elapsed. Therefore, do not display time or use time data until one second elapses after the power supply is resumed.
  (4) The melody and alarm voltage circuitry are built in DVCC1A power supply, however RTC voltage circuitry is built in DVCC1B power supply.

3.22 Analog/Digital Converter

A 10-bit serial conversion analog/digital converter (AD converter) having four channels of analog input is built in.

Figure 3.22.1 shows the block diagram of the AD converter. The four channels of analog input pins (AN4 to AN7) are used also as input dedicated ports D (PD4 to PD7).

Note 1: To reduce the power supply current by PCM mode, the standby state may be maintained with the internal comparator still being enabled, depending on the timing. Check that the AD converter operation is in a stop before executing mode switching.

Note 2: Setting ADMOD1<DACON> = 0 while the AD converter is in a stop can reduce current consumption.

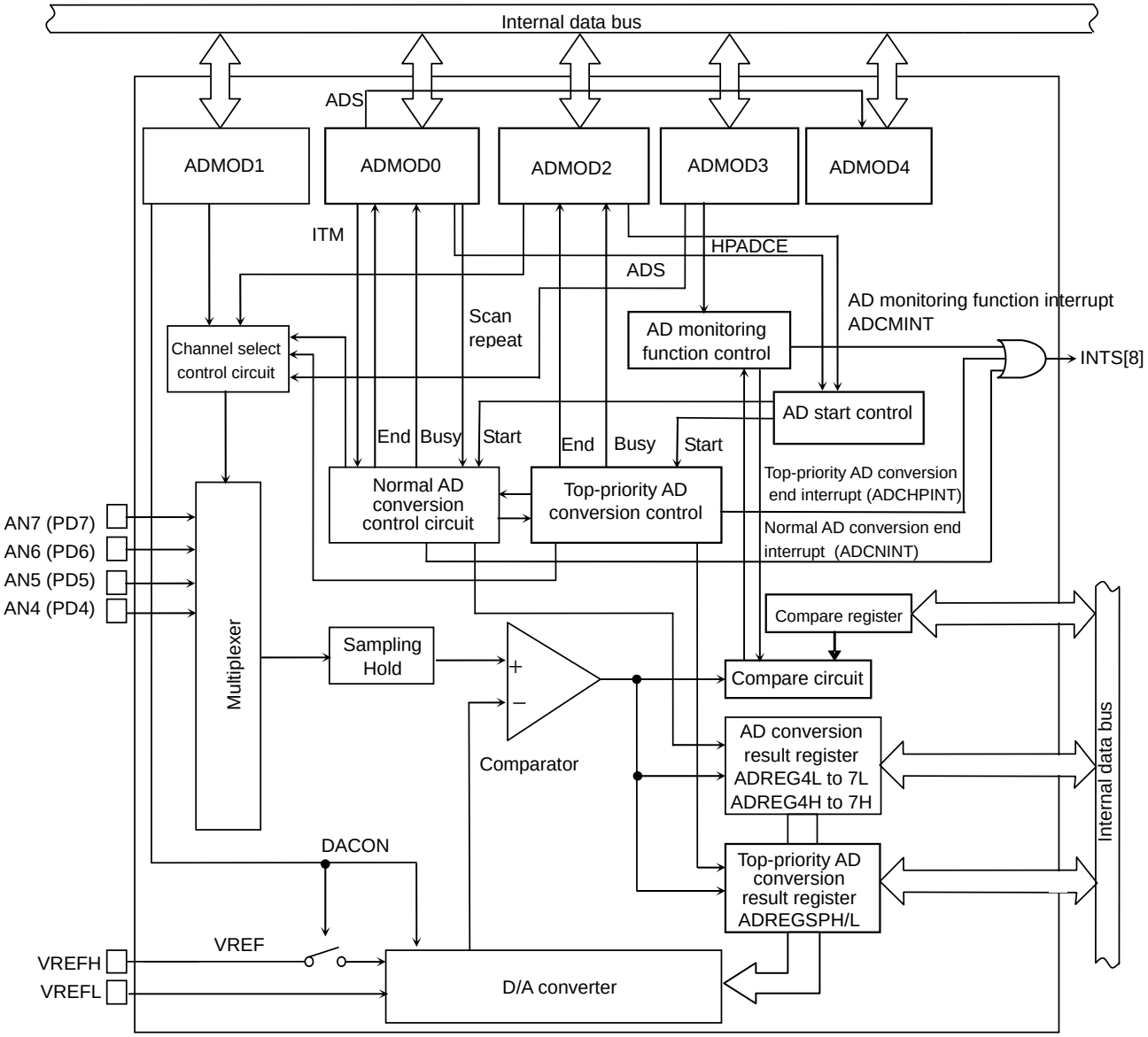


Figure 3.22.1 Block diagram of AD converter

3.22.1 Description of Registers

The following lists the SFRs:

Base address = 0xF008_0000

Register Name	Address (base+)	Description
–	0x0000	Reserved
–	0x0004	Reserved
–	0x0008	Reserved
–	0x000C	Reserved
–	0x0010	Reserved
–	0x0014	Reserved
–	0x0018	Reserved
–	0x001C	Reserved
ADREG4L	0x0020	A/D conversion result lower-order register 4
ADREG4H	0x0024	A/D conversion result higher-order register 4
ADREG5L	0x0028	A/D conversion result lower-order register 5
ADREG5H	0x002C	A/D conversion result higher-order register 5
ADREG6L	0x0030	A/D conversion result lower-order register 6
ADREG6H	0x0034	A/D conversion result higher-order register 6
ADREG7L	0x0038	A/D conversion result lower-order register 7
ADREG7H	0x003C	A/D conversion result higher-order register 7
ADREGSPL	0x0040	Top-priority A/D conversion result lower-order register
ADREGSPH	0x0044	Top-priority A/D conversion result higher-order register
ADCOMREGL	0x0048	A/D conversion result comparison lower-order register
ADCOMREGH	0x004C	A/D conversion result comparison lower-order register
ADMOD0	0x0050	A/D mode control register 0
ADMOD1	0x0054	A/D mode control register 1
ADMOD2	0x0058	A/D mode control register 2
ADMOD3	0x005C	A/D mode control register 3
ADMOD4	0x0060	A/D mode control register 4
–	0x0064	Reserved
–	0x0068	Reserved
–	0x006C	Reserved
ADCLK	0x0070	A/D conversion clock setting register
ADIE	0x0074	A/D interrupt enable register
ADIS	0x0078	A/D interrupt status register
ADIC	0x007C	A/D interrupt clear register
–	0x0080	Reserved
–	:	Reserved
–	0x0FFF	Reserved

Note: Notes for Description of Registers
R/W: Read/Write possible
RO: Readable / Write not reflected
WO: Writable / 0 can be read when read

3.22.1.1 Control Registers

The A/D converter is controlled by the A/D mode control registers (ADMOD0, ADMOD1, ADMOD2, ADMOD3, and ADMOD4). A/D conversion results are stored in the four registers of A/D conversion result higher-order/lower-order registers ADREG4H/L to ADREG7H/L. Top-priority conversion results are stored in ADREGSPH/L.

- ADMOD register

1. ADMOD0 (AD mode control register 0)

Address = (0xF008_0000) + (0x0050)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	EOCFN (Note)	RO	0y0	Normal AD conversion end flag Read: 0y0: Before conversion or being converted 0y1: End Write: Invalid
[6]	ADBFN	RO	0y0	Normal AD conversion BUSY flag Read: 0y0: Conversion stop 0y1: Being converted Write: Invalid
[5]	–	RO	0y0	Always read as 0 when read.
[4]	–	R/W	0y0	Always write as 0
[3]	ITM	R/W	0y0	The A/D conversion interrupt during channel fix and repeat conversion mode. ITM 0y0: Generates an interrupt per one conversion 0y1: Reserved
[2]	REPEAT	R/W	0y0	Specifies repeat mode. 0y0: Single conversion mode 0y1: Repeat conversion mode
[1]	SCAN	R/W	0y0	Specifies scan mode. 0y0: Channel fix mode 0y1: Channel scan mode
[0]	ADS	R/W	0y0	A/D conversion start 0y0: Don't care 0y1: Start Conversion Always read as 0 when read.

R/W : Read/Write RO : Read Only WO : Write Only

Note: As read this register, <EOCFN> is clear to 0.

[Description]

a. <EOCFN>

They are the normal AD conversion end flag.

0y0: Before conversion or being converted

0y1: End

b. <ADBFN>

They are the normal AD conversion BUSY flag.

0y0: Conversion stop

0y1: Being converted

c. <ITM>

The A/D conversion interrupt during channel fix and repeat conversion mode.

0y0: Generates an interrupt per one conversion

0y1: Reserved

d. <REPEAT>

Selects the repeat mode.

0y0: Single conversion mode

0y1: Repeat conversion mode

e. <SCAN >

Selects the scan mode.

0y0: Channel fix mode

0y1: Channel scan mode

f. <ADS>

Selects the A/D conversion start mode.

0y0: Don't care

0y1: Start Conversion mode

Always read as 0 when read.

2. ADMOD1 (AD mode control register 1)

Address = (0xF008_0000) + (0x0054)

<table>
<tr><th>Bit</th><th>Bit Symbol</th><th>Type</th><th>Reset Value</th><th>Description</th></tr>
<tr><td>[31:8]</td><td>–</td><td>–</td><td>Undefined</td><td>Read as undefined. Write as zero.</td></tr>
<tr><td>[7]</td><td>DACON</td><td>R/W</td><td>0y0</td><td>VREF application control
0y0: OFF
0y1: ON</td></tr>
<tr><td>[6]</td><td>–</td><td>RO</td><td>0y0</td><td>Always read as 0 when read.</td></tr>
<tr><td>[5]</td><td>ADSCN</td><td>R/W</td><td>0y0</td><td>Operation mode setting during channel scan
0y0: 4-ch scan
0y1: Reserved</td></tr>
<tr><td>[4:3]</td><td>–</td><td>R/W</td><td>0y00</td><td>Always write as 0.</td></tr>
<tr><td>[2:0]</td><td>ADCH[2:0]</td><td>R/W</td><td>0y000</td><td>Analog input channel select
<table>
<tr><td colspan="2">SCAN</td><td>0</td><td colspan="2">1</td></tr>
<tr><td colspan="2">ADSCN</td><td>0/1</td><td>0</td><td>1</td></tr>
<tr><td rowspan="8">ADCH [2:0]</td><td>0y000</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0y001</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0y010</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0y011</td><td>Reserved</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0y100</td><td>AN4</td><td>AN4</td><td>Reserved</td></tr>
<tr><td>0y101</td><td>AN5</td><td>AN4 to AN5</td><td>Reserved</td></tr>
<tr><td>0y110</td><td>AN6</td><td>AN4 to AN6</td><td>Reserved</td></tr>
<tr><td>0y111</td><td>AN7</td><td>AN4 to AN7</td><td>Reserved</td></tr>
</table></td></tr>
</table>

R/W : Read/Write RO : Read Only WO : Write Only

Note 1: To start AD conversion, be sure to write 1 in the ADMOD1<DACON>, and then wait for 3 μs, which is the time taken until the internal reference voltage is stabilized, and then write 1 in the ADMOD0<ADS>.

Note 2: To switch to standby mode after AD conversion end, set 0 in the ADMOD1<DACON>.

[Description]

a. <DACON>

Controls the VREF application.

0y0: OFF

0y1: ON

b. <ADSCN>

Sets the operation mode during channel scan.

0y0: 4-ch scan

0y1: Reserved

c. <ADCH[2:0]>

Selects analog input channels.

SCAN		0	1	
ADSCN		0/1	0	1
ADCH [2:0]	0y000	Reserved	Reserved	Reserved
	0y001	Reserved	Reserved	Reserved
	0y010	Reserved	Reserved	Reserved
	0y011	Reserved	Reserved	Reserved
	0y100	AN4	AN4	Reserved
	0y101	AN5	AN4 to 5	Reserved
	0y110	AN6	AN4 to 6	Reserved
	0y111	AN7	AN4 to 7	Reserved

3. ADMOD2 (AD mode control register 2)

Address = (0xF008_0000) + (0x0058)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	EOCFHP(Note)	RO	0y0	Top-priority AD conversion end flag Read: 0y0: Before conversion or being converted 0y1: End Write: Invalid
[6]	ADBFHP	RO	0y0	Top-priority AD conversion BUSY flag Read: 0y0: Conversion stop 0y1: Being converted Write: Invalid
[5]	HPADCE	R/W	0y0	Top-priority AD conversion start 0y0: Don't care 0y1: Conversion start. Always read as 0 when read.
[4:3]	–	R/W	0y00	Always write as 0.
[2:0]	HPADCH[2:0]	R/W	0y000	Analog input channel select during top-priority conversion 0y100: AN4, 0y110: AN6 0y101: AN5, 0y111: AN7, other: Reserved

R/W : Read/Write RO : Read Only WO : Write Only

Note: As read this register, <EOCFHP> is clear to 0.

[Description]

a. <EOCFHP>

Used to set the top-priority AD conversion end flag.

0y0: Before conversion or being converted

0y1: End

b. <ADBFHP>

Used to set the top-priority AD conversion BUSY flag.

0y0: Conversion stop

0y1: Being converted

c. <HPADCE>

Controls the start of top-priority AD conversion.

0y0: Don't care

0y1: Conversion start

Note: Always read as 0 when read.

d. <HPADCH[2:0]>

Analog input channel select during top-priority conversion

0y100: AN4 0y110: AN6

0y101: AN5 0y111: AN7

Other: Reserved

4. ADMOD3 (AD mode control register 3)

Address = (0xF008_0000) + (0x005C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	–	R/W	0y0	Always write as 0.
[6]	–	RO	0y0	Always read as 0 when read.
[5]	ADOBIC	R/W	0y0	AD monitoring function interrupt setting 0y0: Smaller than the comparison register settings 0y1: Greater than the comparison register settings
[4:1]	REGS[3:0]	R/W	0y0000	Bit to select the AD conversion result storage register to be compared with the settings of the comparison register during when the AD monitoring function is enabled REGS[3:0] 0y0000: Reserved 0y0001: Reserved 0y0010: Reserved 0y0011: Reserved 0y0100: ADREG4 0y0101: ADREG5 0y0110: ADREG6 0y0111: ADREG7 0y1xxx: ADREGSP
[0]	ADOBSV	R/W	0y0	AD monitoring function 0y0: Disable 0y1: Enable

R/W : Read/Write RO : Read Only WO : Write Only

[Description]

a. <ADOBIC>

Sets the AD monitoring function interrupt.

0y0: Smaller than the comparison register settings

0y1: Greater than the comparison register settings

b. <REGS[3:0]>

Selects the AD conversion result storage register to be compared with the settings of the comparison register during when the AD monitoring function is enabled.

0y0000: Reserved 0y0001: Reserved

0y0010: Reserved 0y0011: Reserved

0y0100: ADREG4 0y0101: ADREG5

0y0110: ADREG6 0y0111: ADREG7

0y1xxx: ADREGSP

c. <ADOBSV>

Controls the AD monitoring function.

0y0: Disable

0y1: Enable

5. ADMOD4 (AD mode control register 4)

Address = (0xF008_0000) + (0x0060)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	–	R/W	0y0	Always write as 0.
[6]	–	R/W	0y0	Always write as 0.
[5]	–	R/W	0y0	Always write as 0.
[4]	–	R/W	0y0	Always write as 0.
[3:2]	–	RO	0y00	Always read as 0 when read.
[1:0]	ADRST[1:0]	R/W	0y00	Resets the ADC software by the write of 0y10 → 0y01. Initializes all except the (ADCLK) register.

R/W : Read/Write RO : Read Only WO : Write Only

[Description]

a. <ADRST[1:0]>

Resets the ADC software by the write of 0y10 → 0y01. Initializes all except the ADCLK register.

- AD conversion result register.

  A/D conversion results are stored in the four registers of A/D conversion result higher-order/lower-order registers ADREG4H/L to ADREG7H/L. Top-priority conversion results are stored in ADREGSPH/L. Since these registers are structured the same, ADREG4 that is the conversion result storage register for the 4 channel is shown below:

1. ADREG4L (AD conversion result lower-order register 4

Address = (0xF008_0000) + (0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7]	ADR41	RO	0y0	AD conversion result lower-order bit 1
[6]	ADR40	RO	0y0	AD conversion result lower-order bit 0
[5:2]	–	RO	0y0000	Always read as 0 when read.
[1]	OVR4	RO	0y0	Overrun flag 0y0: No overrun occurred 0y1: Overrun occurred
[0]	ADR4RF	RO	0y0	AD conversion result storage flag 0y0: No change result 0y1: With conversion result

[Description]

a. <ADR4[1:0]>

They are AD conversion result lower-order bits 1 to 0.

b. <OVR4>

Used for the overrun flag.

0y0: No overrun occurred

0y1: Overrun occurred

c. <ADR4RF>

Used for the AD conversion result storage flag.

0y0: No change result

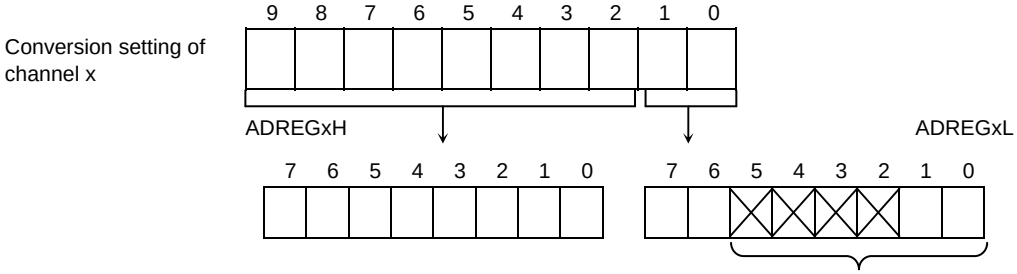
0y1: With conversion result

Note : As ADREG4L to ADREG7L and ADREGSPL Registers are the same composition. Explain Register ADREG4L only, the other Registers are same as ADREG4L.
About the address of ADREG4L to ADREG7L and ADREGSPL Register, please check Register Map.

2. ADREG4H (AD conversion result higher-order register 4))

Address = (0xF008_0000) + (0x0024)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7]	ADR49	RO	0y0	AD conversion result higher-order bit 9
[6]	ADR48	RO	0y0	AD conversion result higher-order bit 8
[5]	ADR47	RO	0y0	AD conversion result higher-order bit 7
[4]	ADR46	RO	0y0	AD conversion result higher-order bit 6
[3]	ADR45	RO	0y0	AD conversion result higher-order bit 5
[2]	ADR44	RO	0y0	AD conversion result higher-order bit 4
[1]	ADR43	RO	0y0	AD conversion result higher-order bit 3
[0]	ADR42	RO	0y0	AD conversion result higher-order bit 2



- Always read as 0 when bits 5 to 2 are read.
- Bit 0 is the AD conversion result storage flag <ADRxRF>. Set to 1 when AD conversion settings are stored. Cleared to 0 when the lower-order register (ADREGxL) is read.
- Bit 1 is the overrun flag <OVRx>. Set to 1 when conversion results are overwritten before reading the both conversion result storage registers (ADREGxH, ADREGxL). Cleared to 0 by flag read.

[Description]

a. <ADR4[9: 2]>

They are AD conversion result higher-order bits 9 to 2.

Note : As ADREG4H to ADREG7H and ADREGSPH Registers are the same composition. Explain Register ADREG4H only, the other Registers are same as ADREG4H.
About the address of ADREG4H to ADREG7H and ADREGSPH Register, please check Register Map.

- AD conversion result comparison register

1. ADCOMREGL (A/D conversion result comparison lower-order register)

Address = (0xF008_0000) + (0x0048)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:6]	ADRCOM[1:0]	R/W	0y00	AD conversion result comparison lower-order bit 1 to 0
[5:0]	–	RO	0y00000	Always read as 0 when read.

[Description]

a. <ADRCOM[1:0]>

They are AD conversion result comparison lower-order bits 1 to 0.

2. ADCOMREGH (A/D conversion result comparison higher-order register)

Address = (0xF008_0000) + (0x004C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	ADRCOM9	R/W	0y0	AD conversion result comparison higher-order bit 9
[6]	ADRCOM8	R/W	0y0	AD conversion result comparison higher-order bit 8
[5]	ADRCOM7	R/W	0y0	AD conversion result comparison higher-order bit 7
[4]	ADRCOM6	R/W	0y0	AD conversion result comparison higher-order bit 6
[3]	ADRCOM5	R/W	0y0	AD conversion result comparison higher-order bit 5
[2]	ADRCOM4	R/W	0y0	AD conversion result comparison higher-order bit 4
[1]	ADRCOM3	R/W	0y0	AD conversion result comparison higher-order bit 3
[0]	ADRCOM2	R/W	0y0	AD conversion result comparison higher-order bit 2

Note: When setting or changing values in this register, keep the AD monitoring function disabled (ADMOD3<ADOBSV> = 0).

[Description]

a. <ADRCOM[9:2]>

They are AD conversion result comparison higher-order bits 9 to 2.

- AD Conversion Clock Setting Register

1. ADCLK (AD conversion clock setting register)

Address = (0xF008_0000) + (0x0070)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7]	–	R/W	0y1	Always write as 1.
[6:4]	–	R/W	0y000	Always write as 0.
[3]	–	RO	0y0	Always read as 0 when read.
[2:0]	ADCLK[2:0]	R/W	0y000	AD prescaler output select AD conversion 1-clock period = 0y000: PCLK 0y001: PCLK/2 0y010: PCLK/4 0y011: PCLK/8 0y1xx: PCLK/16

Note 1: While AD conversion is executed with a clock selected in the register above, at this time, a conversion clock needs to be selected so that the AD conversion clock can be 33 MHz or less in order to meet the guaranteed accuracy.

Note 2: During AD conversion, do not switch the conversion clock.

[Description]

a. <ADCLK[2:0]>

Selects the AD prescaler output.

AD conversion 1-clock period =

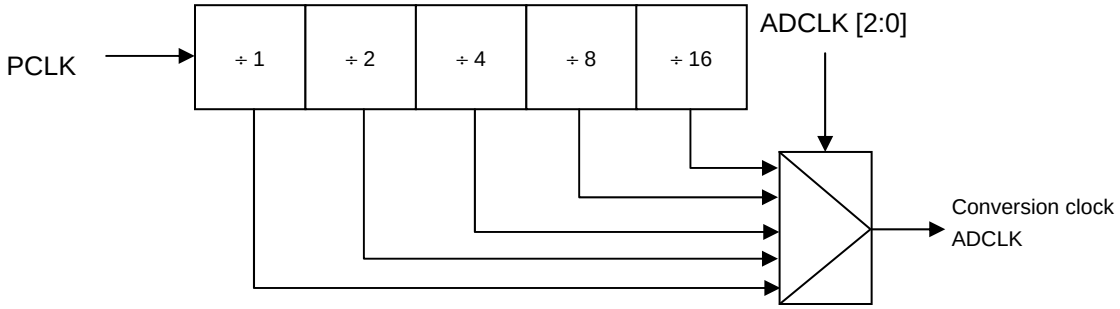
0y000: PCLK

0y001: PCLK/2

0y010: PCLK/4

0y011: PCLK/8

0y1xx: PCLK/16



PCLK	<ADCLK2:0>	ADCLK	AD conversion speed
100MHz	0y00x	–	Setting disabled
	0y010 (PCLK/4)	25MHz	1.84 μsec
96MHz	0y00x	–	Setting disabled
	0y010 (PCLK/4)	24MHz	1.92 μsec

AD conversion speed can be determined with the following formula.

Conversion speed = 46 × (1/ADCLK)

Note: In the period of AD conversion, Don't change the clock of ADCLK .

- Interrupt Register

1. ADIE (A/D interrupt enable register)

Address = (0xF008_0000) + (0x0074)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:3]	–	RO	0y00000	Always read as 0 when read.
[2]	MIE	R/W	0y0	AD monitoring interrupt enable 0y0: Disable 0y1: Enable
[1]	HPIE	R/W	0y0	Top-priority AD conversion interrupt enable 0y0: Disable 0y1: Enable
[0]	NIE	R/W	0y0	Normal AD conversion interrupt enable 0y0: Disable 0y1: Enable

Note: Please set neither Top-priority AD conversion interrupt nor Normal AD conversion Interrupt at the same time.

[Description]

a. <MIE>
Controls the AD monitoring interrupt.
0y0: Disable
0y1: Enable

b. <HPIE>
Controls the top-priority AD conversion interrupt.
0y0: Disable
0y1: Enable

c. <NIE>
Controls the normal AD conversion interrupt.
0y0: Disable
0y1: Enable

2. ADIS (AD interrupt status register)

Address = (0xF008_0000) + (0x0078)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined.
[7:3]	–	RO	0y00000	Always read as 0 when read.
[2]	MIS	RO	0y0	Status of before masking an AD monitoring interrupt 0y0: No 0y1: Interrupt occurred
[1]	HPIS	RO	0y0	Status of before masking a top-priority AD conversion interrupt 0y0: No 0y1: Interrupt occurred
[0]	NIS	RO	0y0	Status of before masking a normal AD conversion interrupt 0y0: No 0y1: Interrupt occurred

[Description]

a. <MIS>

They are the status of before masking an AD monitoring interrupt.

0y0: No

0y1: Interrupt occurred

b. <HPIS>

They are the status of before masking a top-priority AD conversion interrupt.

0y0: No

0y1: Interrupt occurred

c. <NIS>

They are the status of before masking a normal AD conversion interrupt.

0y0: No

0y1: Interrupt occurred

3. ADIC (AD interrupt clear register)

Address = (0xF008_0000) + (0x007C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:3]	–	RO	0y00000	Always read as 0 when read.
[2]	MIC	WO	0y0	AD monitoring interrupt clear 0y0: – 0y1: Clear
[1]	HPIC	WO	0y0	Top-priority AD conversion interrupt clear 0y0: – 0y1: Clear
[0]	NIC	WO	0y0	Normal AD conversion interrupt clear 0y0: – 0y1: Clear

[Description]

a. <MIC>

Controls the AD monitoring interrupt.

0y0: –

0y1: Clear

b. <HPIC>

Controls the top-priority AD conversion interrupt.

0y0: –

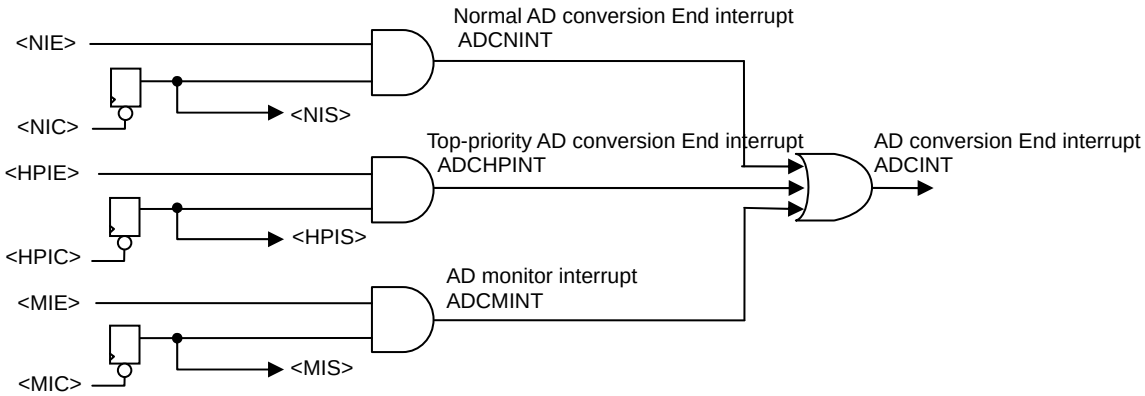
0y1: Clear

c. <NIC>

Controls the normal AD conversion interrupt.

0y0: –

0y1: Clear



3.22.2 Description of Operation

3.22.2.1 Analog Reference Voltage

Apply the analog reference voltage's “H” level side to the VREFH pin and the “L” level side to the VREFL pin. Writing 0 in the ADMOD1<DACON> can turn OFF the switch for VREFH - VREFL. To start AD conversion, be sure to write 1 in the DACON, and then wait for 3 μs, which is the time taken until the internal reference voltage is stabilized, and then write 1 in the ADMOD0<ADS>.

3.22.2.2 Selecting Analog Input Channels

Selecting an analog input channel depends on the operation mode of the AD converter.

(1) For normal AD conversion

When using an analog input channel in fix mode, select one channel from the AN4 to AN7 pins by setting (ADMOD0<SCAN> = 0) ADMOD1<ADCH[2:0]>.

When using an analog input channel in scan mode, select one scan mode from the four scan modes by setting (ADMOD0<SCAN> = 1) ADMOD1 <ADCH[2:0]>.

(2) For top-priority AD conversion

Select one channel from the analog input pins AN4 to AN7 by setting ADMOD2<HPADCH[2:0]>.

After reset, ADMOD0<SCAN> is initialized to 0 and ADMOD1<ADCH[2:0]> to 0y000. Since these settings are used for channel selection, the channel fixed input with the AN0 pin will be selected. Pins not used as analog input channels can be used as normal ports.

3.22.2.3 AD Conversion Start

The AD conversion has the two types of normal AD conversion and top-priority AD conversion.

Normal AD conversion can be started up by setting ADMOD0<ADS> to 1. Top-priority AD conversion can be started up by software by setting ADMOD2<HPADCE> to 1.

For normal AD conversion, one operation mode is selected from the four types of operation modes specified by ADMOD0<REPEAT, SCAN>. The operation mode for top-priority AD conversion is only single conversion by channel fix mode.

When normal AD conversion is started, the AD conversion BUSY flag (ADMOD0<ADBFN>) that shows the state for AD being converted is set to 1.

When top-priority AD conversion is started, the AD conversion BUSY flag (ADMOD2<ADBFHP>) that shows the state for AD being converted is set to 1.

In addition, when top-priority conversion is started during normal AD conversion, ADMOD0<ADBFN> is kept to 1.

<EOCFHP> and <EOCFN> are set to 1 after conversion is completed. This flag is cleared to 0 only when read.

Two type AD conversion can be used. During Normal AD conversion is executing, Top-priority AD conversion can be carried out first.

When ADMOD2<HPADCE> is set to 1 during normal AD conversion, top-priority AD conversion's startup, normal AD conversion being converted currently is cancelled immediately. Then, top-priority AD conversion is started, starting the AD conversion (channel fix single conversion) for the channel specified by ADMOD2<HPADCH[2:0]>. When this result is stored into ADREGSPH/L, normal AD conversion is restarted from the cancelled channel.

But during top-priority AD conversion, can't set top-conversion AD conversion again.

If top-priority AD conversion need to be set again, have to check that top-priority AD conversion being converted currently is end (AD conversion End flag: ADMOD<ADBFHP>). Then top-priority conversion AD conversion can be started.

3.22.2.4 AD Conversion Modes and AD Conversion End Interrupt

For AD conversion, the following four operation modes are provided: For normal AD conversion, selection is available by setting ADMOD0<REPEAT and SCAN>. As for top-priority AD conversion, only single conversion mode by channel fix mode is available.

a. Channel-fix single conversion mode

b. Channel-scan single conversion mode

c. Channel-fix repeat conversion mode

d. Channel-scan repeat conversion mode

(1) Normal AD conversion

To select operation modes, use ADMOD0<REPEAT, SCAN>. After AD conversion is started, ADMOD0<ADBFN> is set to 1. When a specified AD conversion ends, the Normal AD conversion end interrupt is generated, ADMOD0<EOCFN> is set 1 , shows the end of the AD conversion sequence.

a. Channel-fix single conversion mode

Setting ADMOD0 <REPEAT, SCAN> to 0y00 selects the channel-fix single conversion mode.

This mode performs a conversion only one time at one channel selected. After conversion ends, ADMOD0<EOCFN> is set to 1, generating Normal AD conversion End interrupt request. <EOCFN> is cleared to 0 only by being read.

b. Channel-scan single conversion mode

Setting ADMOD0 <REPEAT, SCAN> to 0y01 selects the channel-scan single conversion mode.

This mode performs a conversion only one time at each scan channel selected. After scan conversion ends, ADMOD0<EOCFN> is set to 1, generating Normal AD conversion End interrupt request. <EOCFN> is cleared to 0 only by being read.

c. Channel-fix repeat conversion mode

Setting ADMOD0<REPEAT, SCAN> to 0y10 selects the channel-fix repeat conversion mode.

This mode performs a conversion at one channel selected repeatedly. After conversion ends, ADMOD0<EOCFN> is set to 1. The timing of Normal AD conversion End interrupt request generation can be selected by setting ADMOD0 <ITM>. The timing of <EOCFN> being set is also linked to the interrupt timing.

ADMOD0<EOCFN> is cleared to 0 only by being read.

Setting <ITM> to 0 generates an interrupt request each time an AD conversion ends. In this case, conversion results are always stored into the storage register of ADREGxH/L. At the point of storage, <EOCFN> is set to 1.

d. Channel-scan repeat conversion mode

Setting ADMOD0 <REPEAT, SCAN> to 0y11 selects the channel-scan repeat conversion mode.

This mode performs a conversion at selected scan channels repeatedly. Each time after the conversion at a final channel ends, ADMOD0<EOCFN> is set to 1, generating Normal AD conversion End interrupt request. <EOCFN> is cleared to 0 only by being read.

To stop the repeat conversion mode (mode of c and d) operation, write 0 in ADMOD1 <REPEAT>. At the point when a scan conversion being executed ends, the repeat conversion mode ends.

(2) Top-priority AD conversion

The operation mode is only single conversion by channel fix mode. The settings in ADMOD0<REPEAT, SCAN> are not involved.

When startup conditions are established, a conversion at a channel specified by ADMOD2<HPADCH[2:0]> is performed only one time. When conversion ends, the top-priority AD conversion end interrupt is generated, which sets 1 in ADMOD2<EOCFHP>. The EOCFHP flag is cleared to 0 only by being read.

Table 3.22.1 Relationship between AD conversion mode, interrupt generation timing, and flag operation

Conversion mode	Interrupt generation timing	EOCFN set timing (Note)	ADMOD0		
			ITM	REPEAT	SCAN
Channel fix Single conversion	After conversion end	After conversion end	–	0	0
Channel fix Repeat conversion	Per one conversion	Each time after one conversion ends	0	1	0
Channel scan Single conversion	After scan conversion end	After scan conversion end	–	0	1
Channel scan Repeat conversion	Each time after one scan conversion ends	Each time after one scan conversion ends	–	1	1

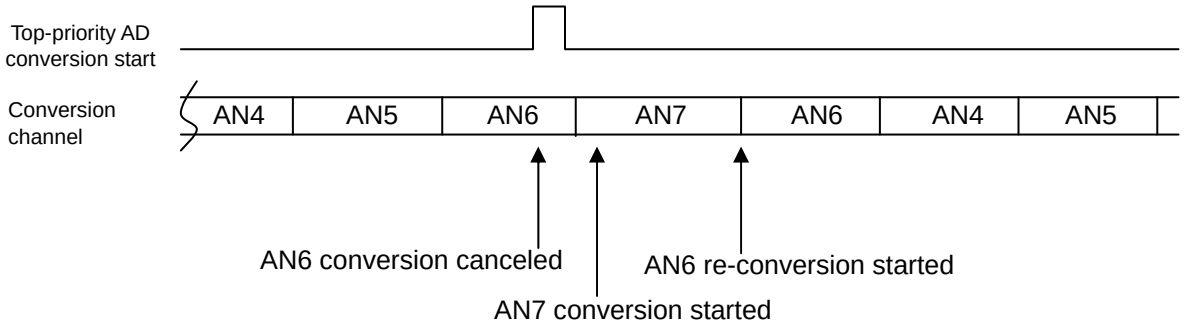
Note: EOCFN is cleared to 0 only by being read.

3.22.2.5 Top-Priority Conversion Mode

Top-priority AD conversion can be performed by interrupting into normal AD conversion. Top-priority AD conversion can be started up by software while setting ADMOD2<HPADCE> to 1

When top-priority AD conversion is started up during normal AD conversion, the AD conversion being converted currently is cancelled immediately to execute the single conversion at a channel specified by ADMOD2<HPADCH[2:0]>. The conversion result is stored into ADREGSPH/L, generating a top-priority AD conversion interrupt. After that, conversion is restarted from the channel where normal AD conversion was cancelled. Note that a top-priority AD conversion started up during another top-priority AD conversion is ignored.

Example: When AN7 top-priority AD conversion is started up with ADMOD2 <HPADCH[2:0]> = 0y111 during repeat scan conversion at channels AN4 to AN6 with ADMOD0 <REPEAT,SCAN> = 0y11 and ADMOD1<ADCH[2:0]> = 0y0110



3.22.2.6 AD Monitoring Function

Setting ADMOD3<ADOBSV> to 1 enables the AD monitoring function.

The value of Result storage register that is appointed by ADMOD3<REGS [3:0]> is compared with the value of AD conversion result register (H/L), ADMOD3<ADOIBC> can select greater or smaller of comparison format. As register ADIE<MIE> is Enable, This comparison operation is performed each time when a result is stored in the corresponding conversion result storage register. When conditions are met, the interrupt is generated. Be careful that the storage registers assigned for the AD monitoring function are usually not ready by software, which means that the overrun flag <OVRx> is always set and the conversion result storage flag <ADRxRF> is also set.

3.22.2.7 AD Conversion Time

One AD conversion takes 46 clocks including sampling clocks. The AD conversion clock is selected from 1/1, 1/2, 1/4, 1/8 PCLK by <ADCLK[2:0]>. To meet the guaranteed accuracy, the AD conversion clock needs to be set from 0.625MHz to 33 MHz, or equivalently from 1.39μs to 73.6μs of AD conversion time.

3.22.2.8 Storage and Read of AD Conversion Results

A/D conversion results are stored in the A/D conversion result higher-order/lower-order registers (ADREG4H/L to ADRG7H/L) for the normal AD conversion (ADREG4H/L to ADREG7H/L are read-only registers)

AD conversion results of channels AN4, AN5, AN6 and AN7 are each stored into ADREG4H/L, ADREG5H/L, ADREG6H/L, and ADREG7H/L.

Table 3.22.2 shows the correspondence between analog input channels and AD conversion result registers.

Table 3.22.2 Correspondence between analog input channels and AD conversion result registers

Analog input channel (Port D)	AD conversion result register
AN4	ADREG4H/L
AN5	ADREG5H/L
AN6	ADREG6H/L
AN7	ADREG7H/L

Note: In order to detect overruns without omission, read the conversion result storage register's higher-order bits first, and then read the lower-order bits next. As this result, receiving the result of OVRn = 0 and ADRnRF = 1 for overruns existing in the lower-order bits means that a correct conversion result has been obtained.

3.22.2.9 Data Polling

To process AD conversion results by using data polling without using interrupts, perform a polling on ADMOD0<EOCFN>. After confirming that ADMOD0<EOCFN> is set to 1, read the AD conversion storage register.

3.23 Watchdog Timer (WDT) (Runaway Detection Timer)

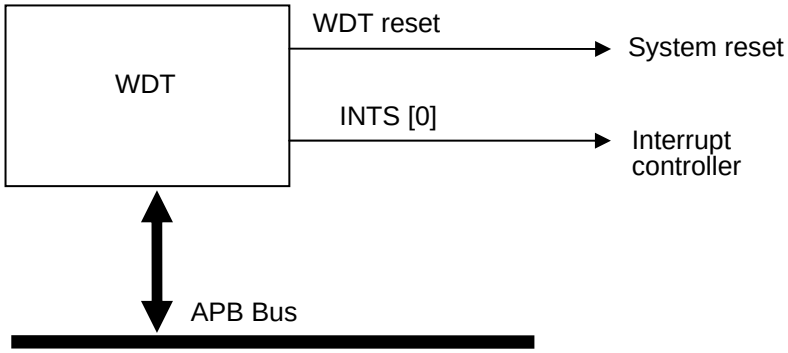
The TMPA901CM contains a watchdog timer (WDT) for runaway detection.

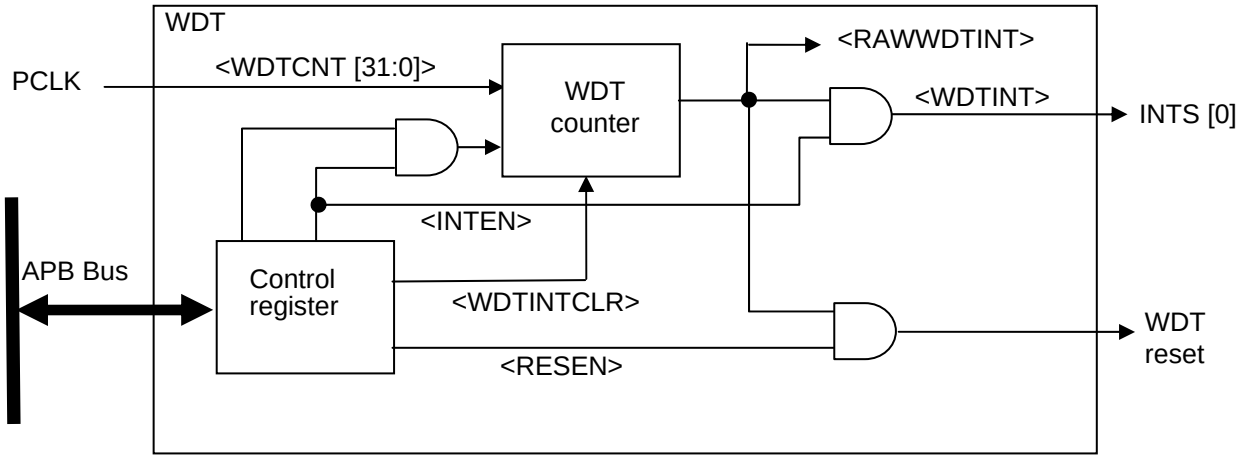
The watchdog timer is provided for detecting a CPU malfunction (runaway) due to causes such as noise and for restoring the CPU to a normal state. When the watchdog timer detects a runaway condition, it generates an interrupt to notify the interrupt controller and CPU of this condition. (Interrupt source signal to Interrupt controller: INTS [0])

By connecting the watchdog timer output to the internal reset pin, a reset can be forcefully generated.

Note: Please set to disable the WDT in Debug operation.

3.23.1 Block diagram





3.23.2 Register Functions

The following lists the SFRs:

Base address = 0xF001_0000

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 = (0xF001_0000) + (0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	WDTCNT	R/W	0xFFFFFFFF	WDT counter setting value 0x00000001 to 0xFFFFFFFF

[Description]

a. <WDTCNT>

Specifies the value to be set to the WDT 32 bit counter (The clock of WDT counter is PCLK).

After WdogControl <INTEN> is enabled, the value set in WdogLoad<WDTCNT> is loaded into the internal decrement counter. The value of counter can be set from 0x00000001 to 0xFFFFFFFF. (0 can't be set)

When reading this register, the setting value is read out.

2. WdogValue (The current value for the watchdog counter)

Address = (0xF001_0000) + (0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	CWDTCNT	RO	0xFFFFFFFF	Current value of the WDT counter

[Description]

a. <CWDTCNT>

This bit can be read the current value of watch dog counter.

3. WdogControl (Watchdog control register)

Address = (0xF001_0000) + (0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as 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

[Description]

a. <RESEN>

Controls the WDT reset output. Time until releasing reset is after PCLK 5 clocks.

b. <INTEN>

0y1: Enables the WDT counter and the WDT interrupt. When this bit is set to 1, the value set in the WdogLoad register is loaded into the WDT counter and the counter starts decrementing.

4. WdogIntClr (Clears the watchdog interrupt)

Address = (0xF001_0000) + (0x000C)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	WDTINTCLR	WO	Undefined	WDT interrupt clear (Writing any value clears the interrupt.)

[Description]

a. <WDTINTCLR>

Writing any value to this register clears the WDT interrupt and loads the value set in the WdogLoad register into the WDT counter.

5. WdogRIS (Watchdog raw interrupt status)

Address = (0xF001_0000) + (0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined.
[0]	RAWWDTINT	RO	0y0	WDT interrupt raw status 0y0: No interrupt 0y1: Interrupt requested

[Description]

a. <RAWWDTINT>

Indicates the raw status of the WDT interrupt. The <RAWWDTINT> value is ANDed with the interrupt enable signal (WdogControl<INTEN>) to generate an interrupt (WdogMIS<WDTINT>).

6. WdogMIS (Watchdog masked interrupt status)

Address = (0xF001_0000) + (0x0014)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	–	–	Undefined	Read as undefined.
[0]	WDTINT	RO	0y0	WDT interrupt enabled status 0y0: No interrupt 0y1: Interrupt requested

[Description]

a. <WDTINT>

This bit is status bit of the interrupt from WDT counter. The AND value of WdogRIS <RAWWDTINT> and WdogControl <INTEN> is read out.

7. WdogLock (Watchdog Lock register)

Address = (0xF001_0000) + (0x0C00)

Bit	Bit Symbol	Type	Reset Value	Description
[31:0]	REGWEN	WO	undefined	Enables/disables writes to other WDT registers. 0x1ACCE551: Enable Others: Disable (Initial value: Enable)

Address = (0xF001_0000) + (0x0C00)

Bit	Bit Symbol	Type	Reset Value	Description
[31:1]	Reserved	–	undefined	Read as undefined.
[0]	REGWENST	RO	0y0	Indicates the enabled/disabled status of writes to other WDT registers. 0y0: Enabled (Not locked) 0y1: Disabled (Locked)

[Description]

a. <REGWEN>

Disables writes to other WDT registers to prevent the WDT registers from being inadvertently rewritten by runaway of program, etc.

Write except 0x1ACCE551: Disabled writes to WDT register except this register.

Write 0x1ACCE551: Enabled writes to WDT register except this register.

b. <REGWENST>

Indicates the enabled/disabled (not locked/locked) status of writes to other WDT registers.

3.24 PMC (Power Management Circuit)

This product contains a power management circuit that manages standby currents against current leaks from microprocessing products. The following eight systems of power supply are conceivable:

- 3.3-V power supply for A/D converters (for A/D converters: AVCC3AD & AVSS3AD)
- 3.3-V digital power supply (for general pins : DVCC3IO & DVSSCOM (Note))
- 3.3-V and 1.8-V power supplies for memory (for memory control: DVCCM & DVSSCOM)
- 3.3-V power supply for USB Device (for USB Device control: AVDD3T/C & AVSS3T/C)
- 3.3-V power supply for USB Host (for USB Host control: AVCC3H)
- 1.5-V-A internal power supply (for general circuits: DVCC1A & DVSSCOM)
- 1.5-V-B internal power supply (for RTC and PMC: DVCC1B & DVSSCOM)
- 1.5-V-C power supply for oscillators (for high frequency oscillators and PLL: DVCC1C & DVSS1C)

Note: The power of Low Frequency Oscillator (XT1,XT2) is 3.3-V digital power .

Each power supply is independent. (VSS is partially common.)

In the power-cut mode, power supplies to most part of the internal circuits are cut off externally to reduce the leak current in a standby state.

At this time, the state of each external pin can be fixed as “H output”, “L output”, “High-Z” or “input”. (See the backup register output data register list later in this section).

Of the eight power supplies, those that should be supplied in the PCM state are DVCC3IO, DVCCM, AVCC3AD, and DVCC1B.

The power supplies that should be cut off are DVCC1A, DVCC1C, AVDD3T, AVDD3C and AVCC3H. Even if these power supplies are cut off after the TMPA901CM enters the PCM state, no floating current will be generated in the TMPA901CM.

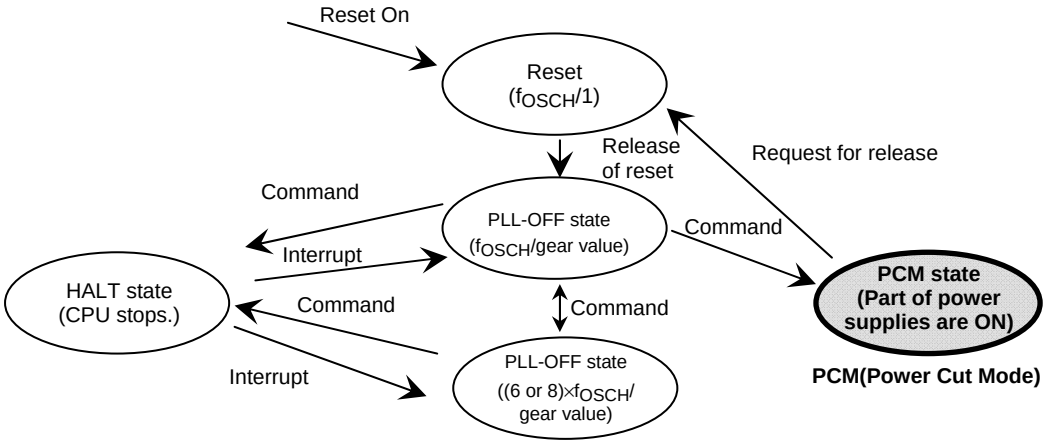


Figure 3.24.1 State Transition Diagram of TMPA901CM

3.24.1 Power Supply System

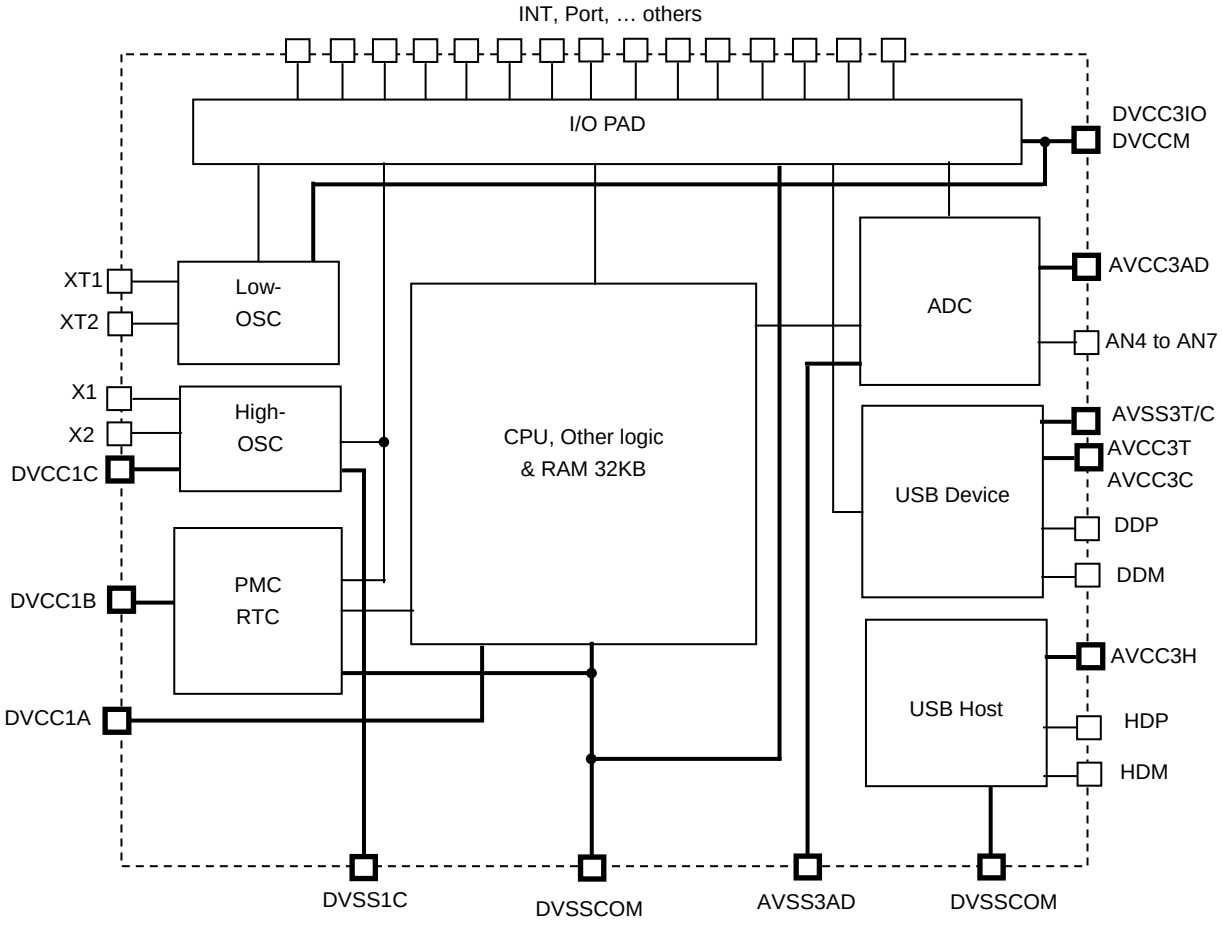


Figure 3.24.2 Power Supply System

3.24.2 Example of Connection and System Application

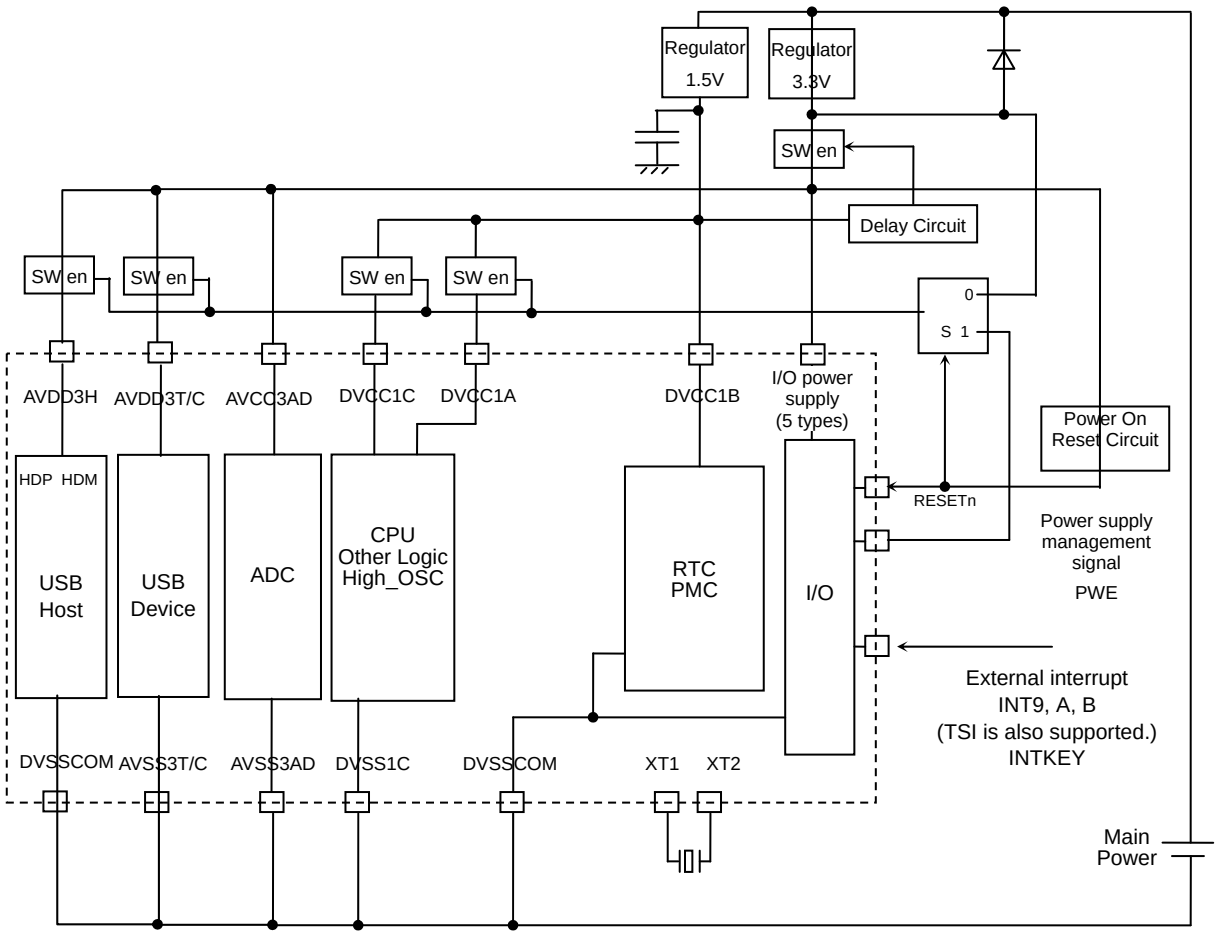


Figure 3.24.3 Example of Power Supply System

Figure 3.24.3 shows an example of the external circuit using this system.

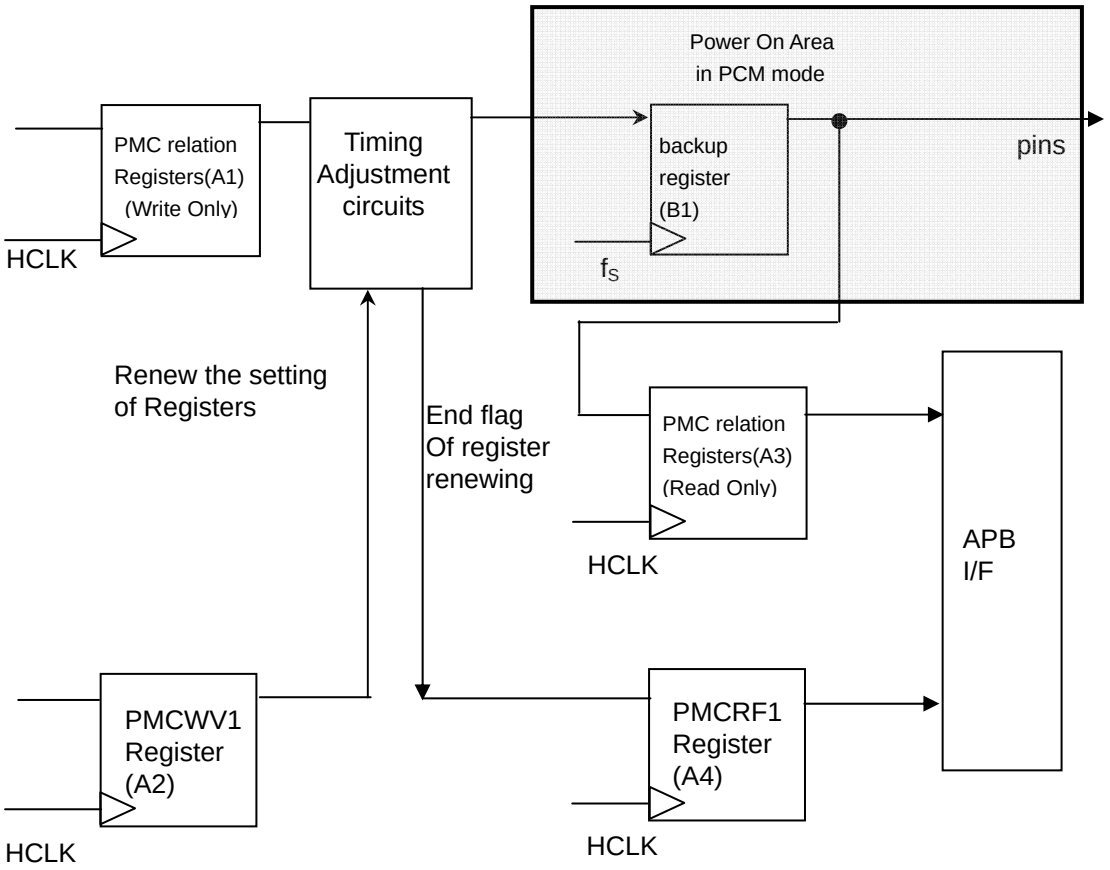
The power management pin (PWE) controls the power supply to circuits and pins. In normal states including system reset, the PWE pin outputs “High” to supply power to all blocks.

In the Power Cut Mode, the PWE pin outputs “Low” to cut off the power to most part of the internal circuits including the CPU, high-frequency oscillators, and USB power supply for reducing current consumption.

The Power Cut Mode is released by a wake-up request. Then, the PWE pin outputs “High” again to supply power to the internal circuits.

3.24.3 PMC Registers Composition

To cut the current in Power Cut Mode, PMC Registers in next Table (Address (0xF002_0200 to 0xF002_041C)) Composition is designed as below figure.



There are different PMC setting registers in Power Off area (A1, A2, A3, A4) and Power On area (B1) to realize Power Cut Mode. And have to set the PMCWV1 register to renew the backup registers setting in Power On area.

3.24.4 Description of Operation

The following shows a flowchart for entering and exiting the PCM state.

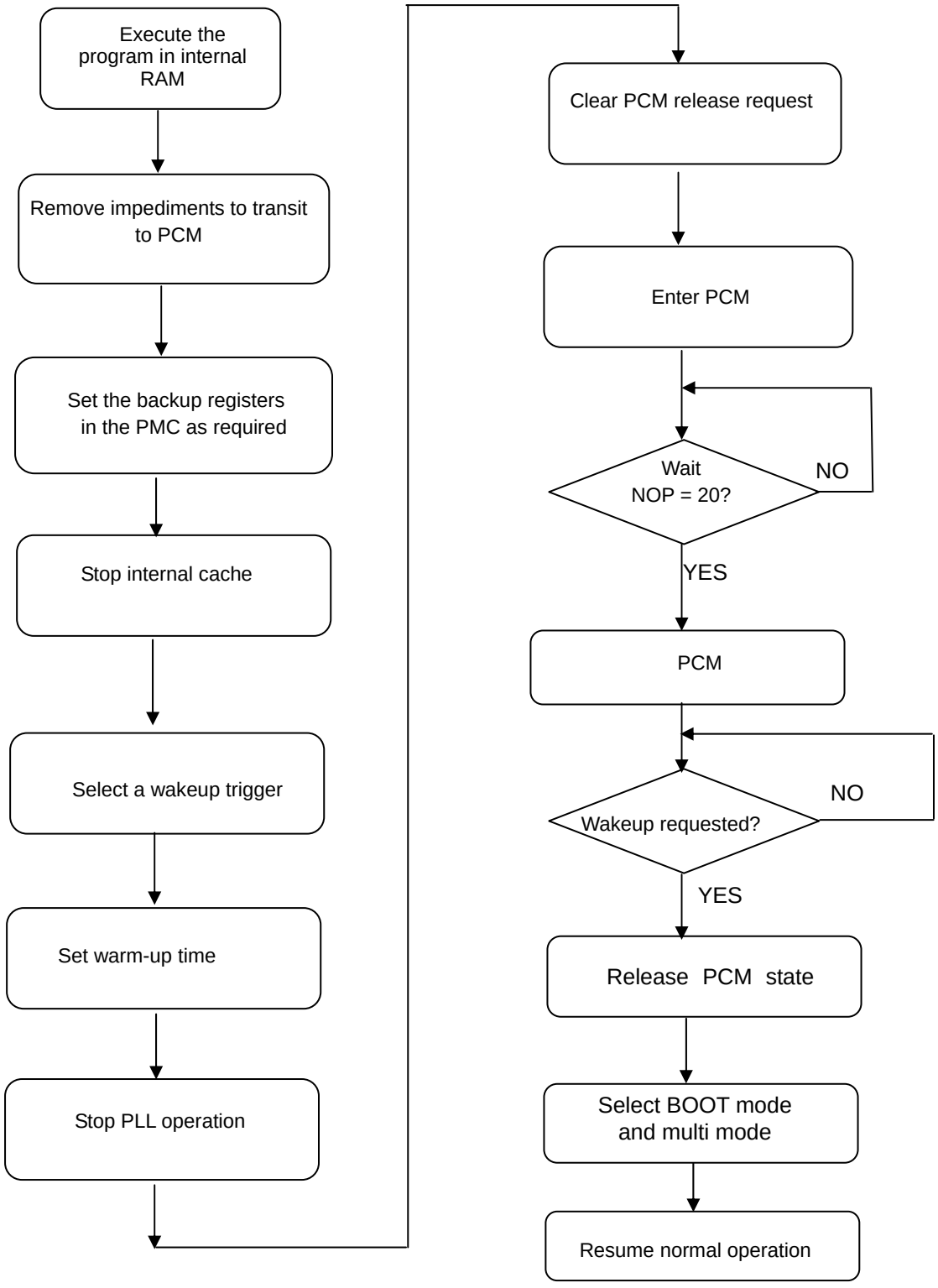


Figure 3.24.4 Flowchart for Entering and Exiting the PCM state

3.24.4.1 Entering the PCM state

In the Power Cut Mode, power supplies to the internal circuits including the CPU are cut off. To enter the Power Cut Mode, the following procedure must be observed to prepare for operation after exiting the Power Cut Mode, to define the external pin states during the Power Cut Mode, and to ensure proper mode transition.

1. Execution procedure

(1) Program execution area

For entering the PCM state, the program must be executing in the internal RAM.

(2) Remove possible impediments to transition to the PCM mode

Before entering the PCM mode, stop all functions that may interfere with the mode transition operation.

a. Disable interrupts
b. Stop the watchdog timer (The watchdog timer is initially stopped.)
c. Stop the AD converter.
d. Stop DMA operation
   - Stop the LCD controller.
   - Disable the SDRAM auto refresh function (so that the self refresh mode is enabled.)
   - Stop DMA transfer.

(3) Set the pin states

Set the backup registers to fix the state of each pin during the PCM state.

In the PCM state, the port states are controlled by the backup registers in the PMC.

PC2 should be set as the PWE pin.

Only the CKE pin (which is controlled by SDRAM) allows the pin state on exiting the PCM state to be different from the pin state after system reset. For details, refer to 3.24.4.3.

(4) Set the wakeup conditions

Set the external pin for waking up from the PCM state.

The enable register, edge selection register and wakeup interrupt flag register are provided for each external pin.

The internal RTC, an external key or an external interrupt can be selected as a wakeup trigger. (Interrupts not used can be masked.)

To use PD6 as the TSI interrupt pin (INTA), the debounce circuit must be disabled. For details, refer to the chapter on the TSI.

(5) Stop OFD operation

Stop the OFD circuit operation

(6) Stop PLL operation.

Stop the PLL circuit operation by setting high frequency clock to f_{OSCH}.

(7) Set the warm-up time: PMCCTL<WUTM1:0>

The external PWE pin changes from "0" to "1" approximately 1.5 XT1 (48μs) after the wake-up interrupt. Then, after a specified warm-up period and an additional interval of approximately 1 XT1 (32μs), the internal reset signal is released. Since power stabilization time depends on the response of the power source to be used and conditions on the system, determine the warm-up time in consideration of the period required until power is stabilized. (The warm-up time can be selected in the range of 15.625 ms to 125 ms.)

(8) Disable the internal cache memory.

(9) Clear the wake-up request signals in the PMC

Before entering the Power Cut Mode, the wake-up request signals in the PMC circuit must be cleared.

(10) Set the relevant PMC registers, and then set the corresponding bits in the PMCWV1 register to enable the newly set values.

(11) Whether or not the values in each PMC register for which the corresponding bit in the PMCWV1 register is enabled have been copied into the backup register can be checked by reading the PMCRF1 register.

(12) Enable the Power Cut Mode (PMCCTL<PCM_ON> = "1")

Note: It is not possible to set PMCCTL<PCM_ON> to "0y1" and to change the settings of other bits in the same register (<PMCPWE>, <WMTM1>, <WMTM0>) simultaneously. <PMCPWE>, <WMTM1> and <WMTM0> should be set while <PCM_ON> = "0y0". Do not change the values of these bits when writing "0y1" to <PCM_ON>.

(13) Insert dummy instructions before transition to the Power Cut Mode, approximately 7 XT1 (224μs)

Note: Programs to be started after Warm-up

Either the built-in BOOT_ROM or external memory (SMCCS0n) is selected and the program is started according to the settings of the external AM0/1 pins after Wakeup as in the case system reset is asserted.
Whether system reset starting or Wakeup from the PCM state occurred can be known by checking the flag of PMCCTL<PMC_ON> in the PMC circuit in the initial routine of the starting program.

- Wakeup from the PCM state: PMCCTL<PMC_ON> = 0y1
- System reset starting: PMCCTL<PMC_ON> = 0y0

In the startup program it is necessary to preapre a routine for checking PMCCTL<PCM_ON> and branching program execution depending on the result. 。

3.24.4.2 Wakeup from PCM status

An external interrupt is used to wake up from the Power Cut Mode.

Whether a system reset or wake-up from the Power Cut Mode occurred can be known by checking the <PCM_ON> bit in the PMCCTL register in the initial routine of the startup program.

After wake-up from the Power Cut Mode, either the internal BOOT ROM or external memory (SMCCS0n) is selected according to the external AM0 and AM1 pins to start program execution.

(It is prohibited to resume operation from a reset state while DVCC1A is cut off. Before applying a reset, make sure that DVCC1A is stably powered.) The interrupts that can be used to wake up from the Power Cut Mode are RTC, RESETn, INT9, INTB to INTH, INTA (TSI interrupt) and KI0 to KI3 interrupts. 。For details, refer to 3.26.5 “PCM Wake-Up Pins”.

When a wake-up request is accepted, the PWE pin is set to “1” and power is supplied to each block that has been placed in the Power Cut Mode.

After the warm-up time set in PMCCTL<WUTM1:0> has elapsed, HOT_RESET is automatically released.

During the Power Cut Mode, the state of each external pin remains unchanged. However, upon release of the internal HOT_RESET, all pins except the DMCCKE pin (Note) are initialized to the system reset values. 。

Whether a system reset or wake-up from the Power Cut Mode occurred can be known by checking the <PCM_ON> bit in the PMCCTL register in the initial routine of the startup program.

* PMCCTL<PCM_ON> in the PMC is not initialized at wake-up from the Power Cut Mode. After wake-up, PMCCTL<PCM_ON> should be cleared by using PMCRES<RES_PCMON>.

The interrupt that triggered wake-up from the Power Cut Mode can be checked by using the BxxRINT flags in the PMC.

Note: After a system reset, the DMCCKE pin is always initialized to the “H” level. Therefore, if SDRAM has been set to self-refresh mode before entering the Power Cut Mode, the self-refresh mode will be cleared when the DMCCKE pin is initialized to the “H” level after wake-up from the Power Cut Mode. To avoid this situation, it is possible only for the DMCCKE pin, which controls SDRAM, to specify different pin levels after a system reset and after wake-up from the Power Cut Mode. The DMCCKE pin level after wake-up from the Power Cut Mode can be set in the relevant PMC register.

	DMCCKE pin status
After system reset	“H” status
After releasing PCM status	If set to “L” status; DMCCKECTL<DMCCKEHLD> = 0y1 BSJOE<BSJOE6> = 0y1 BSJDATA<BSJDATA6> = 0y0

3.24.4.3 Status Transitions before, during and after the Power Cut Mode (PMC)

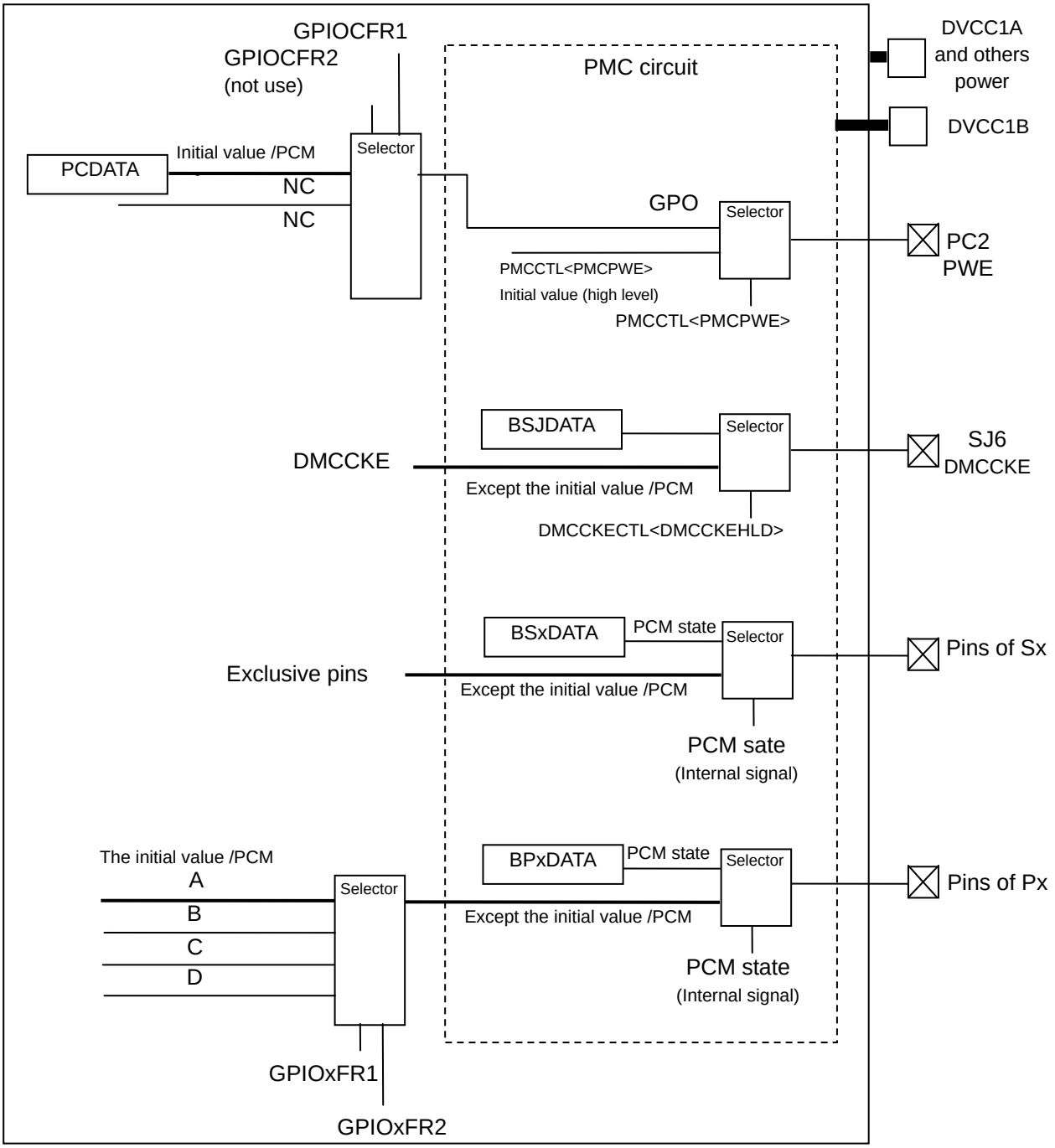
	Transition under way	PCM status Power Cut	Release under way	Normal status
	→	→	→	→
DVCC1A	Power ON	Power OFF	Power ON	Power ON
CPU, RAM, etc	RESET	Stop	RESET→Restart	Operating
Melody / Alarm circuit	RESET	Stop	RESET→Restart	Operating
DVCC1B	Power ON	Power ON	Power ON	Power ON
PMC circuit	Operating	Operating	Operating	Operating
RTC circuit	Operating	Operating	Operating	Operating
DVCC1C	Power ON	Power OFF	Power ON	Power ON
High-frequency oscillator	Operating	Stop	Restart	Operating
DVCC3IO	Power ON	Power ON	Power ON	Power ON
PWE pin	"H" Output	"L" Output	"L→H" Output	"H" Output
Low-frequency oscillator	Operating	Operating	Operating	Operating
Reset, AM pin, etc. Basic pins	Operating	Operating	Operating	Operating
JTAG pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
NANDF pin	Pin fixed (Special function)	Pin fixed (PMC)	RESET→Restart	Operating
KEYOUT pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
KEYIN pin	Input pin	Input pin	Input pin	Input pin
I^2C0 pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
INT pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
UART0/1 pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
SPIC0 pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
$I^2S0/1$ pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
LCDC pin	Pin fixed (Exclusive function)	Pin fixed (PMC)	RESET→Restart	Operating
DVCCM	Power ON	Power ON	Power ON	Power ON
Pins related to memory control	Pin fixed (Exclusive function)	Pin fixed (PMC)	RESET→Restart	Operating
DMCCKE pin	Pin fixed (Exclusive function)	Pin fixed (PMC)	Pin fixed (PMC)	Software processing→ Operating
AVCC3AD	Power ON	Power ON	Power ON	Power ON
AN4, AN5, AN7 pin	Input pin	Input pin	Input pin	Input pin
AN6 pin	Input pin	Pull-down (PMC)	RESET→Restart	Operating
TSI pin	Pin fixed (Port)	Pin fixed (PMC)	RESET→Restart	Operating
AVDD3T/C	Power ON	Power OFF	Power ON	Power ON
USB Device pin	Operating	STOP	RESET→Restart	Operating
AVCC3H	Power ON	Power OFF	Power ON	Power ON
USB Host pin	Operating	STOP	RESET→Restart	Operating

Note 1: The interrupt that triggered wake-up from the Power Cut Mode can be checked by using the BxxRINT flags in the PMC.

Note 2: After wake-up from the Power Cut Mode, PMCCTL<PCM_ON> remains “1”. To enter the Power Cut Mode again, PMCCTL<PCM_ON> should be cleared by using PMCRES<RES_PCMON>.

Note 3: In the above table, “Pin fixed (PMC)” means that the state of each external pin is controlled and fixed by the PMC even during the Power Cut Mode. Special control is required for some pins (PC2, SJ6), as shown in the circuit diagram below.

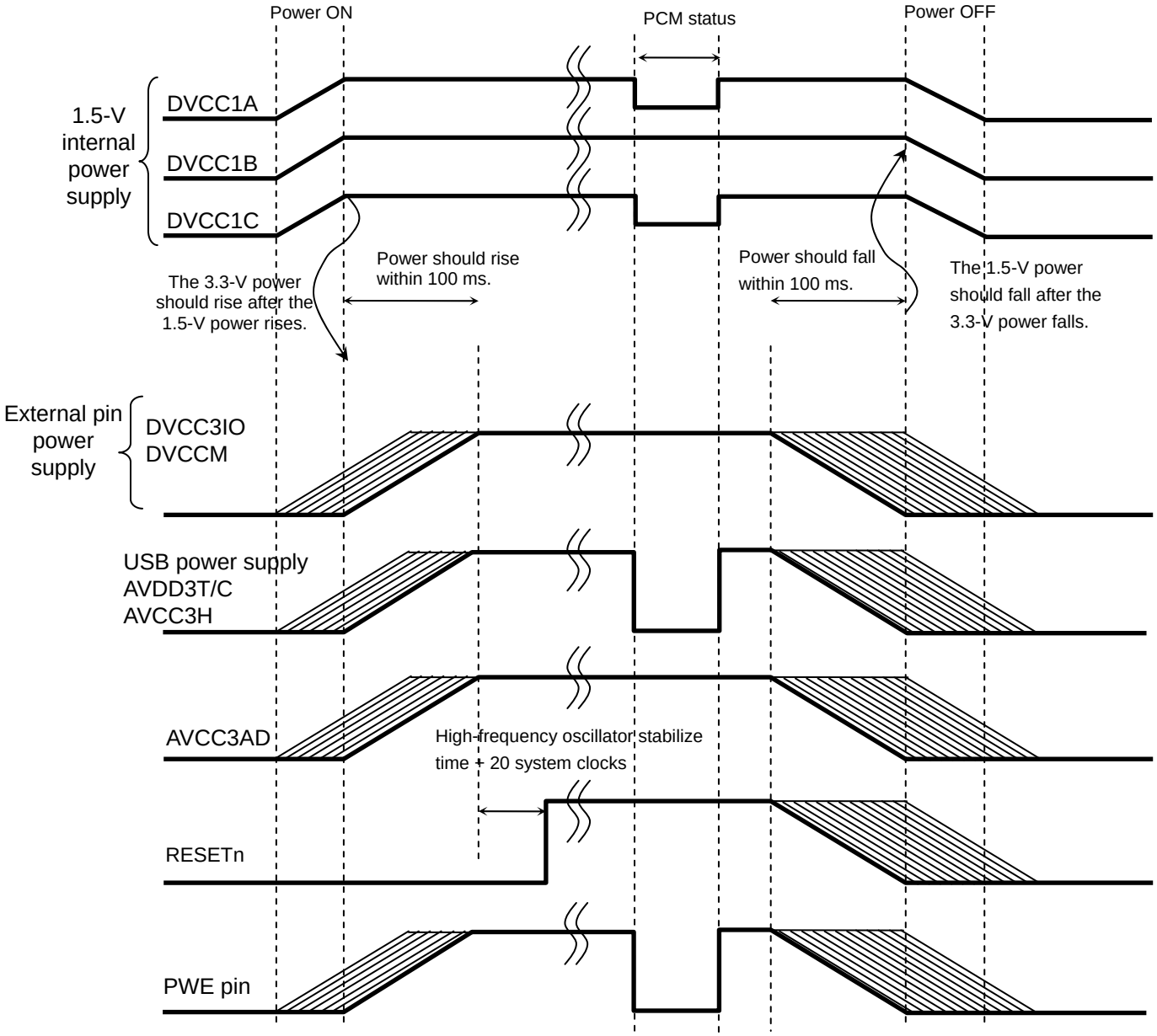
TMPA901CMXBG



3.24.5 Notes on Operation

Power ON/OFF sequence (initial power ON/complete power OFF)

For the initial power ON, the internal power should be supplied first, and for the complete power OFF, the internal power should be cut off last.



Note 1: Simultaneous rising and falling of the internal 1.5-V power and the external pin power is possible. However, the external pins may become unstable momentarily at that time. Therefore, rising and falling of the external power should be made while the internal 1.5-V power is stable as shown by the thick line in the above figure if the devices connecting to the LSIs in the surrounding parts can be affected.

Note 2: Do not allow the 3.3-V power to rise earlier than the 1.5-V power. In the same way, do not allow the 3.3-V power to fall after the 1.5-V power.

3.24.6 PCM Status Release Pins

The power-cut mode is reset on interrupt request.

The table below shows the external pins for which the power-cut mode can be released.

Note: When the pins in the table are not used, the power-cut mode can also be released on interrupt from the built-in RTC.

PORT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PA					KI3	KI2	KI1	KI0
PC	I2C0DA INT9							
PD	PY INTB	PX INTA(INTTSI)						

3.24.7 Description of Registers

The following lists the SFRs:

Table 3.24.1 Register list (1/3) Base address = 0xF002_0000

Register Name	Address (base+)	Description
BPADATA	0x0900	PortA Data Set Register when Power Cut Mode
BPBDATA	0x0904	PortB Data Set Register when Power Cut Mode
BPCDATA	0x0908	PortC Data Set Register when Power Cut Mode
BPDDATA	0x090C	PortD Data Set Register when Power Cut Mode
–	0x0914	Reserved
–	0x0918	Reserved
–	0x0924	Reserved
–	0x0928	Reserved
–	0x092C	Reserved
–	0x0930	Reserved
BPNDATA	0x0934	PortN Data Set Register when Power Cut Mode
–	0x0944	Reserved
BPTDATA	0x094C	PortT Data Set Register when Power Cut Mode
BPUDATA	0x0950	PortU Data Set Register when Power Cut Mode
BPVDATA	0x0954	PortV Data Set Register when Power Cut Mode
–	0x0B80	Reserved
BPBOE	0x0B84	PortB Data Out Enable Control when Power Cut Mode
BPCOE	0x0B88	PortC Data Out Enable Control when Power Cut Mode
BPDOE	0x0B8C	PortD Data Out Enable Control when Power Cut Mode
–	0x0B90	Reserved
–	0x0B94	Reserved
–	0x0B98	Reserved
–	0x0BA4	Reserved
–	0x0BA8	Reserved
–	0x0BAC	Reserved
–	0x0BB0	Reserved
BPNOE	0x0BB4	PortN Data Out Enable Control when Power Cut Mode
–	0x0BC4	Reserved
BPTOE	0x0BCC	PortT Data Out Enable Control when Power Cut Mode
BPUOE	0x0BD0	PortU Data Out Enable Control when Power Cut Mode
BPVOE	0x0BD4	PortV Data Out Enable Control when Power Cut Mode

Table 3.24.2 Register list (2/3) Base address = 0xF002_0000

Register Name	Address (base+)	Description
BSADATA	0x0800	SA Data Set Register when Power Cut Mode
BSBDATA	0x0804	SB Data Set Register when Power Cut Mode
–	0x0808	Reserved
–	0x080C	Reserved
BSEDATA	0x0810	SE Data Set Register when Power Cut Mode
BSFDATA	0x0814	SF Data Set Register when Power Cut Mode
BSGDATA	0x0818	SG Data Set Register when Power Cut Mode
BSHDATA	0x081C	SH Data Set Register when Power Cut Mode
BSJDATA	0x0824	SJ Data Set Register when Power Cut Mode
BSKDATA	0x0828	SK Data Set Register when Power Cut Mode
BSLDATA	0x082C	SL Data Set Register when Power Cut Mode
–	0x084C	Reserved
–	0x0850	Reserved
BSAOE	0x0A80	SA Data Out Enable Control when Power Cut Mode
BSBOE	0x0A84	SB Data Out Enable Control when Power Cut Mode
–	0x0A88	Reserved
–	0x0A8C	Reserved
BSEOE	0x0A90	SE Data Out Enable Control when Power Cut Mode
BSFOE	0x0A94	SF Data Out Enable Control when Power Cut Mode
BSGOE	0x0A98	SG Data Out Enable Control when Power Cut Mode
BSHOE	0x0A9C	SH Data Out Enable Control when Power Cut Mode
BSJOE	0x0AA4	SJ Data Out Enable Control when Power Cut Mode
BSKOE	0x0AA8	SK Data Out Enable Control when Power Cut Mode
BSLOE	0x0AAC	SL Data Out Enable Control when Power Cut Mode
–	0x0ACC	Reserved
–	0x0AD0	Reserved
BPAIE	0x0D80	PORT A WakeUp Input Enable
BPCIE	0x0D88	PORT C WakeUp Input Enable
BPDIE	0x0D8C	PORT D WakeUp Input Enable
–	0x0D94	Reserved
–	0x0DB4	Reserved
–	0x0DC4	Reserved

Table 3.24.3 Register list (3/3) Base address = 0xF002_0000

Register Name	Address (base+)	Description
BPARELE	0x0200	PortA Enable Register of Wake-up trigger from Power Cut Mode
BPDRELE	0x0204	PortD Enable Register of Wake-up trigger from Power Cut Mode
BRTRELE	0x0208	RTC Request Enable Register of Wake-up trigger from Power Cut Mode
BPXRELE	0x020C	Others Port Enable Register of Wake-up trigger from Power Cut Mode
BPAEDGE	0x0220	PortA Selection Register of Wake-up trigger Edge from Power Cut Mode
BPDEDGE	0x0224	PortD Selection Register of Wake-up trigger Edge from Power Cut Mode
BPXEDGE	0x022C	Others Ports Selection Register of Wake-up trigger Edge from Power Cut Mode
BPARINT	0x0240	PortA Wake-up Interrupt status Register
BPDRINT	0x0244	PortD Wake-up Interrupt status Register
BRTRINT	0x0248	RTC Wake-up Interrupt status Register
BPXRINT	0x024C	Others Ports Wale-up Interrupt status Register
PMCDRV	0x0260	External Port Driverbility control Register
DMCCKECTL	0x0280	DMCCKE pin setting Register (PCM mode)
PMCCTL	0x0300	Power Management Circuit Control Register
PMCWV1	0x0400	Renew Control for PMC Registers
-	0x0408	Reserved
PMCRES	0x041C	<PCM_ON> Flag Clear Register for PMCCTL_R<PMC_ON>

Table 3.24.4 backup register output data register list 1

register Address	correspond to port	register name	type	bit7	bit6	bit5	bit4	bit3	bit2	bit1	0bit
0xF002_0900	PA	BPADATA Note1)	W	–	–	–	–	BPADATA3	BPADATA2	BPADATA1	BPADATA0
0xF002_0904	PB	BPBDATA	W	–	–	–	–	BPBDATA3	BPBDATA2	BPBDATA1	BPBDATA0
0xF002_0908	PC	BPCDATA	W	BPCDATA7	BPCDATA6	–	BPCDATA4	BPCDATA3	–	–	–
0xF002_090C	PD	BPDDATA	W	Read as undefined. Write as zero.							
0xF002_0914	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_0918	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_0924	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_0928	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_092C	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_0930	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_0934	PN	BPNDATA	W	–	–	–	–	–	–	BPNDATA1	BPNDATA0
0xF002_0944	–	Reserved	W	–	–	–	–	–	–	–	–
0xF002_094C	PT	BPTDATA	W	BPTDATA7	BPTDATA6	BPTDATA5	BPTDATA4	BPTDATA3	BPTDATA2	BPTDATA1	BPTDATA0
0xF002_0950	PU	BPUDATA	W	BPUDATA7	BPUDATA6	BPUDATA5	BPUDATA4	BPUDATA3	BPUDATA2	BPUDATA1	BPUDATA0
0xF002_0954	PV	BPVDATA	W	BPVDATA7	BPVDATA6	BPVDATA5	BPVDATA4	BPVDATA3	BPVDATA2	BPVDATA1	BPVDATA0

These registers define the output data in the Power Cut Mode. Each register is initialized to “0” after reset, and retains the previous value after hot reset.

Register bit value = 0y0: “L” output

Register bit value = 0y1: “H” output

Note1: In the Power Cut Mode, the BPADATA register is initialized to 0x00 and the internal pull-up circuit of port A is turned off. To continue using the pull-up circuit of port A, BPADATA must be set to 0xFF before entering the Power Cut Mode..

Table 3.24.5 backup register output Enable register list 1

register Address	correspond to port	register name	type	bit7	bit6	bit5	bit4	bit3	bit2	bit1	0bit
0xF002_0B80	PA	Reserved	–	—				Read as undefined. Write as zero.			
0xF002_0B84	PB	BPBOE	W	—				BPBOE3	BPBOE2	BPBOE1	BPBOE0
0xF002_0B88	PC	BPCOE	W	BPCOE7	BPCOE6	—	BPCOE4	BPCOE3	—	—	—
0xF002_0B8C	PD	BPDOE	W	BPDOE7	BPDOE6	BPDOE5	BPDOE4	—			
0xF002_0B94	PF	BPFOE	W	—							
0xF002_0B98	PG	BPGOE	W	—							
0xF002_0BA4	PJ	BPJOE	W	—							
0xF002_0BA8	PK	BPKOE	W	—							
0xF002_0BAC	PL	BPLOE	W	—							
0xF002_0BB0	PM	BPMOE	W	—							
0xF002_0BB4	PN	BPNOE	W	—						BPNOE1	BPNOE0
0xF002_0BC4	PR	BPROE	W	—							
0xF002_0BCC	PT	BPTOE	W	BPTOE7	BPTOE6	BPTOE5	BPTOE4	BPTOE3	BPTOE2	BPTOE1	BPTOE0
0xF002_0BD0	PU	BPUOE	W	BPUOE7	BPUOE6	BPUOE5	BPUOE4	BPUOE3	BPUOE2	BPUOE1	BPUOE0
0xF002_0BD4	PV	BPVOE	W	BPVOE7	BPVOE6	BPVOE5	BPVOE4	BPVOE3	BPVOE2	BPVOE1	BPVOE0

These registers enable or disable data output in the Power Cut Mode. Each register is initialized to “0” after reset, and retains the previous value after hot reset.

Register bit value = 0y0: Disable output

Register bit value = 01y: Enable output

Note: When using a touch screen function at PortD, the BPDOE and the BPDDATA setting are shown belows.

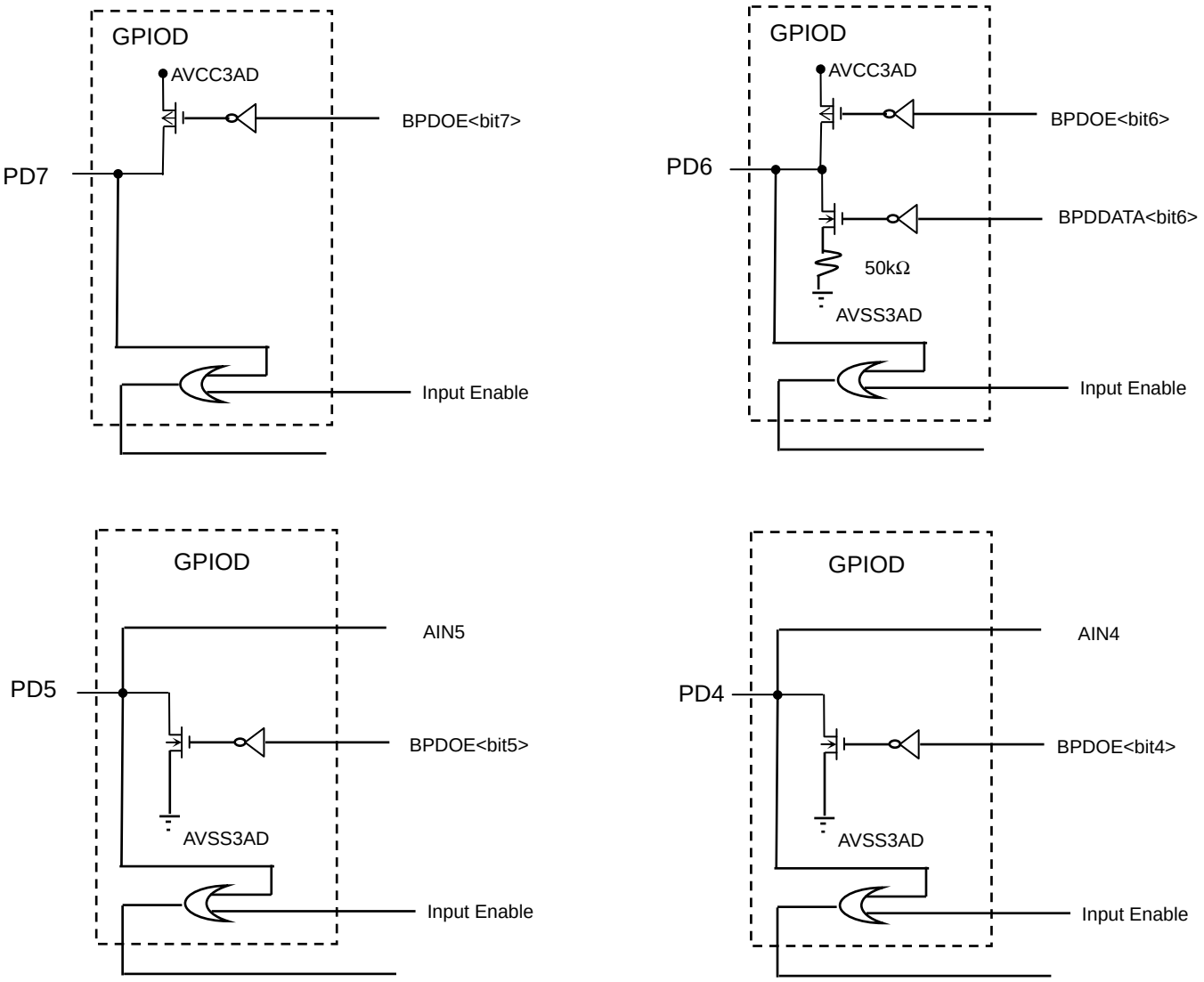


Table 3.24.6 backup register output data register list 2

register Address	correspond to port	register name	type	bit7	bit6	bit5	bit4	bit3	bit2	bit1	0bit
0xF002_0800	SA	BSADATA	W	BSADATA7	BSADATA6	BSADATA5	BSADATA4	BSADATA3	BSADATA2	BSADATA1	BSADATA0
0xF002_0804	SB	BSBDATA	W	BSBDATA7	BSBDATA6	BSBDATA5	BSBDATA4	BSBDATA3	BSBDATA2	BSBDATA1	BSBDATA0
0xF002_0808	—	Reserved	W								
0xF002_080C	—	Reserved	W								
0xF002_0810	SE	BSEDATA	W	BSEDATA7	BSEDATA6	BSEDATA5	BSEDATA4	BSEDATA3	BSEDATA2	BSEDATA1	BSEDATA0
0xF002_0814	SF	BSFDATA	W	BSFDATA7	BSFDATA6	BSFDATA5	BSFDATA4	BSFDATA3	BSFDATA2	BSFDATA1	BSFDATA0
0xF002_0818	SG	BSGDATA	W	BSGDATA7	BSGDATA6	BSGDATA5	BSGDATA4	BSGDATA3	BSGDATA2	BSGDATA1	BSGDATA0
0xF002_081C	SH	BSHDATA	W	BSHDATA7	—	—	BSHDATA4	BSHDATA3	BSHDATA2	—	—
0xF002_0824	SJ	BSJDATA	W	—	BSJDATA6	BSJDATA5	BSJDATA4	BSJDATA3	BSJDATA2	BSJDATA1	BSJDATA0
0xF002_0828	SK	BSKDATA	W	—	—	BSKDATA5	BSKDATA4	—	—	BSKDATA1	BSKDATA0
0xF002_082C	SL	BSLDATA	W	—	Note	BSLDATA5	BSLDATA4	—	BSLDATA2	BSLDATA1	BSLDATA0
0xF002_084C	—	Reserved	W								
0xF002_0850	—	Reserved	W								

These registers define the output data in the Power Cut Mode. Each register is initialized to “0” after reset, and retains the previous value after hot reset.

Register bit value = 0y0: “L” output

Register bit value = 0y1: “H” output

Note: Read as undefined.

Table 3.24.7 backup register output Enable register list 2

register Address	correspond to port	register name	type	bit7	bit6	bit5	bit4	bit3	bit2	bit1	0bit
0xF002_0A80	SA	BSAOE	W	BSAOE7	BSAOE6	BSAOE5	BSAOE4	BSAOE3	BSAOE2	BSAOE1	BSAOE0
0xF002_0A84	SB	BSBOE	W	BSBOE7	BSBOE6	BSBOE5	BSBOE4	BSBOE3	BSBOE2	BSBOE1	BSBOE0
0xF002_0A88	—	Reserved	W								
0xF002_0A8C	—	Reserved	W								
0xF002_0A90	SE	BSEOE	W	BSEOE7	BSEOE6	BSEOE5	BSEOE4	BSEOE3	BSEOE2	BSEOE1	BSEOE0
0xF002_0A94	SF	BSFOE	W	BSFOE7	BSFOE6	BSFOE5	BSFOE4	BSFOE3	BSFOE2	BSFOE1	BSFOE0
0xF002_0A98	SG	BSGOE	W	BSGOE7	BSGOE6	BSGOE5	BSGOE4	BSGOE3	BSGOE2	BSGOE1	BSGOE0
0xF002_0A9C	SH	BSHOE	W	BSHOE7	—	—	BSHOE4	BSHOE3	BSHOE2	—	—
0xF002_0AA4	SJ	BSJOE	W	—	BSJOE6	BSJOE5	BSJOE4	BSJOE3	BSJOE2	BSJOE1	BSJOE0
0xF002_0AA8	SK	BSKOE	W			BSKOE5	BSKOE4			BSKOE1	BSKOE0
0xF002_0AAC	SL	BSLOE	W	—	Note	BSLOE5	BSLOE4	—	BSLOE2	BSLOE1	BSLOE0
0xF002_0ACC	—	Reserved	W								
0xF002_0AD0	—	Reserved	W								

These registers enable or disable data output in the Power Cut Mode. Each register is initialized to “0” after reset, and retains the previous value after hot reset.

Register bit value = 0y0: Enable output

Register bit value = 01y: Disable output

Note: Read as undefined.

1. BPARELE register

Address = (0xF002_0000) + (0x0200)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:4]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[3]	BPARELE3	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[2]	BPARELE2	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[1]	BPARELE1	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[0]	BPARELE0	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable

[Description]

a. <BPARELE[3:0]>

Enable PCM release request of key input KI [3:0].

Note: When an enable setting is required, BPAIE[3:0] register bit also need to set to be an enable similarly.

2. BPDRELE register

Address = (0xF002_0000) + (0x0204)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[6]	BPDRELE6	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[5:0]	–	–	Undefined	Undefined	Read as undefined. Write as zero.

[Description]

a. <BPDRELE6>

Enable PCM release request of INTA (INTTSI).

Note: When an enable setting is required, BPDIE[6]register bit also need to set to be an enable similarly.

3. BRTRELE register

Address = (0xF002_0000) + (0x0208)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:1]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[0]	BRTRELE0	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable

[Description]

a. <BRTRELE0>

Enable PCM release request of RTC.

4. BPXRELE register

Address = (0xF002_0000) + 0x020C

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7:2]	Reserved	R/W	0y0	hold eve data	Read as undefined. Write as zero.
[1]	PD7INTB	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[0]	PC7INT9	R/W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable

(個別説明)

a. < PD7INTB>

Enable PCM release request of PD7(INTB).

Note: When an enable setting is required, BPDIE[7]register bit also need to set to be an enable similarly.

b. < PC7INT9>

Enable PCM release request of PC7(INT9) .

Note: When an enable setting is required, BPCIE[7]register bit also need to set to be an enable similarly.

5. BPAEDGE register

Address = (0xF002_0000) + (0x0220)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:4]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[3]	BPAEDGE3	R/W	0y0	hold eve data	edge selection of PCM release request 0y0 : rise edge 0y1 : fall edge
[2]	BPAEDGE2	R/W	0y0	hold eve data	edge selection of PCM release request 0y0 : rise edge 0y1 : fall edge
[1]	BPAEDGE1	R/W	0y0	hold eve data	edge selection of PCM release request 0y0 : rise edge 0y1 : fall edge
[0]	BPAEDGE0	R/W	0y0	hold eve data	edge selection of PCM release request 0y0 : rise edge 0y1 : fall edge

[Description]

a. <BPAEDGE[3:0]>

Edge selection of PCM release request of key input KI [3:0].

6. BPDEDGE register

Address = (0xF002_0000) + (0x0224)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[6]	BPDEDGE6	R/W	0y0	hold eve data	Edge selection of PCM release request 0y0: rise edge 0y1: fall edge
[5:0]	–	–	Undefined	Undefined	Read as undefined. Write as zero.

[Description]

a. <BPDEDGE6>

Edge selection of PCM release request of INTA (INTTSI).

7. BPXEDGE register

Address = (0xF002_0000) + 0x022C

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7:2]	Reserved	–	Undefined	Undefined	Read as undefined. Write as zero.
[1]	BPXEDGEB	R/W	0y0	データ保持	Edge selection of PCM release request 0y0: rise edge 0y1: fall edge
[0]	BPXEDGE9				

a. <BPXEDGEB>

Edge selection of PCM release request of INTB.

b. <BPXEDGE9>

Edge selection of PCM release request of INT9.

8. BPARINT register

Address = (0xF002_0000) + (0x0240)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:4]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[3]	BPARINT3	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request
[2]	BPARINT2	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request
[1]	BPARINT1	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request
[0]	BPARINT0	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request

[Description]

a. <BPARINT[3:0]>

PCM release interrupt status of key input KI [3:0].

Write:

0y0: Clear

0y1: Reserved

Read:

0y0: no interrupt request

0y1: interrupt request

For the factor of PCM release can be confirmed.

Please write this register bit as 0 before entering PCM status.

9. BPDRINT register

Address = (0xF002_0000) + (0x0244)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:7]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[6]	BPDRINT6	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request
[5:0]	–	–	Undefined	Undefined	Read as undefined. Write as zero.

[Description]

a. <BPDRINT6>

PCM release interrupt status of INTA (INTTSI).

Write:

0y0: Clear

0y1: Reserved

Read:

0y0: no interrupt request

0y1: interrupt request

For the factor of PCM release can be confirmed.

Please write this register bit as 0 before entering PCM status.

10. BRTRINT register

Address = (0xF002_0000) + (0x0248)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:1]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[0]	BRTRINT0	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request

[Description]

a. <BRTRINT0>

PCM release interrupt status of RTC.

Write:

0y0: Clear

0y1: Reserved

Read:

0y0: no interrupt request

0y1: interrupt request

For the factor of PCM release can be confirmed.

Please write this register bit as 0 before entering PCM status.

11. BPXRINT register

Address = (0xF002_0000) + 0x024C

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7:2]	Reserved	–	Undefined	Undefined	Read as undefined. Write as zero.
[1]	PD7INTB	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request
[0]	PC7INT9	R/W	Undefined	hold eve data	PCM release interrupt status Write: 0y0: Clear 0y1: Reserved Read: 0y0: no interrupt request 0y1: interrupt request

[Description]

a. <PD7INTB>
PCM release interrupt status of INTB.

b. <PC7INT9>
PCM release interrupt status of INT9.

Write:
0y0: Clear
0y1: Reserved
Read:
0y0: no interrupt request
0y1: interrupt request

For the factor of PCM release can be confirmed.
Please write this register bit as 0 before entering PCM status.

12. PMCDRV register

Address = (0xF002_0000) + (0x0260)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:5]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[4]	DRV_LCD	R/W	0y1	hold eve data	LCDC relation port drive power 0y0: 12 mA (1.8 V to 3.0 V) 0y1: 6 mA (3.0 V to 3.6 V)
[3:2]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[1]	DRV_MEM1	R/W	0y0	hold eve data	memory relation port drive power 0y00: Reserved 0y01: 1/2 (3.3 V ±0.3 V) 0y10: Reserved 0y11: 1/1 (1.8 V ±0.1 V)
[0]	DRV_MEM0	R/W	0y1	hold eve data	

[Description]

a. < DRV_LCD, DRV_MEM[1:0]>

These bits can be used to change the drive capability of ports related to LCD and memory according to the voltage range to be used.

13. DMCCKECTL register

Address = (0xF002_0000) + (0x0280)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:1]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[0]	DMCCKEHLD	R/W	0y0	hold eve data	output selection of SJ6 0y0: DMCCKE 0y1: PMC register setting

[Description]

a. <DMCCKEHLD >

After a system reset, the DMCCKE pin is always initialized to the "H" level. Therefore, if SDRAM has been set to self-refresh mode before entering the Power Cut Mode, the self-refresh mode will be cleared when the DMCCKE pin is initialized to the "H" level after wake-up from the Power Cut Mode. To avoid this situation, it is possible only for the DMCCKE pin, which controls SDRAM, to specify different pin levels after a system reset and after wake-up from the Power Cut Mode. The DMCCKE pin level after wake-up from the Power Cut Mode can be set in the relevant PMC register.

	DMCCKE pin status
After a system reset	"H" level
After wake-up from Power Cut Mode	To set the DMCCKE pin to "L" level: DMCCKECTL<DMCCKEHLD>= 0y1 BSJOE<BSJOE6>= 0y1 BSJDATA<BSJDATA6>= 0y0

14. PMCCTL register

Address = (0xF002_0000) + (0x0300)

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7]	PCM_ON Note)	R/W	0y0	hold eve data	Power Cut Enable 0y0: Disable 0y1: Enable
[6]	PMCPWE	R/W	0y1	hold eve data	output selection of PWE 0y0: Port function (PC2) 0y1: PMC register output (PWE)
[5:3]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[2]	Reserved	–	Undefined	Undefined	Read as undefined. Write as zero.
[1]	WUTM1	R/W	0y0	hold eve data	Warm-up timing setting 0y00: 2^9 (15.625 ms) 0y01: 2^{10} (31.25 ms) 0y10: 2^{11} (62.5 ms) 0y11: 2^{12} (125 ms)
[0]	WUTM0	R/W	0y0	hold eve data	

[Description]

a. <PCM_ON>

The Power Cut Mode is entered by writing “1” to <PCM_ON>.

Note1: The Power Cut Mode is etnered by writing “1” to <PCM_ON>. At this time, <PMCPWE> and <WMTM1:0> in the same register cannot be changed simultaneously. <PMCPWE> and <WMTM1:0> should be set while <PCM_ON>=0. When writing “1” to <PCM_ON>, do not change the values of <PMCPWE> and <WMTM1:0>.

Note2: <PCM_ON> cannot be cleared by writing “0”. To clear this bit, use the <RES_PCMON> bit in the PMCRES register.

Note3: The external PWE signal changes from “0” to “1” approximately 1.5 XT1 (48μs) after a wake-up request interrupt. Then, after a specified warm-up period and an additional interval of approximately 1 XT1 (32μs), the internal reset signal is released. Since power stabilization time depends on the response of the power source to be used and conditions on the system, determine the warm-up time in consideration of the period required until power is stabilized.

15. PMCWV1 register

Address = (0xF002_0000) + 0x0400

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:6]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[6]	PMCBDTV	W	0y0	hold eve data	Back-up register value update 0y0: Do not update 0y1: Update
[5]	PMCCTLV	R/W	0y0	hold eve data	PMCCTL register value update 0y0: Do not update 0y1: Update
[4]	DMCCKECTLV	R/W	0y0	hold eve data	DMCCKECTL register value update 0y0: Do not update 0y1: Update
[3]	PMCDRVV	R/W	0y0	hold eve data	PMCDRV register value update 0y0: Do not update 0y1: Update
[2]	BPARINTV	R/W	0y0	hold eve data	BPARINT register value update 0y0: Do not update 0y1: Update
[1]	BPAEDGEV	R/W	0y0	hold eve data	BPAEDGE register value update 0y0: Do not update 0y1: Update
[0]	BPARELEV	R/W	0y0	hold eve data	BPARELE register value update 0y0: Do not update 0y1: Update

[Description]

There are different PMC setting registers in Power Off area and Power On area to realize Power Cut Mode. And have to set the PMCWV1 register to renew the backup registers setting in Power On area.

For how to control PMC registers, see “3.24.3 PMC Registers Composition

a. <PMCCTLV>, <DMCCKECTLV>, <PMCDRVV> and other registers

0y0: Do not update

When <register bit>=0y0, values newly written to the appropriate register are not reflected in the corresponding backup register in the DVCC1B circuit.

0y1: Update

When <register bit>=0y1, values newly written to the appropriate register are reflected in the corresponding backup register in the DVCC1B circuit.

Note: PMCCTL<PCM_ON> is a special bit that cannot be updated by using the PMCWV1 register. Use the PMCRES register to update PMCCTL<PCM_ON>.

16. PMCRES register

Address = (0xF002_0000) + 0x041C

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7]	RES_PCMON	R/W	0y0	0y0	PMCCTL<PCM_ON> clear 0y0: Invalid 0y1: Clear
[6:0]	–	–	Undefined	Undefined	Read as undefined. Write as zero.

[Description]

a. <RES_PCMON>

By writing "1" to <RES_PCMON>, PMCCTL<PCM_ON> can be cleared. This bit cannot be used for waking up from the Power Cut Mode.

0y0: Invalid

0y1: <PCM_ON> clear

17. BPAIE register

Address = (0xF002_0000) + 0x0D80

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:4]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[3]	BPAIE3	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[2]	BPAIE2	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[1]	BPAIE1	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[0]	BPAIE0	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable

[Description]

a. <BPAIE[3:0]>

Enable PCM release request of key input KI [3:0].

Note: When an enable setting is required, BPARELE[3:0] register bit also need to set to be an enable similarly.

18. BPCIE register

Address = (0xF002_0000) + 0x0D88

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7]	BPAIE7	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[6:0]	–	–	Undefined	Undefined	Read as undefined. Write as zero.

[Description]

a. <BPCIE7>

Enable PCM release request of INT9.

Note: When an enable setting is required, BPXRELE[0] register bit also need to set to be an enable similarly.

19. BPDIE register

Address = (0xF002_0000) + 0x0D8C

Bit	Bit Symbol	Type	Reset Value	Hot Reset Value	Description
[31:8]	–	–	Undefined	Undefined	Read as undefined. Write as zero.
[7]	BPDIE7	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[6]	BPDIE6	W	0y0	hold eve data	PCM release request Enable 0y0: disable 0y1: enable
[5:0]	–	–	Undefined	Undefined	Read as undefined. Write as zero.

[Description]

a. <BPDIE[7:6]>

Enable PCM release request of PD7(INTB) and PD6(INTA).

Note: When an enable setting is required, BPDRELE[6] register bit also need to set to be an enable similarly.

Note: When an enable setting is required, BPXRELE[1] register bit also need to set to be an enable similarly.

3.24.8 Program Examples

The program example shown below is written assuming that it is executed from the internal RAM.

Before entering the Power Cut Mode, stop all functions that may interfere with transition to the Power Cut Mode and set the relevant pins as required.

- Stop the watchdog timer. (The watchdog timer is stopped in the initial state.)
- Stop the AD converter.
- Stop DMA operation.
- Stop the LCD controller.
- Stop the auto-refresh mode of SDRAM (change to the self-refresh mode).
- Stop DMA transfer.
- Fix pin levels as required.

  At the same time as fixing pin levels, also set the levels of the relevant pins during the Power Cut Mode. For each external interrupt that can be used for waking up from the Power Cut Mode, the active edge can be selected in the relevant register. To use the PD6 pin as INTA (INTTSI interrupt), it is necessary to disable the debounce circuit.

- Disable interrupts.
- Stop OFD operation.
- Stop PLL operation.
- Disable the internal cache memory.

```
;   Example) Wakeup from the Power Cut Mode by using  Port A[0].

;----------- Back up Data set Register in Power Cut Mode

        LDR     r0,=BPADATA         ; Port A Pull-up enable for PMC mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPBDATA         ; PB Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPCDATA         ; PC Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPDDATA         ; PD Data set in Power Cut Mode
        MOV     r1,#0x00000040
        STR     r1,[r0]

        LDR     r0,=BPNDATA         ; PN Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPTDATA         ; PT Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPUDATA         ; PU Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPVDATA         ; PV Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]
```

```
;---------- Back Up Data Output Enable Register

        LDR     r0,=BPBOE       ; PB Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPCOE       ; PC Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPDOE       ; PD Output enable
        MOV     r1,#0x000000F0
        STR     r1,[r0]

        LDR     r0,=BPNOE       ; PN Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPTOE       ; PT Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPUOE       ; PU Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BPVOE       ; PV Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]
```

```
;---------- Back up Data set Register in Power Cut Mode

        LDR     r0,=BSADATA     ; SA Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSBDATA     ; SB Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSEDATA     ; SE Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSFDATA     ; SF Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSGDATA     ; SG Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSHDATA     ; SH Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSJDATA     ; SJ Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSKDATA     ; SK Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSLDATA     ; SL Data set in Power Cut Mode
        MOV     r1,#0x000000FF
        STR     r1,[r0]
```

```
;---------- Back Up Data Output Enable Register

        LDR     r0,=BSAOE       ; SA Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSBOE       ; SB Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSEOE       ; SE Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSFOE       ; SF Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSGOE       ; SG Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSHOE       ; SH Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSJOE       ; SJ Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSKOE       ; SK Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]

        LDR     r0,=BSLOE       ; SL Output enable
        MOV     r1,#0x000000FF
        STR     r1,[r0]
```

```
;------------ Wake Up Enable Register

        LDR     r0,=BPAIE       ; PA0, set enable
        MOV     r1,#0x00000001
        STR     r1,[r0]

        LDR     r0,=BPCIE       ; Disable
        MOV     r1,#0x00000000
        STR     r1,[r0]

        LDR     r0,=BPDIE       ; Disable
        MOV     r1,#0x00000000
        STR     r1,[r0]

;------------ Wake Up Enable Register

        LDR     r0,=BPARELE     ; PA0, set enable
        MOV     r1,#0x00000001
        STR     r1,[r0]

        LDR     r0,=BPDRELE     ; Disable
        MOV     r1,#0x00000000
        STR     r1,[r0]

        LDR     r0,=BRTRELE     ; Disable
        MOV     r1,#0x00000000
        STR     r1,[r0]

        LDR     r0,=BPXRELE     ; Disable
        MOV     r1,#0x00000000
        STR     r1,[r0]

;------------ Wake Up Source Edge Select Register

        LDR     r0,=BPAEDGE     ; PA0, set rising edge
        MOV     r1,#0x00000000
        STR     r1,[r0]

        LDR     r0,=BPDEDGE     ;
        MOV     r1,#0x00000000
        STR     r1,[r0]

        LDR     r0,=BPXEDGE     ;
        MOV     r1,#0x00000000
        STR     r1,[r0]
```

```
;------------ Wake Up Source Initial Register

                LDR         r0,=BPARINT         ; Clear the status of  wakeup request
                MOV         r1,#0x00000000
                STR         r1,[r0]

                LDR         r0,=BPDRINT         ; Clear the status of  wakeup request
                MOV         r1,#0x00000000
                STR         r1,[r0]

                LDR         r0,=BRTRINT         ; Clear the status of  wakeup request
                MOV         r1,#0x00000000
                STR         r1,[r0]

                LDR         r0,=BPXRINT         ; clear the status of  wakeup request
                MOV         r1,#0x00000000
                STR         r1,[r0]

;------------ Pre PMCCTL Register set

                LDR         r0,=PMCCTL          ;PMCCTL pre set
                MOV         r1,#0x00000043
                STR         r1,[r0]

;------------ PMC Write Valid Register

                LDR         r0,=PMCWV1          ;PMCWV1
                MOV         r1,#0x0000007F
                STR         r1,[r0]

                NOP
                NOP
                NOP
                NOP
                NOP
                NOP

;------------ Write Valid Flag Check

WAIT_PMCWV1
                LDR         r0,=PMCWV1
                LDR         r0,[r0]
                AND         r0,#0x7F
                CMP         r0,#0x7F
                BNE     WAIT_PMCWV1

                NOP
                NOP

                LDR         r0,=PMCCTL          ;PMCCTL ;PMC MODE
                MOV         r1,#0xc3
                STR         r1,[r0]


NOP_LOOP
                NOP
                NOP
                NOP
                NOP
                B       NOP_LOOP

                HALT                            ;
```

3.25 USB Host Controller

The USB Host Controller (USBHC) is compliant with the USB Specification Revision 2.0 and the Open HCI Specification Release 1.0a and supports USB transfers at 12 Mbps (full-speed).The USBHC is connected to the Multi layer Bus System via on-chip SRAM.

The USBHC is subject to some restrictions. For details, see section 3.25.8.

3.25.1 System Overview

The key features of the USBHC are as follows:

(1) Supports full-speed (12 Mbps) USB devices. But Not supports Low-Speed (1.5Mbps)

(2) Supports control, bulk, interrupt and isochronous transfers.

(3) Contains two 16-byte FIFO buffers (IN and OUT) in the bus bridge logic for connecting with the CPU, allowing a maximum of 16-byte burst transfers.

(4) Supports data transfers between the FIFO buffers in the bus bridge logic and the on-chip SRAM.

3.25.2 System Configuration

The USBHC consists of the following three blocks:

(1) USBHC core (OHCI)

(2) USB transceiver

(3) CPU bus bridge logic

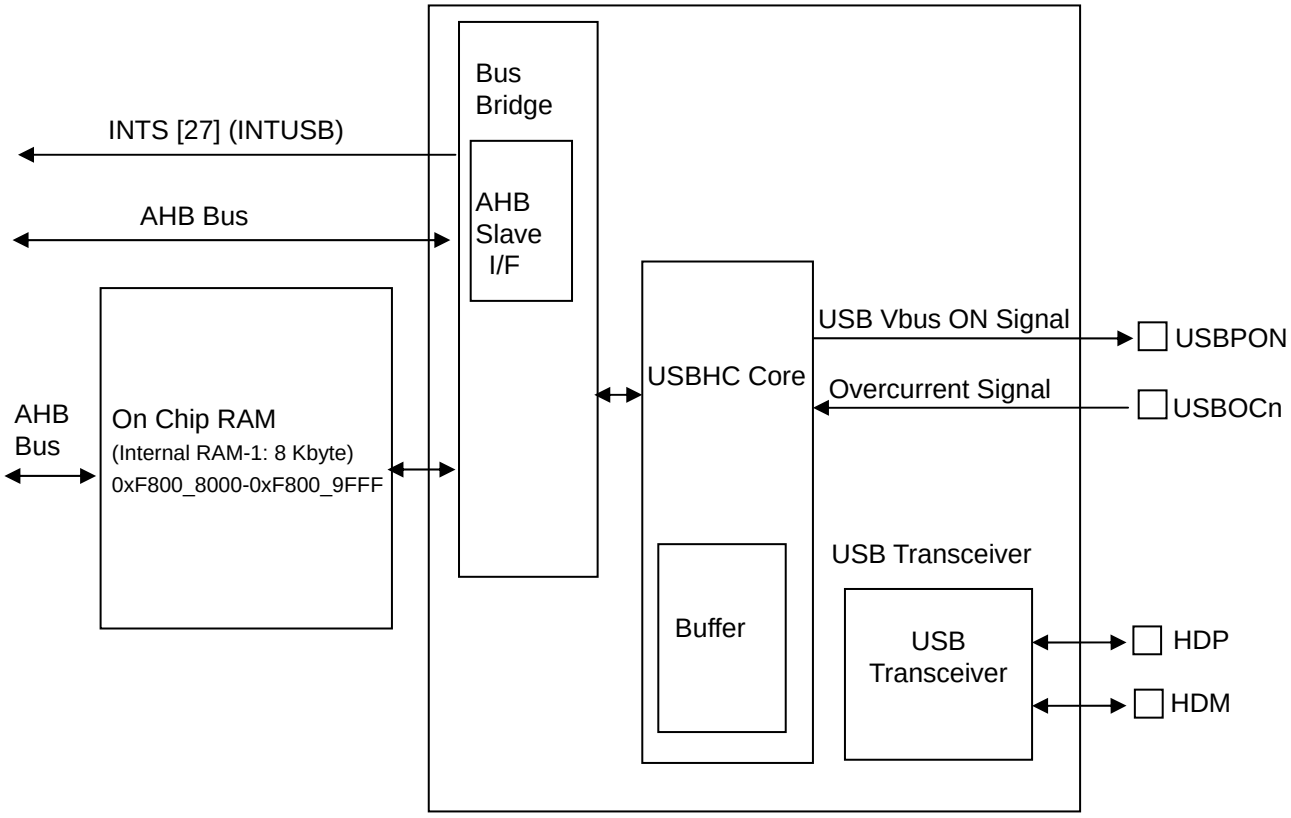


Figure 3.25.1 USB Host Controller

3.25.3 Interrupts

The USBHC generates the following interrupts:

- Scheduling Overrun
- HcDoneHead Write back
- Start of Frame
- Resume Detect
- Unrecoverable Error
- Frame Number Overflow
- Root Hub Status Change
- Ownership Change

When an event that causes an interrupt occurs, the USBHC sets the corresponding bit in the HcInterruptStatus register. At this time, if the MasterInterruptEnable (MIE) bit is enabled and the corresponding bit in the HcInterruptEnable register is enabled, a USB interrupt (INTSUB) is generated.

The USBHC driver software can clear each bit in the HcInterruptStatus register by writing a 1 to it (The driver software cannot set these bits, and the USBHC cannot clear these bits).

3.25.4 Reset

The USBHC is initialized by a hardware or software reset.

3.25.4.1 Hardware Reset

A hardware reset is generated by the external reset pin or internal reset(WDT reset, OFD reset, PCM release).

- All registers are initialized.
- The USBHC outputs the reset signal on the USB bus
- The USB state changes to the USBRESET state.
- List processing and SOF token generation are disabled.
- The FrameNumber field of the HcFmNumber register is not incremented.

3.25.4.2 Software Reset

A software reset is generated when the HostControllerReset bit in the HcCommandStatus register is set to 1.

- All OHCI registers are initialized.

  The Host Controller Driver does not modify the InterruptRouting bit and the RemoteWakeupConnected bit in the HcControl register.

  The HcBCR0 register is not initialized.
- The USBHC outputs the reset signal on the USB bus
- The USB state changes to the USBSUSPEND state.

  (The FunctionalState bit in the HcController register is set to 0x03 (USBSUSPEND).)

3.25.5 Bus Power Control

The USBHC has a control signal for an external power IC for Vbus. This signal is controlled by the USBPON pin (PT4).

To use PT4 as the USBPON pin, the port T control register (PTFC) must be set appropriately. Then, setting the LPSC bit in the HcRhStatus register to 1 makes the USBPON pin output high level.

The USBOCn pin (PT5) is used to detect overcurrent conditions. When low level is detected on this pin, the USBHC sets the OCI bit in the OHCI HcRhStatus register to 1. (To use PT5 as the USBOCn pin, the port T control register (PTFC) must be set appropriately.)

3.25.6 Registers

The USBHC contains a set of control registers compliant with the Open HCI Specification (OHCI) which are mapped into the memory space. The bus bridge logic for connecting with the CPU also includes control registers.

These registers are directly accessible from the CPU via a 32-bit bus.

Table 3.25.1 Registers Compliant with the USB Open HCI Specification

Base Address= 0xF450_0000

Register Name	Address (base+)	Description
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	
HcBCR0	0x0080	

Note 1: The addresses listed in Table 3.25.1 are those mapped on the CPU registers.

Note 2: The Open HCI Specification Release 1.0a specifies the FrameRemaining (FR) and FrameRemainingToggle (FRT) bits in the HcFmRemaining register and the FrameNumber (FN) bit in the HcFmNumber register as read-only to the Host Control Driver (HCD). However, the USB 1.1 OHCI Host Control Core allows write accesses to these registers by HCD for debug purposes. If HCD writes to these registers, undefined results will be obtained. These bits must not be written by HCD.

1. HcRevision Register

Address = (0xF450_0000) + (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)																
Reset state																
	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							
Reset state									0	0	0	1	0	0	0	0

Bit	Mnemonic	Field name	Function
[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 of 0x11 corresponds to version 1.1. All of the HC implementations that are compliant with this specification will have a value of 0x10.

2. HcControl Register

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 HostControllerFunctionalState and RemoteWakeupConnected.

Address = (0xF450_0000) + (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)																
Reset state																
	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	
Reset state						0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[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 upstream resume signaling. When this bit is set and the ResumeDetected bit in HcInterruptStatus is set, a remote wakeup is signaled to the host system. Setting this bit has no impact on the generation of hardware interrupt.
[9]	RWC	RemoteWakeup Connected	This bit indicates whether the 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 (Power on Self Test) POST. The HC clears the bit upon a hardware reset but does not alter it upon a software reset.
[8]	IR	Interrupt Routing	This bit determines the routing of interrupts generated by events registered in HcInterruptStatus. If cleared, 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 the HC.

Bit	Mnemonic	Field name	Function
[7:6]	HCFS	HostController FunctionalState ForUSB	00:USBRESET 01:USBRESUME 10:USBOPERATIONAL 11:USBSUSPEND A transition to UsbOperational from another state causes SOF generation to begin 1 ms later. HCD may determine whether the HC has begun sending SOFs by reading the StartofFrame field of the HcInterrupt register. This field may be changed by the HC only when in the UsbSuspend state. The HC may move from the UsbSuspend state to the UsbResume state after detecting the resume signaling from a downstream port. The HC enters UsbSuspend after a software reset, whereas it enters UsbReset after a hardware reset. The latter also resets the Root Hub and asserts subsequent reset signaling to downstream ports.
[5]	BLE	BulkListEnable	This bit is set to enable 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. The HC checks this bit whenever it determines to process the list. When disabled, HCD may modify the list. If HcBulkCurrentED 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	This bit is set to enable the processing of the Control list in the next Frame. If cleared by HCD, processing of the Control list does not occur after the next SOF. The HC must check this bit whenever it determines to process the list. When disabled, HCD may modify the list. If HcControlCurrentED 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 list in a Frame, the HC checks the status of this bit when it finds an Isochronous ED (F=1). If set (enabled), the HC continues processing the EDs. If cleared (disabled), the HC halts processing the periodic list (which now contains only isochronous EDs) and begins processing the Bulk and Control lists. The setting of this bit is also valid in the next Frame. * This product has some restrictions on isochronous transfers.
[2]	PLE	PeriodicList Enable	This bit is set to enable the processing of the periodic list in the next Frame. If cleared by HCD, processing of the periodic list does not occur after the next SOF. The 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 nonperiodic lists, the 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 switch to Bulk EDs. The internal count will be retained when crossing the frame boundary. In case of a reset, HCD is responsible for restoring this value. CBSR No. of Control EDs over Bulk EDs served 00 1:1 01 2:1 10 3:1 11 4:1

3. HcCommandStatus Register

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 must ensure 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 SchedulingOverrunCount field indicates the number of frames with which the Host Controller has detected the scheduling overrun error. This occurs when the Periodic 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 = (0xF450_0000) + (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	
Reset state															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			
Reset state													0	0	0	0

Bit	Mnemonic	Field name	Function
[31:18]	−	Reserved	
[17:16]	SOC	Scheduling OverrunCount	These bits are incremented at each scheduling overrun error. It is initialized to 00b and wraps around at 11b. This will be incremented when a scheduling overrun is detected even if SchedulingOverrun in HcInterruptStatus 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 the OS HCD to request a change of HC control. When set, the HC will set the OwnershipChange field in HcInterruptStatus. After the changeover, this bit is cleared and remains so until the next request is made from the OS HCD.

Bit	Mnemonic	Field name	Function
[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 the HC begins to process the head of the Bulk list, it checks BF. As long as BulkListFilled is 0, the HC will not start processing the Bulk list. If BulkListFilled is 1, the HC will start processing the Bulk list and will set BF to 0. If the HC finds a TD on the list, then the 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 the 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. This is set by HCD whenever it adds a TD to an ED in the Control list. When the HC begins to process the head of the Control list, it checks CLF. As long as ControlListFilled is 0, the HC will not start processing the Control list. If CF is 1, the HC will start processing the Control list and will set ControlListFilled to 0. If the HC finds a TD on the list, then the HC will set ControlListFilled to 1 causing the Control list processing to continue. If no TD is found on the Control list, and if HCD does not set ControlListFilled, then ControlListFilled will still be 0 when the 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 the HC. Regardless of the functional state of the 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 of HcControl, and no host bus accesses are allowed. This bit is cleared by the HC upon completion of the reset operation. The reset operation must be completed within 10 μs. This bit, when set, should not cause a reset to the Root Hub and no subsequent reset signaling should be asserted to its downstream ports.

4. HcInterruptStatus Register

This 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 is 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 may clear specific bits in this register by writing 1 to bit positions to be cleared. The Host Controller Driver may not set any of these bits. The Host Controller will never clear the bit.

Address = (0xF450_0000) + (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														
Reset state		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						
Reset state										0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31]	−	Reserved	
[30]	OC	Ownership Change	This bit is set by the HC when HCD sets the OwnershipChangeRequest field in HcCommandStatus. This event, when unmasked, will always 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] is changed.
[5]	FNO	FrameNumber Overflow	This bit is set when the MSb of HcFmNumber (bit 15) changes value from 0 to 1 or from 1 to 0, and after HccaFrameNumber is updated.
[4]	UE	Unrecoverable Error	This bit is set when the HC detects a system error not related to USB. The HC should not proceed with any processing nor signaling before the system error is corrected. HCD clears this bit after the HC is reset.

Bit	Mnemonic	Field name	Function
[3]	RD	ResumeDetected	This bit is set when the HC detects that a device on USB is asserting resume signaling. This is the transition from no resume signaling to resume signaling causing this bit to be set. This bit is not set when HCD sets the UsbResume state.
[2]	SF	StartofFrame	This bit is set by the HC at each start of a frame and after the update of HccaFrameNumber. The HC also generates a SOF token at the same time.
[1]	WDH	WritebackDone Head	This bit is set immediately after the HC writes HcDoneHead to HccaDoneHead. Further updates of the HccaDoneHead will not occur until this bit is cleared. HCD should only clear this bit after it saves 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 of HcCommandStatus to be incremented.

5. HcInterruptEnable Register

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 = (0xF450_0000) + (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															
Reset state	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						
Reset state										0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31]	MIE	MasterInterrupt Enable	A '0' written to this field is ignored by the HC. A '1' written to this field disables 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: Ignored 1: Disables interrupt generation due to Ownership Change.
[29:7]		Reserved	
[6]	RHSC	RootHubStatus Change	0: Ignored 1: Enables interrupt generation due to Root Hub Status Change.
[5]	FNO	FrameNumber Overflow	0: Ignored 1: Enables interrupt generation due to Frame Number Overflow.
[4]	UE	Unrecoverable Error	0: Ignored 1: Enables interrupt generation due to Unrecoverable Error.
[3]	RD	ResumeDetected	0: Ignored 1: Enables interrupt generation due to Resume Detect.
[2]	SF	StartofFrame	0: Ignored 1: Enables interrupt generation due to Start of Frame.
[1]	WDH	WritebackDone Head	0: Ignored 1: Enables interrupt generation due to HcDoneHead Writeback.
[0]	SO	Scheduling Overrun	0: Ignored 1: Enables interrupt generation due to Scheduling Overrun.

6. HcInterruptDisable Register

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 = (0xF450_0000) + (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															
Reset state	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						
Reset state										0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31]	MIE	MasterInterrupt Enable	A '0' written to this field is ignored by the 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: Ignored 1: Disables interrupt generation due to Ownership Change.
[29:7]		Reserved	
[6]	RHSC	RootHubStatus Change	0: Ignored 1: Disables interrupt generation due to Root Hub Status Change.
[5]	FNO	FrameNumber Overflow	0: Ignored 1: Disables interrupt generation due to Frame Number Overflow.
[4]	UE	Unrecoverable Error	0: Ignored 1: Disables interrupt generation due to Unrecoverable Error.
[3]	RD	ResumeDetected	0: Ignored 1: Disables interrupt generation due to Resume Detect.
[2]	SF	StartofFrame	0: Ignored 1: Disables interrupt generation due to Start of Frame.
[1]	WDH	WritebackDone Head	0: Ignored 1: Disables interrupt generation due to HcDoneHead Writeback.
[0]	SO	Scheduling Overrun	0: Ignored 1: Disables interrupt generation due to Scheduling Overrun.

7. HcHCCA Register

The HcHCCA register contains 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 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 = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0								

Bit	Mnemonic	Field name	Function
[31:8]	HCCA	HostController Communication Area	This is the base address of the Host Controller Communication Area.
[7:0]	−	Reserved	

8. HcPeriodCurrentED Register

The HcPeriodCurrentED register contains the physical address of the current Isochronous or Interrupt Endpoint Descriptor.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Function
[31:4]	PCED	PeriodCurrent ED	This is used by the HC to point to the head of one of the Periodic lists which will be processed in the current Frame. The content of this register is updated by the HC after a periodic ED is processed. HCD may read the content in determining which ED is currently being processed at the time of reading.
[3:0]	−	Reserved	

9. HcControlHeadED Register

The HcControlHeadED register contains the physical address of the first Endpoint Descriptor of the Control list.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Function
[31:4]	CHED	ControlHeadED	The HC traverses the Control list starting with the HcControlHeadED pointer. The content is loaded from HCCA during the initialization of the HC.
[3:0]	−	Reserved	

10. HcControlCurrentED Register

The HcControlCurrentED register contains the physical address of the current Endpoint Descriptor of the Control list.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Function
[31:4]	CCED	ControlCurrentED	This pointer is advanced to the next ED after serving the present one. The HC continues processing the list from where it left off in the last Frame. When it reaches the end of the Control list, the HC checks the ControlListFilled field of HcCommandStatus. If set, the HC copies the content of HcControlHeadED to HcControlCurrentED and clears the bit. If it is not set, the HC does nothing. HCD is allowed to modify this register only when the ControlListEnable field of HcControl is cleared. When set, HCD only reads the instantaneous value of this register. Initially, this is set to zero to indicate the end of the Control list.
[3:0]	−	Reserved	

11. HcBulkHeadED Register

The HcBulkHeadED register contains the physical address of the first Endpoint Descriptor of the Bulk list.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Function
[31:4]	BHED	BulkHeadED	The HC traverses the Bulk list starting with the HcBulkHeadED pointer. The content is loaded from HCCA during the initialization of the HC.
[3:0]	−	Reserved	

12. HcBulkCurrentED Register

The HcBulkCurrentED register contains the physical address of the current endpoint of the Bulk list. As the Bulk list will be served in a round-robin fashion, the endpoints will be ordered according to their order of insertion to the list.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Function
[31:4]	BCED	BulkCurrentED	This is advanced to the next ED after the HC has served the present one. The HC continues processing the list from where it left off in the last Frame. When it reaches the end of the Bulk list, the HC checks the BulkListFilled field of HcCommandStatus. If set, the HC copies the content of HcBulkHeadED to HcBulkCurrentED and clears the bit. If it is not set, the HC does nothing. HCD is only allowed to modify this register when the BulkListEnable field of HcControl is cleared. When set, HCD only reads the instantaneous value of this register. This is initially set to zero to indicate the end of the Bulk list.
[3:0]	−	Reserved	

13. HcDoneHead Register

The HcDoneHead register contains the physical address of the last completed Transfer Descriptor that has been added to the Done queue. In normal operation, the Host Controller Driver should not need to read this register as its content is periodically written to the HCCA.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	Mnemonic	Field name	Function
[31:4]	DH	DoneHead	When a TD is completed, the HC writes the content of HcDoneHead to the NextTD field of the TD. The HC then overwrites the content of HcDoneHead with the address of this TD. This is set to zero whenever the HC writes the content of this register to HCCA. It also sets the WritebackDoneHead field of HcInterruptStatus.
[3:0]	−	Reserved	

14. HcFmInterval Register

The HcFmInterval register contains 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 may transmit or receive without causing scheduling overrun. The Host Controller Driver may carry out minor adjustment on the FrameInterval by writing a new value over the present one at each SOF. This provides the programmability required for the Host Controller to synchronize with an external clocking resource and to adjust any unfixed local clock offset.

Address = (0xF450_0000) + (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														
Reset state	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													
Reset state			1	0	1	1	1	0	1	1	0	1	1	1	1	1

Bit	Mnemonic	Field name	Function
[31]	FIT	FrameInterval Toggle	HCD toggles this bit each time 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 a scheduling overrun. The field value is calculated by 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 should store the current value of this field before resetting the HC. By setting the HostControllerReset field of HcCommandStatus the HC resets this field to its nominal value. HCD may choose to restore the stored value upon the completion of the Reset sequence.

15. HcFmRemaining Register

The HcFmRemaining register is a 14-bit down counter showing the bit time remaining in the current Frame.

Address = (0xF450_0000) + (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															
Reset state	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													
Reset state			0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31]	FRT	FrameRemaining Toggle	This bit is loaded from the FrameIntervalToggle field of HcFmInterval whenever FrameRemaining reaches 0. This bit is used by HCD for the synchronization between FmInterval and FmRemaining.
[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 value specified in HcFmInterval at the next bit time boundary. When entering the UsbOperational state, the HC re-loads the content with the FrameInterval of HcFmInterval and uses the updated value from the next SOF.

16. HcFmNumber Register

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 can use the 16-bit value specified in this register and generate a 32-bit frame number without requiring frequent access to the register.

Address = (0xF450_0000) + (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)																
Reset state																
	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31:16]	−	Reserved	
[15:0]	FN	FrameNumber	This is incremented when HcFmRemaining is re-loaded. It will be rolled over to 0x0000 after 0xFFFF. When entering the UsbOperational state, this will be incremented automatically. The content will be written to HCCA after the HC increments the FrameNumber at each frame boundary and sends a SOF but before the HC reads the first ED in that Frame. After writing to HCCA, the HC will set StartofFrame in HcInterruptStatus.

17. HcPeriodicStart Register

The HcPeriodicStart register has a 14-bit programmable value which determines the earliest time the HC should start processing the periodic list.

Address = (0xF450_0000) + (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)																
Reset state																
	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													
Reset state			0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31:14]	−	Reserved	
[13:0]	PS	PeriodicStart	After a hardware reset, this field is cleared. This is then set by HCD during HC initialization. The value is calculated roughly as 10% off from HcFmInterval. A typical value will be 0x3E67. When HcFmRemaining reaches the value specified, processing of the periodic lists will have priority over Control/Bulk processing. The HC will therefore start processing the Interrupt list after completing the current Control or Bulk transaction that is in progress.

18. HcLSThreshold Register

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 = (0xF450_0000) + (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)																
Reset state																
	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											
Reset state					0	1	1	0	0	0	1	0	1	0	0	0

Bit	Mnemonic	Field name	Function
[31:12]	−	Reserved	
[11:0]	LST	LSThreshold	This field contains a value which is compared to the FrameRemaining field prior to initiating a low-speed transaction. The transaction is started only if FrameRemaining is larger than this field. The value is calculated by HCD with the consideration of transmission and setup overhead.

19. HcRhDescriptorA Register

The HcRhDescriptorA register is one of the two registers describing the characteristics of the Root Hub. Reset values are implementation-specific. The descriptor length (11), descriptor type (TBD), and hub controller current (0) fields of the hub Class Descriptor are emulated by HCD. All other fields are located in the HcRhDescriptorA and HcRhDescriptorB registers.

Address = (0xF450_0000) + (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															
Reset state	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							
Reset state				0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	Mnemonic	Field name	Function
[31:24]	POTPGT	PowerOnTo PowerGoodTime	This byte specifies the duration HCD has to wait before it accesses 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 × 2 ms.
[23:13]		Reserved	
[12]	NOCP	NoOverCurrent Protection	This bit describes how the overcurrent status for the Root Hub ports is 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: Overcurrent protection not supported
[11]	OCPM	OverCurrent ProtectionMode	This bit describes how the overcurrent status for the Root Hub ports is reported. At reset, this fields should reflect the same mode as that of PowerSwitchingMode. This field is valid only if the NoOverCurrentProtection 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 whether ports are always powered. It is implementation-specific. When this bit is cleared, the PowerSwitchingMode 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 NoPowerSwitching field is cleared. 0: All ports are powered at the same time. 1: Each port is powered individually. This mode allows port power to be controlled by either the global switching or per-port switching. If the PortPowerControlMask bit is set, the port responds only to port power commands (Set/ClearPortPower). If the port mask is cleared, then the port is controlled only by the global power switch (Set/ClearGlobalPower).
[7:0]	NDP	Number DownstreamPorts	These bits specify the number of downstream ports supported by the Root Hub. It is implementation-specific. This module has one port, so 0x01 is read.

20. HcRhDescriptorB Register

The HcRhDescriptorB register is one of the two resisters for describing the characteristics of the Root Hub. These fields are written during initialization to correspond with the system operation. Reset values are implementation-specific.

Address = (0xF450_0000) + (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															
Reset state	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															
Reset state	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31:16]	PPCM	PortPower ControlMask	Each bit indicates if a port is affected by a global power control command when PowerSwitchingMode of HcRhDescriptorA is set. When set, the port's power state is only affected 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 indicates a port of the Root Hub. When cleared, the attached device is removable. When set, the attached device is not removable. bit0: Reserved bit1: Device attached to Port#1

21. HcRhStatus Register

The HcRhStatus register is divided into two fields. 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 0.

Address = (0xF450_0000) + (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
Reset state	–														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
Reset state	0														0	0

Bit	Mnemonic	Field name	Function
[31]	CRWE	ClearRemote WakeupEnable	Writing a 1 clears DeviceRemoteWakeupEnable. Writing a 0 has no effect.
[30:18]	−	Reserved	
[17]	OCIC	OverCurrent Indicator Change	This bit is set by hardware when a change occurs in the OCI field of this register. HCD clears this bit by writing a 1. Writing a 0 has no effect.
[16]	LPSC	LocalPower StatusChange	(read)LocalPowerStatusChange The Root Hub does not support the local power status feature, and therefore this bit is always read as 0. (write)SetGlobalPower In global power mode (PowerSwitchingMode = 0), this bit is set to 1 to turn on power to all ports (clear PortPowerStatus). In per-port power mode, it sets PortPowerStatus only on ports whose PortPowerControlMask bit is not set. Writing 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) Writing a 1 sets DeviceRemoteWakeupEnable. Writing a 0 has no effect.
[14:2]		Reserved	
[1]	OCI	OverCurrent Indicator	This bit reports current 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 to be implemented, this bit should always be set to 0.
[0]	LPS	LocalPower Status	(read)LocalPowerStatus The Root Hub does not support the local power status feature, and therefore this bit is always read as 0. (write)ClearGlobalPower In global power mode (PowerSwitchingMode = 0), this bit is set to 1 to turn off power to all ports (clear PortPowerStatus). In per-port power mode, this bit clears PortPowerStatus of ports whose PortPowerControlMask bit is not set. Writing a 0 has no effect.

22. HcRhPortStatus Register

The HcRhPortStatus register is used to control and report port events on a per-port basis. NumberDownstreamPorts of 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 must be postponed until the transaction completes. Reserved bits should always be written 0.

Address = (0xF450_0000) + (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
Reset state												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
Reset state							X	0				0	0	0	0	0

Bit	Mnemonic	Field name	Function
[31:21]	−	Reserved	
[20]	PRSC	PortResetStatus Change	This bit is set at the end of the 10-ms port reset signal. HCD writes a 1 to clear this bit. Writing a 0 has no effect. 0: Port reset is not complete 1: Port reset is complete
[19]	OCIC	PortOverCurrent IndicatorChange	This bit is valid only if overcurrent conditions are reported on a per-port basis. This bit is set when Root Hub changes the PortOverCurrentIndicator bit. HCD writes a 1 to clear this bit. Writing a 0 has no effect. 0: No change in PortOverCurrentIndicator 1: PortOverCurrentIndicator is changed
[18]	PSSC	PortSuspend StatusChange	This bit is set when the full resume sequence is completed. This sequence includes the 20-s resume pulse, LS EOP, and 3-ms resynchronization delay. HCD writes a 1 to clear this bit. Writing a 0 has no effect. This bit is also cleared when ResetStatusChange is set. 0: Resume is not completed 1: Resume is completed

Bit	Mnemonic	Field name	Function
[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. HCD writes a 1 to clear this bit. Writing a 0 has no effect. 0: No change in PortEnableStatus 1: PortEnableStatus is changed
[16]	CSC	ConnectStatus Change	This bit is set whenever a connect or disconnect event occurs. HCD writes a 1 to clear this bit. Writing 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 should not occur if the port is disconnected. 0: No change in CurrentConnectStatus 1: CurrentConnectStatus is 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 the CurrentConnectStatus is set. 0: A full-speed device is attached 1: A low-speed device is attached (write)ClearPortPower HCD clears the PortPowerStatus bit by writing a 1 to this bit. Writing a 0 has no effect.

Bit	Mnemonic	Field name	Function
[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 to be enabled is determined by PowerSwitchingMode 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 should be reset. 0: Port power off 1: Port power on (write)SetPortPower HCD writes a 1 to set the PortPowerStatus bit. Writing 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, port reset signaling is asserted. After reset is complete, 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 HCD sets the port reset signaling by writing a 1 to this bit. Writing 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.

Bit	Mnemonic	Field name	Function
[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 the 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: An overcurrent condition detected (write)ClearSuspendStatus HCD writes a 1 to initiate a resume. Writing 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 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 HCD sets the PortSuspendStatus bit by writing a 1 to this bit. Writing 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.

Bit	Mnemonic	Field name	Function
[1]	PES	PortEnable Status	(read)PortEnableStatus This bit indicates whether the port is enabled or disabled. The Root Hub may clear this bit when an overcurrent condition, disconnect event, switched-off power, or operational bus 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 when port is suspend when SuspendStatusChange is set. 0: Port is disabled 1: Port is enabled (write)SetPortEnable HCD sets PortEnableStatus by writing a 1. Writing 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 is connected 1: A device is connected (write)ClearPortEnable HCD writes a 1 to this bit to clear the PortEnableStatus bit. Writing a 0 has no effect. The CurrentConnectStatus is not affected by any write. Note: This bit always reads 1 when the attached device is nonremovable (DeviceRemoveable[NDP]).

23. HcBCR0 Register

The HcBCR0 register controls clock supply from the USB bridge logic to the USB host core and the SUSPEND state of the USB transceiver. To enter Power Cut Mode with the USB transceiver in the SUSPEND state, write a 1 to the TRANS_SUSP bit.

Address = (0xF450_0000) + (0x0080)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bit Symbol	Reserved	TRNSSUSP	OVCE	Reserved												
Read/Write	R	R/W	R/W	R												
Reset state	0	1	0	0												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bit Symbol	Reserved														Reserved	Reserved
Read/Write	R														R/W	R/W
Reset state	0														0	0

Bit	Mnemonic	Field name	Function
[31]	–	Reserved	
[30]	TRNS_SUSP	Transceiver Suspend	This bit controls the SUSPEND state of the USB transceiver. To enter STOP mode or power cut mode with the USB transceiver in the SUSPEND state, set this bit to 1. 0: - (Controlled by the USB Host Controller) 1: Suspend
[29]	OVCE	USB Host Over Current Input Enable	USB Host Over Current input Enable 0y0: Enable 0y1: Disable
[28:2]	–	Reserved	
[1:0]	–	Reserved	Write as zero

Note: When the over current input enable bit is not set, the USBOCn pin must be used as output port.

3.25.7 Notes on Setting

3.25.7.1 Notes on USB Clock Setting

Before access the USB Host circuits(registers), please set the CLKCR5<USBH_CLKEN>= 0y1, and the USB host clock output is Enable status.

To use X1USB as the USB clock, set the procedure as following and can select the frequency at 1/4f_{PLL} or 1/3f_{PLL}.

(1) When X1USB is used.

```
SYSCR8  ←  0yXXXX_X001      ; USBH_CLKSEL=0y001
                            ; X1USB clock
CLKCR5  ←  0yXXX1_XXXX      ; USBH_CLKEN=1
                            ; USB HOST Clock enable
```

Note 1: the setting for selecting X1USB should be made while the X1USB are in a stable state.

Note 2: Before entering power cut mode (PMC), set CLKCR5<USBH_CLKEN> = 0y0. After exiting power cut mode, make sure that X1USB is in a stable state before selecting X1USB as the USB clock.

Note 3: When stopping the USB, set CLKCR5<USBH_CLKEN> = 0y0.

(2) When $f_{PLL}/4$ is used after reset release (using the PLL).

In case of using X1=24MHz and 8PLL

```
SYSCR8  ←  0yXXXX_X100      ; USBH_CLKSEL=0y100
                            ; 1/4fPLL of PLL output clock
CLKCR5  ←  0yXXX1_XXXX      ; USBH_CLKEN=1
                            ; USB HOST Clock enable
```

Note: the setting for selecting f_{PLL} hould be made while the PLL clock is in a stable state.

(3) When $f_{PLL}/3$ is used after reset release (using the PLL).

In case of using X1=24MHz and 6PLL

```
SYSCR8  ←  0yXXXX_X010      ; USBH_CLKSEL=0y010
                            ; 1/3fPLL of PLL output clock
CLKCR5  ←  0yXXX1_XXXX      ; USBH_CLKEN=1
                            ; Clock enable for USB HOST
```

Note: the setting for selecting f_{PLL} hould be made while the PLL clock is in a stable state.

3.25.7.2 Notes on Oscillator

When using this device with a built-in USB Host controller, it is recommended to use a crystal oscillator under 24MHz ± 100ppm based on the USB specification.

To generate USB clocks by the built-in PLL, the specifications provided by USB may not be satisfied depending on the implementation environment, condition, or fluctuation.

To be certified for the USB logo compliance, the 48MHz clock with an accuracy of ±100ppm or lower must be input through X1USB.

3.25.7.3 Notes on Entering Power Cut Mode

To shift to the power cut mode, set the USB to SUSPEND state first. After state in Power Cut Mode, the power supply of USB host controller which are DVCC1A and AVCC3H can be turn off by external circuits.

3.25.8 Restrictions on the USB Host Controller

1. For an isochronous transfer, a frame number to be transferred is defined in an Isochronous Transfer Descriptor (ITD). However, when frame numbers are not synchronized between the Host and software, and if the descriptor to be executed with a previous frame is scheduled later, the host determines that a time error has occurred and writes back DATAOVERRUN to the CC field of the ITD. However, if the following conditions are met, the host will write back inappropriate status (NOERROR).

   <Conditions>

   The above problem occurs if both the following two conditions are met:

   1. ITD.FC[2:0] = R[2:0]
   2. ITD.FC[2:0] < R[15:0]

   where ITD.FC indicates the number of times an ITD is executed, and

   R = HcFmNumber (current frame number) – ITD.SF (transfer start frame number).

   Make sure that each ITD is synchronized to the current frame number. If not, this ITD should not be linked.

2. For a low-speed IN transfer by the host, the USB 2.0 Specification defines the inter-packet delay (the time from when the Host receives a data packet to when the host transfers a handshake packet) as less than 7.5 bit times. In this product, however, the inter-packet delay is about 9.2 bit times in the worst case.

3. If a fatal error occurs on the USB system and the host detects this error (e.g., Master Abort, Target Abort, etc. on the PCI bus), the OHCI core sets the UnrecoverableError (UE) bit in the HcInterruptStatus register.

   At this time, if the Unrecoverable Error (UE) bit is set and the UE bit in the HcInterruptEnable register is set, a hardware interrupt is generated.

   After this interrupt is detected, a software reset (HcCommandStatus.HCR = 1'b1) is required to recover from the UE state, and the host then moves to the SUSPEND state.

   After the software reset, OHCI registers are initialized. If a remote wake-up occurs on the device, the host remains in the SUSPEND state.

   When the remote wake-up function is to be used, a program for recovering from the SUSPEND state must be implemented.

   Programming examples:

   1) After initializing OHCI registers by a software reset, set a value other than USBSUSPEND (2'b11) to the HcControl.HCFS field.
   2) When a remote wake-up is detected, the HcInterruptStatus.RD bit is set to 1'b1. After detecting this interrupt, set a value other than USBSUSPEND (2'b11) to the HcControl.HCFS field.

4. When HcRhDescriptorA.NPS[9] is set to 1'b1 when an overcurrent is occurred, PortResetStatus.PRS[4] and PortSuspendStatus.PSS[2] of the HcRhPortStatus register is not cleared. Therefore, do not set HcRhDescriptorA.NPS[9] = 1'b1 and HcRhDescriptorB.DR[PortNo] = 1'b1.

5. To set the HcRhStatus.DRWE[15] when connecting with Full-Speed/Low-Speed device, the status is not shifted from USBSUSPEND to USBRESUME even Remote Wake-up is occurred.
   As some restrictions, the status is not shifted from USBSUSPEND to USBRESUME even if using Remote Wake-up event. However, HcInterruptStatus.RD[3] is set appropriately at the same time, so that software can be modified to switch the status by checking this bit.
   When not to use Remote Wake-up event, do not set the HcRhStatus.DRWE[15] to 1'b1.

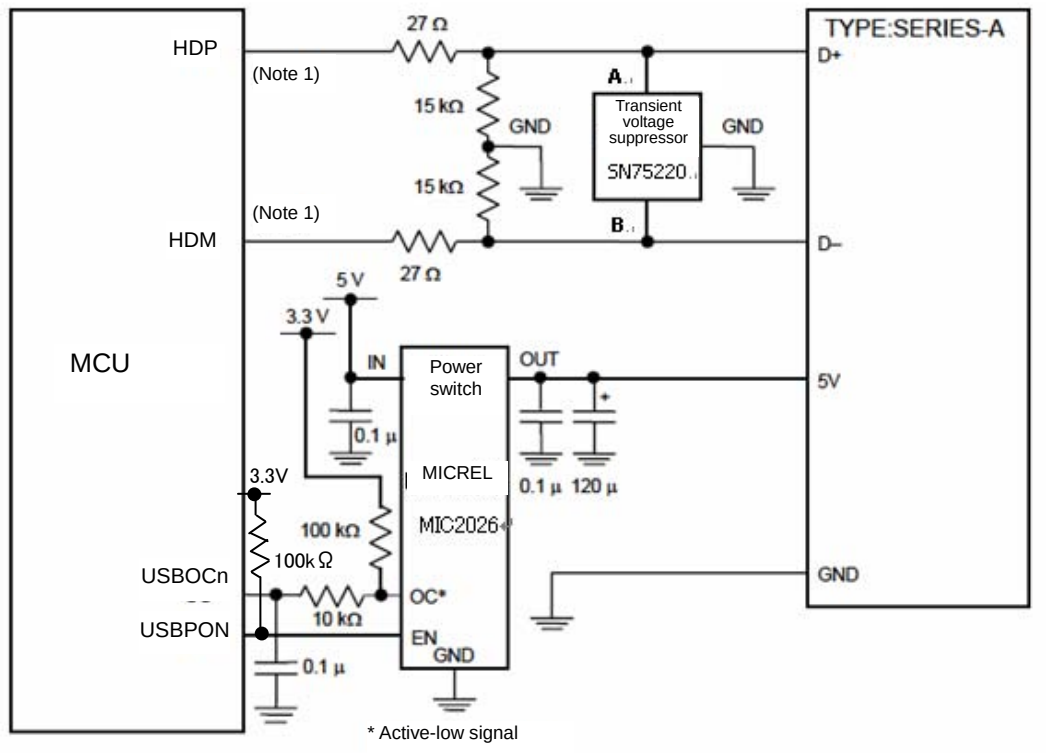
6. When supporting the overcurrent for device system, NoOverCurrentProtection.NOCP[12] of HcRhDescriptorA register must set to 1'b0.

7. To not use the overcurrent detection function of the USB host controller, the PT5 (USBOC) pin must be used as an output port or a fixed output.

8. When Isochronous transfers are used, do not generate schedule overrun.

9. If port is disabled after port reset is performed, perform reset port again.

3.25.9 Connection Example



Bus power switch device: TPS2052 from Texas Instruments
MIC2526-1BM from MICREL
MIC2536-1BN from MICREL
Transient voltage suppressor device: SN75240 from Texas Instruments
Resistance precision: 5%
Resistance rating: 1/2 W for 27 ohms (Recommended)
Capacitor: Low ESR type 120 uF capacitor (OS-CON, etc.) (Recommended)

Note 1: Do not apply voltages exceeding the absolute maximum rating.

Note 2: When designing your board, make sure that the HDP and HDM- pins are placed at the equal distance from the USB A receptacle.

Note 3: A suppressor device is not required in the USB specifications.

Note 4: After releasing a reset, USBOCn and USBPON pin will be input mode. These pins need to correspond in the circuit.

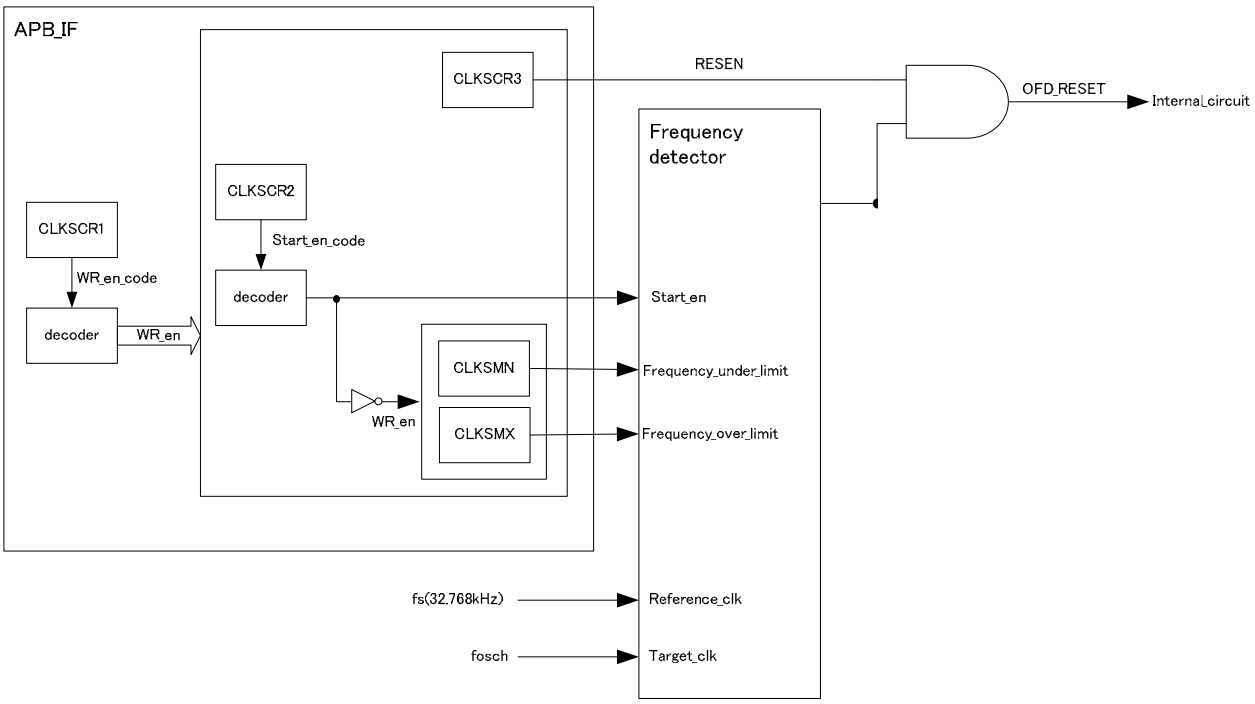
3.26 OFD (Oscillation Frequency Detector)

3.26.1 Outline

The Oscillation Frequency Detector (OFD) generates a reset when the high-frequency oscillation frequency (f_{OSCH}) falls or rises outside of the normal frequency range specified by the lower and higher detection frequency setting registers. When an abnormal frequency condition is detected, it can be notified to an internal reset circuit. The high-frequency clock (f_{OSCH}) is used as a detection clock, and the low-frequency clock (f_s: 32 KHz) is used as a reference clock of the OFD. If the 32 kHz reference clock stops due to an external cause, etc., the OFD detects an abnormal condition and resets the internal circuitry.

Note: In the PCM mode, the OFD circuitry is not powered and is not operational. So stop the OFD operation before the PCM execution.

3.26.2 OFD Block Diagram



3.26.3 Description of Operation

1) Reset generation and release

The OFD generates a reset on the second or third rising edge of the low-frequency clock after detecting a clock (f_{OSCH}) fault (including a stop state). The OFD is also capable of detecting low-frequency clock faults.

When a high-frequency clock fault occurs and the frequency ratio goes outside the range specified in the CLKSMN and CLKSMX registers, the OFD generates a reset on the second rising edge of the low-frequency clock after detecting a clock fault. Generation of a reset does not clear the clock fault detection, and the reset is released when the clock resumes stable oscillation and the frequency ratio returns to the specified normal range.

In the case of the low-frequency clock, a reset is generated when the clock stops. The reset is not released until the low-frequency clock resumes oscillation. When both fs and fosch clock are in unstable, the operation is not guaranteed.

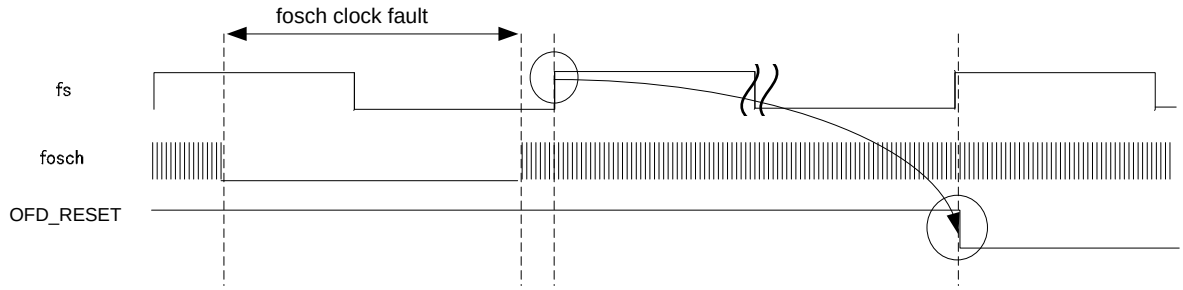


Figure 3.26.1 Timing of reset generating (fosch clock fault)

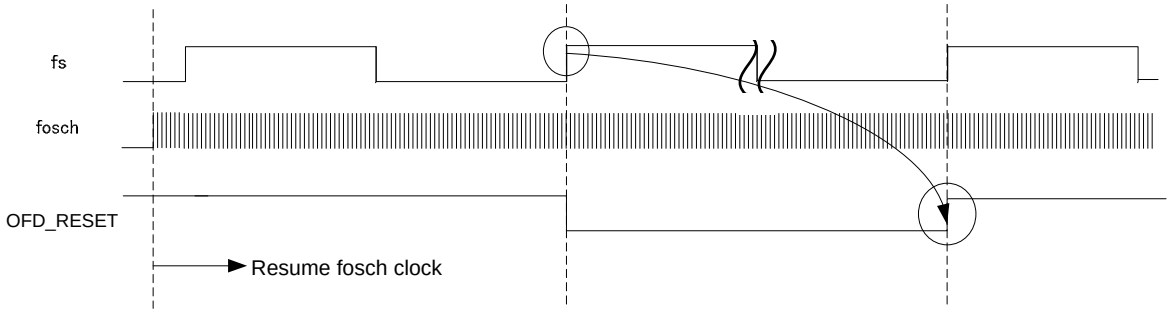


Figure 3.26.2 Timing of reset releasing (fosch clock fault)

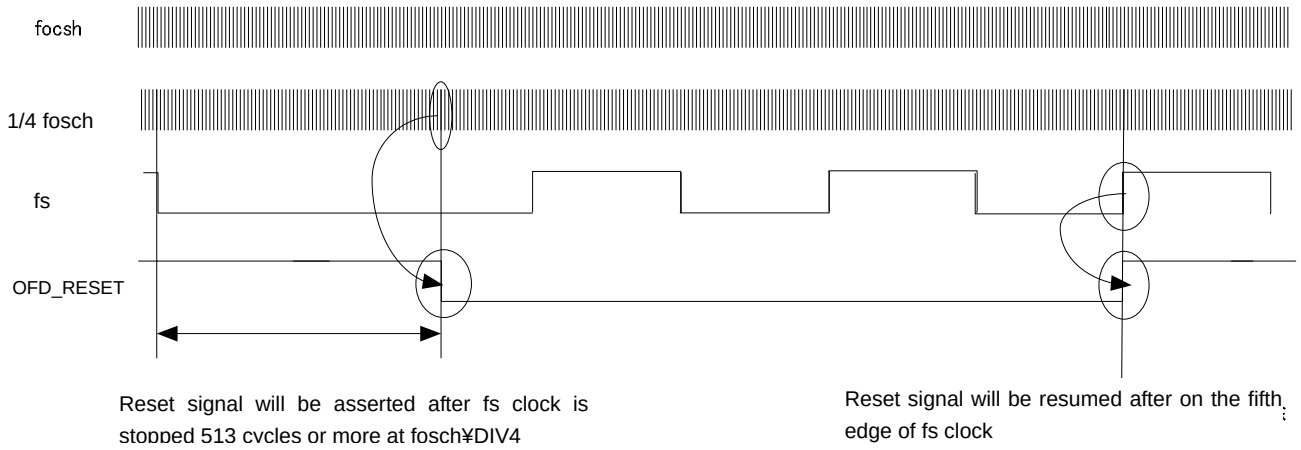


Figure 3.26.3 Timing of reset generating and releasing (fs clock fault)

Note : A clock fault detection reset is also generated if either of the high-frequency or low-frequency clock frequency temporarily goes outside the specified range of the frequency ratio due to noise, etc.

2) Initialization of OFD registers

The OFD registers (CLKSCR1, CLKSCR2, CLKSCR3, CLKSMN and CLKSMX) are initialized by an external reset (RESETn pin = Low).

The OFD registers are also initialized when the PCM mode is exited.

Note: The OFD registers are not initialized by a WDT reset or OFD reset.

3) How to confirm the clock fault detection

The fault detection can be confirmed by OFD status flag (CLKSCR3<CLKSF>=1) in OFD circuit.

4) How to set the minimum and maximum clock fault detection values

The configuration frequency = (frequency) + (frequency error ratio) + (OFD circuit error ratio) + α

The actual registers (CLKSMN and CLKSMX) setting value is calculated by above the configuration values

The following shows the calculation formulas and register setting examples when f_{OSCH} = 24 MHz (assuming the guaranteed oscillation frequency error is ± 1%).

- Example of calculation for the configuration frequency

The OFD of this product is capable of detecting f_{OSCH} deviations exceeding 10%. It is therefore recommended to set CLKSMN and CLKSMX to approximately ± 10% of f_{OSCH}.

Frequency (Low) = 24 MHz*(1-10%) = 21.6 MHz

Frequency (High) = 24 MHz*(1+10%) = 26.4 MHz

- How to set the CLKSMN and CLKSMX registers

Frequency (Low) = 21.6 MHz

Frequency (High) = 26.4 MHz

When the oscillation clock frequency goes outside the specified range, an OFD reset is generated.

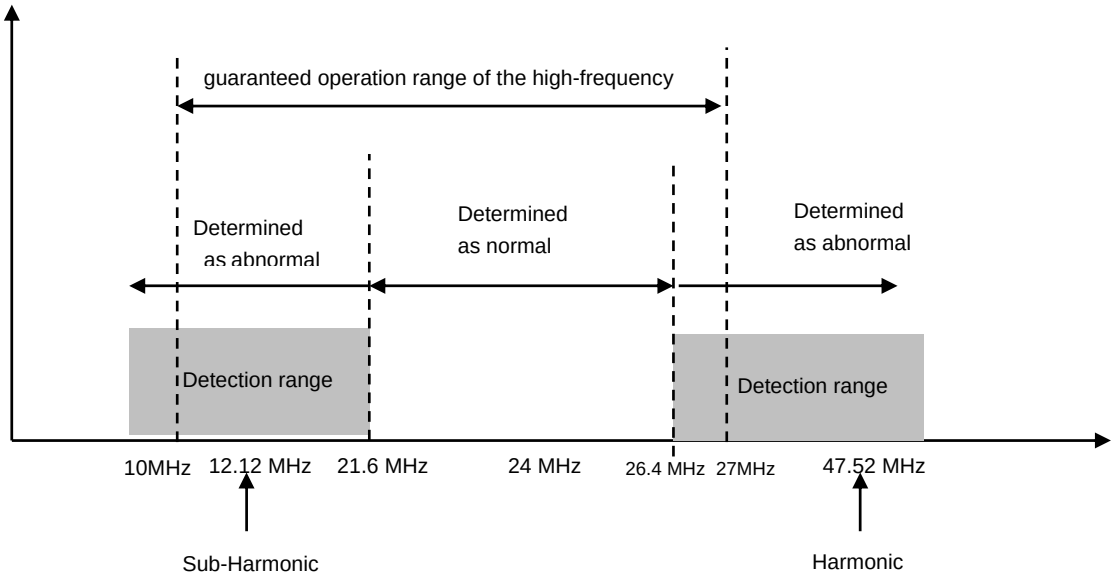
$$\text{CLKSMN set value} = \frac{frequency(Low)}{\text{fs}\times 4} = \frac{21.6}{32.768\times 10^{-3}\times 4} \quad 165 = 0xA5$$

(fractions to be rounded up)

$$\text{CLKSMX set value} = \frac{frequency(High)}{\text{fs}\times 4} = \frac{26.4}{32.768\times 10^{-3}\times 4} \quad 201 = 0xC9$$

(fractions to be rounded up)

注) 本周波数検知回路は発振周波数の高調波、低調波発振を検知するための回路です。発振子および発振器の発振精度誤差を測定する回路ではありません。



Target detection frequencies: How to calculate the subharmonic and harmonic of f_{OSCH} (assuming the guaranteed oscillation frequency error is ± 1%)

Sub-harmonic (allowing for the maximum high-frequency oscillator error)
= 24 MHz × (1+0.01) ÷2 = 12.12 MHz
Harmonic (allowing for the minimum high-frequency oscillation error)
= 24 MHz × (1-0.01) ×2 = 47.52 MHz

The following shows the recommended configuration when f_{OSCH} = 24 MHz and 10MHz (assuming the oscillation frequency error and the OFD capable of detecting are 10% totally).

f_{OSCH}	Upper: Sub Harmonic	Upper: Recommended CLKSMN value	Upper: Detects frequency range (Low)
	Lower: Harmonic	Lower: Recommended CLKSMX value	Lower: Detects frequency range (High)
24MHz	12.12 MHz	0xA5	4.0 MHz to 21.6MHz or Stops
	47.52 MHz	0xC9	26.4 MHz to 60 MHz (MAX)
10MHz	5.05 MHz	0x45	4.0 MHz to 9.0MHz or Stops
	19.8 MHz	0x54	11.0 MHz to 60 MHz (MAX)

Note: Regarding the operating specification of the high frequency oscillator, refer to the chapter 4 of Electrical Characteristics.

3.26.4 Register List

Base address = 0xF009_0000

Register Name	Address (base+)	Description
CLKSCR1	0x0000	Oscillation frequency detection control register 1
CLKSCR2	0x0004	Oscillation frequency detection control register 2
CLRSCR3	0x0008	Oscillation frequency detection control register 3
CLKSMN	0x0010	Lower detection frequency setting register
CLKSMX	0x0020	Higher detection frequency setting register

1. CLKSCR1 (Oscillation frequency detection control register 1)

Address = (0xF009_0000)+(0x0000)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	CLKWEN	R/W	0x06	OFD register write enable code: 0x06: Disable writing to CLKSCR2, CLKSCR3, CLKSMN and CLKSMX 0xF9: Enable writing to CLKSCR2, CLKSCR3, CLKSMN and CLKSMX Others: Reserved (Note 1)

[Explanation]

a. < CLKWEN >

0x06: Disable writing to the CLKSCR2, CLKSCR3, CLKSMN and CLKSMX registers

0xF9: Enable writing to the CLKSCR2, CLKSCR3, CLKSMN and CLKSMX registers

Note 1: Only “ 0x06 ” and “ 0xF9 ” can be written to CLKSCR1. All values other than “0xF9” are handled as “0x06”, disabling writing to CLKSCR2, CLKSCR3, CLKSMN and CLKSMX.

2. CLKSCR2 (Oscillation frequency detection control register 2)

Address = (0xF009_0000)+(0x0004)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	CLKSEN	R/W	0x00	OFD operation enable code: 0x00: Disable 0xE4: Enable Others: Invalid (Note 1)

[Explanation]

a. < OFDEN >

0x00: Enable the OFD operation

0xE4: Disable the OFD operation

Note 1: Only “0x00” and “0xE4” can be written to CLKSCR2. Writing a value other than “0x00” and “0xE4” to CLKSCR2 is invalid (the register value cannot be changed.)

Note 2: When the disable code “0x06” is written to CLKSCR1, writing to CLKSCR2 is disabled and any write attempts are ignored. Even when write operation is disabled, CLKSCR2 can be read.

Note3: It takes 2 fs cycles in maximums until the configuration is executed after writing into register. So when writing it in this register once, other configurations should be set after passing 2 fs cycles. The Enable/Disable status flag is not available. The read data value and the operating state are not much in time-lag period.

3. CLKSCR3 (Oscillation frequency detection control register 3)

Address = (0xF009_0000)+(0x0008)

Bit	Bit Symbol	Type	Reset Value	Description
[31:2]	–	–	Undefined	Read as undefined. Write as zero.
[1]	RESEN	R/W	0y0	OFD reset enable: 0y0: Disable 0y1: Enable
[0]	CLKSF	R/W	0y0	High speed oscillation frequency detection flag: Read: 0y0: OSC normal 0y1: OSC abnormal Write: 0y0: Invalid 0y1: Clear the flag to ”0”

[Explanation]

a. <RESEN>

0y0: Disable OFD reset.

0y1: Enable OFD reset. When an abnormal condition is detected, an internal reset is generated.

Note: This RESEN bit takes 2 fs cycles in maximums until the configuration is executed after writing into this bit. This bit can be poling whether or not the configulation is set and after that the other register setting can be done.

b. <CLKSF>

Read

0y0: The frequency of the high-frequency oscillation clock is within the specified range.

0y1: The frequency of the high-frequency oscillation clock is outside the specified range.

<CLKSF>=0y1 remains set until it is cleared. Even when the high-frequency oscillation clock frequency returns to the specified range.

.

Write

0y0: Invalid

0y1: Clear the flag to “0”

4. CLKSMN (Lower detection frequency setting register)

Address = (0xF009_0000)+(0x0010)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	LDFS	R/W	0xA5	Low detection frequency setting :

5. CLKSMX (Higher detection frequency setting register)

Address = (0xF009_0000)+(0x0020)

Bit	Bit Symbol	Type	Reset Value	Description
[31:8]	–	–	Undefined	Read as undefined. Write as zero.
[7:0]	HDFS	R/W	0xC9	High detection frequency setting :

[Explanation]

<LDFS> and < HDFS >

These registers are for detection frequency setting value.

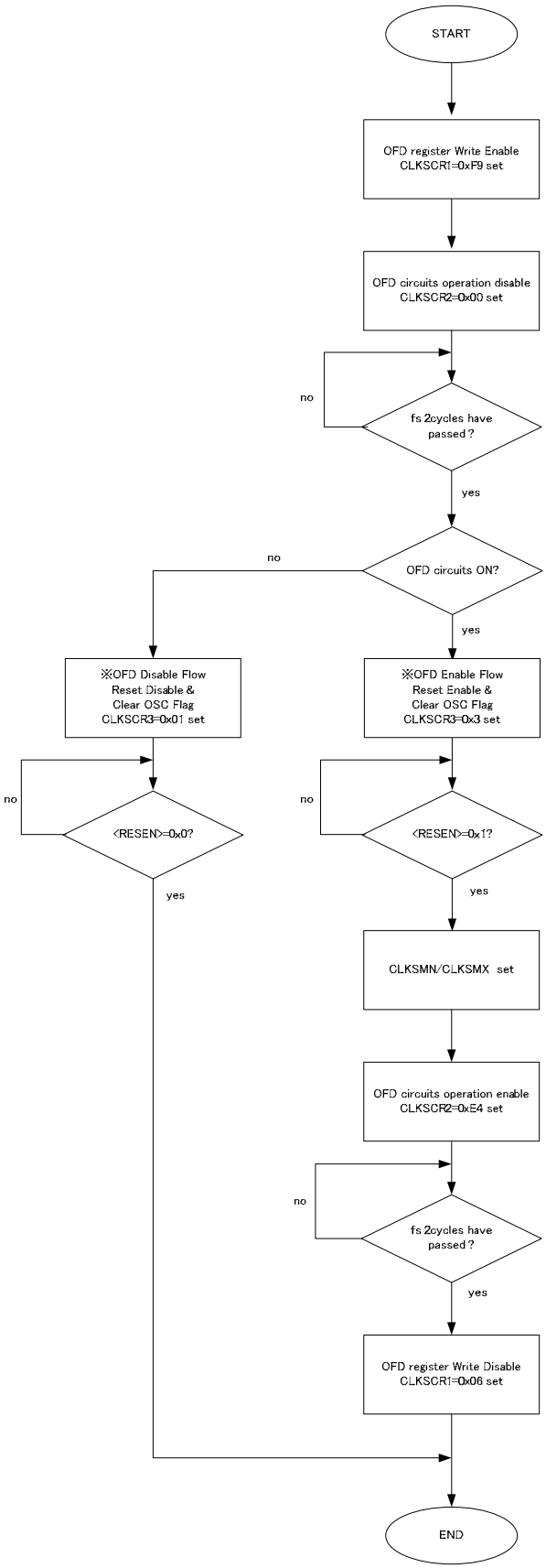
Note 1: CLKSMN and CLKSMX cannot be written while the frequency detection operation is enabled (CLKSCR2 = “0xE4”) or writing to the OFD registers is disabled (CLKSCR1="0x06").

Note 2: CLKSMN and CLKSMX are protected from write operation by writing “0x06” to CLKSCR1. These registers can be read regardless of the setting of CLKSCR1.

Note 3: The values to be set to CLKSMN and CLKSMX should be determined depending on the clock frequencies to be used to satisfy the condition CLKSMN < CLKSMX. For how to calculate the CLKSMN and CLKSMX set values, see examples in 3.26.3 “Description of Operation”.

Note 4: The setting of CLKSMN and CLKSMX should be set with enough safe including several percent error. Otherwise the internal reset might be always assert into the CPU. (Deadlock state)

3.26.5 Programming example



1) Programming example of OFD operation enable/disable

-- Enable setting example --

```
CLKSCR1=0xF9                    ; OFD register Write Enable
        ↓
CLKSCR2=0x00                    ; OFD circuits operation disable
        ↓
  Wait fs 2cycles
        ↓
CLKSCR3=0x03                    ; RESEN Enable & Clear OSC Flag
        ↓
Poling until the RESEN=1 state  ; Check the RESEN Bit status
        ↓
Set the CLKSMN and the CLKSMX registers
        ↓
CLKSCR2=0xE4                    ; OFD circuits operation enable
        ↓
  Wait fs 2cycles
        ↓
CLKSCR1=0x06                    ; OFD register Write Disable
```

-- Disable setting example --

```
CLKSCR1=0xF9                    ; OFD register Write Enable
        ↓
CLKSCR2=0x00                    ; OFD circuits operation disable
        ↓
  Wait fs 2cycles
        ↓
CLKSCR3=0x01                    ; RESEN Disable
        ↓
Poling until the RESEN=0 state  ; Check the RESEN Bit status
        ↓
CLKSCR1=0x06                    ; OFD register Write Disable
```

Note: In the PCM mode, the OFD circuitry is not powered and is not operational. So stop the OFD operation before the PCM execution.

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
DVCC3IO DVCCM	Power supply voltage	-0.3 to 3.8	V
DVCC1A DVCC1B DVCC1C		-0.3 to 2.0	
AVCC3AD AVDD3T AVDD3C AVCC3H		-0.3 to 3.8	
V_{IN}	Input voltage	-0.3 to DVCC3IO+0.3 (Note 1) -0.3 to DVCCM+0.3 (Note 2) -0.3 to AVCC3AD+0.3 (Note 3) -0.3 to AVDD3T+0.3 (Note 4) -0.3 to AVDD3C+0.3 (Note 4) -0.3 to AVCC3H+0.3	V
I_{OL}	Output current (per pin)	5	mA
I_{OH}	Output current (per pin)	-5	mA
ΣI_{OL}	Output current (total)	80	mA
ΣI_{OH}	Output current (total)	-80	mA
P_D	Power consumption (Ta = 85°C)	800	mW
T_{SOLDER}	Soldering temperature (10s)	260	°C
T_{STG}	Storage temperature	-65 to 150	°C
T_{OPR}	Operating temperature	-20 to 85	°C

Note 1: Do not exceed the absolute maximum rating of DVCC3IO (SM2-4, SM6,SM7, SN0-2, SP0-5, PA0-3, PB0-3, PC2-4, PC6-7, PN0, PN1, PT0-7, PU0-7, PV0-7)

Note 2: Do not exceed the absolute maximum rating of DVCCM (SA0-7, SB0-7, SE0-7, SF0-7, SG0-7, SH2,SH3,SH4,SH7, SJ0-6, SK0, SK1, SK4, SK5, SL0-2, SL4-SL6)

Note 3: For PD4-7, VREFH, VREFL the absolute maximum rating of AVCC3AD is applied.

Note 4: For the USB, DDM and DDP- pins, the absolute maximum rating of AVCC3T/3C is applied.

Note 5: The absolute maximum ratings are rated values that must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, the device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when desigining products that include this device, ensure that no absolute maximum rating value will ever be exceeded.

Solderability

Test parameter	Test condition	Note
Solderability	Use of Sn-37Pb solder Bath Solder bath temperature = 230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	Use of Sn-3.0Ag-0.5Cu solder bath Solder bath temperature = 245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	

4.2 DC Electrical Characteristics

Operating Voltage

Symbol	Parameter	Min	Typ	Max	Unit	Condition	
DVCC3IO	General I/O Power Supply Voltage (DVCC3IO) (DVSSCOMx = AVSS = 0V)	3.0	3.3	3.6	V	X1 = 10 to 27MHz CPU CLK (to 200MHz)	XT1 = 30 to 34kHz
DVCCM_1	Memory I/O Power	3.0	3.3	3.6			
DVCCM_2	Memory I/O Power	1.7	1.8	1.9			
AVCC3AD	ADC Power	3.0	3.3	3.6			
AVDD3T/3C	USB Device Power	3.15	3.3	3.45			
AVCC3H	USB Host Power	3.0	3.3	3.6			
DVCC1A	Internal Power A	1.4	1.5	1.6			
DVCC1B	Internal Power B						
DVCC1C	High CLK oscillator and PLL Power						

It is assumed that all power supply pins of the same rail are electrically connected externally and are supplied with the equal voltage.

Note: The power of I2S function is supplied by DVCC3IO.

Input Voltage (1)

Symbol	Parameter	Min	Typ	Max	Unit	Condition
VIL0	Input Low Voltage for SM2, SM6-7, SN0-2, SP0-3 PA0-3, PT0-7	-0.3	–	0.3 × DVCC3IO	V	3.0 ≤ DVCC3IO ≤ 3.6V
VIL1	Input Low Voltage for SA0-7, SB0-7, SL4-6		–	0.3 × DVCCM		3.0 ≤ DVCCM ≤ 3.6V SELDVCCM = 1
VIL2			–	0.3 × DVCCM		1.7 ≤ DVCCM ≤ 1.9V SELDVCCM = 0
VIL5(Note)	Input Low Voltage for PD4-7		–	0.3 × AVCC3AD		3.0 ≤ AVCC3AD ≤ 3.6V
VIL6	Input Low Voltage for SM4, PC6, PC7, PN0, PN1		–	0.25 × DVCC3IO		3.0 ≤ DVCC3IO ≤ 3.6V

Note: When Ports PD4 to PD7 are used as general-purpose inputs.

Input Voltage (2)

Symbol	Parameter	Min	Typ	Max	Unit	Condition
VIH0	Input High Voltage for SM2, SM6-7, SN0-2, SP0-3 PA0-3, PT0-7	0.7 × DVCC3IO	–	DVCC3IO + 0.3	V	3.0 ≤ DVCC3IO ≤ 3.6
VIH1	Input High Voltage for SA0-7, SB0-7, SL4-6	0.7 × DVCCM	–	DVCCM + 0.3		3.0 ≤ DVCCM ≤ 3.6
VIH2		0.7 × DVCCM	–	DVCCM + 0.3		1.7 ≤ DVCCM ≤ 1.9
VIH5 (Note)	Input High Voltage for PD4-7	0.7 × ADCC3AD	–	AVCC3AD + 0.3		3.0 ≤ AVCC3AD ≤ 3.6
VIH6	Input High Voltage for SM4, PC6, PC7, PN0, PN1	0.75 × DVCC3IO	–	DVCC3IO + 0.3		3.0 ≤ DVCC3IO ≤ 3.6V

Note: When Ports PD4 to PD7 are used as general-purpose inputs.

Output Voltage (1)

Symbol	Parameter	Min	Typ	Max	Unit	Condition
VOL0	Output Low Voltage for SM3, SP4, SP5, PB0-3, PC2-4, PC6, PC7, PN0, PN1, PT0-7	–	–	0.4	V	IOL = 2.0 mA 3.0 ≤ DVCC3IO ≤ 3.6V
VOL1	Output Low Voltage for SA0-7, SB0-7, SE0-7, SF0-7, SG0-7, SH2-4, SH7, SJ0-6, SK0-1, SK4-5, SL0-2, SL4-5					IOL = 2.0 mA 3.0 ≤ DVCCM ≤ 3.6V
VOL2						IOL = 2.0 mA 1.8 ≤ DVCCM ≤ 1.9V
VOL5	Output Low Voltage for PD4-7					IOL = 2.0 mA 3.0 ≤ AVCC3AD ≤ 3.6V

Output Voltage (2)

Symbol	Parameter	Min	Typ	Max	Unit	Condition
VOH0	Output High Voltage for SM3, SP4, SP5, PB0-3, PC2-4, PC6, PC7, PN0, PN1, PT0-7	DVCC3IO－0.4	–	–	V	IOH = -1.0 mA 3.0 ≤ DVCC3IO ≤ 3.6V
VOH1	Output High Voltage for SA0-7, SB0-7, SE0-7, SF0-7, SG0-7, SH2-4, SH7, SJ0-6, SK0-1, SK4-5, SL0-2, SL4-5	DVCCM－0.4				IOH = -1.0 mA 3.0 ≤ DVCCM ≤ 3.6V
VOH2		DVCCM－0.4				IOH = -1.0 mA 1.8 ≤ DVCCM ≤ 1.9V
VOH5	Output High Voltage for PD4-7	AVCC3AD－0.4				IOH = -1.0 mA 3.0 ≤ AVCC3AD ≤ 3.6V

Others

Symbol	Parameter	Min	Typ	Max	Unit	Condition	
IMon	Internal resistor (ON) MX, MY pins	–	–	30	Ω	VOL = 0.2V	3.0 ≤ AVCC3AD ≤ 3.6V
IMon	Internal resistor (ON) PX, PY pins	–	–	30		VOH = AVCC3AD – 0.2V	
ILI	Input Leakage Current	–	0.02	±5	μA		
ILO	Output Leakage Current	–	0.05	±10	μA		
R	Pull Up/Down Resistor for RESETn, PA0-3, PD6	30	50	70	kΩ		
CIO	Pin Capacitance	–	1.0	–	pF	fc = 1MHz	
VTH	Schmitt Width for SM4, PA0-3, PD6, PD7, PC6, PC7, PN0-1, PT4-6	–	0.6	–	V	3.0 ≤ DVCC3IO ≤ 3.6V	

Note 1: Typical values show those with Ta = 25 °C and DVCC3IO = 3.3 V,
DVCCM = 3.3V or DVCCM = 1.8V、DVCC1A,1B,1C = 1.5V unless otherwise noted.

Note 2: The above values do not apply when debug mode is used.

Symbol	Parameter	Min	Typ	Max	Unit	Condition		
ICC	NORMAL (note2)	–	1.8	3.0	mA	PLL_ON f_{FCLK} = 200MHz Ta ≤ 70°C	DVCC3IO = 3.6V	
		–	10	12			DVCC3LCD = 3.6V	
		–	5	7			AVCC3AD = 3.6V	
		–	50	–			AVDD3T = 3.3V AVDD3C = 3.3V	
		–	0.7	2.0			AVCC3H = 3.3V	
		–	28	36			DVCCM = 3.6V	Only SDRAM access
			13	17				Only NORF access
		–	15	19			DVCCM = 1.9V	Only SDRAM access
			7	9				Only NORF access
		–	136	206			DVCC1A = 1.6V DVCC1B = 1.6V DVCC1C = 1.6V	
		–	105	205		PLL_ON f_{FCLK} = 150MHz Ta ≤ 85°C	DVCC1A = 1.6V DVCC1B = 1.6V DVCC1C = 1.6V	
	CPU HALT	–	6.1	10	mA	PLL_OFF f_{FCLK} =25MHz	DVCC3IO = 3.6V DVCCM = 3.6V DVCC3LCD = 3.6V AVCC3AD = 3.6V AVDD3T = 3.6V AVDD3C = 3.6V AVCC3H = 3.6V	
		–	16	116			DVCC1A = 1.6V DVCC1B =1.6V DVCC1C = 1.6V	

Operationg Conditions:

NORMAL

CPU: DRYSTONE rev2.1

Instruction Cache : ON, Data Cache: ON

Program Execution area : internal RAM, Data area: internal RAM, Stack area: internal RAM

USB: Default condition

LCDC: HVGA_16bpp

A/DC: 5μs repeat conversion

I^2S, CMS: Stop

SDHC: Stop

UART: 480kbps transmission

SSP: 100kbps transmission

PWM: 100kHz output

SDRAM: 100MHz CL=2 BL=8 32bit bus Read/Write (External bus start operation duty is approximately 17%)

Or NORF: asynchronous NORF 160ns access 16bit bus continuous read

CPU HALT

CPU: HALT, Peripheral Circuit: TSB original program

USB: Suspend

LCDC: Simplicity operation (VRAM: Internal RAM)

A/DC, I^2S, CMS, UART, SSP, PWM, External memory access: No operarion

Note 1: Typical values show those with Ta = 25°C, DVCC3IO = AVCC3H = 3.3V,
DVCCM = 3.3V or DVCCM = 1.8V, DVCC1A,1B,1C = 1.5V unless otherwise noted.

Note 2: IC measurement conditions:
CL = 25 pF for bus pins, other output pins = open, input pins = level fixed

Note 3: The above values do not apply when debug mode is used.

Symbol	Parameter	Min	Typ	Max	Unit	Condition	
ICC	Power Cut Mode (With PMC function)	–	4.5	100	μA	Ta ≤ 85°C	DVCC3IO = 3.6V DVCCM = 3.6V AVCC3AD = 3.6V AVDD3T = OPEN AVDD3C = OPEN AVCC3H = OPEN
				25		Ta ≤ 70°C	
				15		Ta ≤ 50°C	
			1.5	230		Ta ≤ 85°C	DVCC1A = 0V DVCC1B = 1.6V, DVCC1C = 0V XT = 32kHz X1,X2 = OFF
				135		Ta ≤ 70°C	
				70		Ta ≤ 50°C	

Note 1: Typical values show those with Ta = 25°C, DVCC3IO = 3.3 V,

DVCCM = 3.3V or DVCCM = 1.8V, DVCC1A,1B,1C = 1.5V, unless otherwise noted.

Note 2: IC measurement conditions:

CL = 50 pF for bus pins, other output pins = open, input pins = level fixed

(Operating at 8-wait access for external memory)

Note 3: The above values do not apply when debug mode is used.

4.3 AC Electrical Characteristics

All AC specifications shown below are the measurement results under the following conditions unless specified otherwise.

AC measurement conditions

- The letter "T" used in the equations in the table represents the period of internal bus frequency (f_{HCLK}) which is one-half of the CPU clock (f_{FCLK}).
- Output level: High = 0.7 × DVCCM, Low = 0.3 × DVCCM
- Input level: High = 0.9 × DVCCM, Low = 0.1 × DVCCM

Note: The "Equation" column in the table shows the specifications under the conditions DVCCM=1.7 to 1.9 V or DVCCM=3.0 to 3.6 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

4.3.1 Basic Bus Cycles

Read cycle (asynchronous mode)

No.	Parameter	Symbol	Equation		100 MHz N=10 M=3 K=10 L=6	96 MHz N=10 M=3 K=10 L=6	48 MHz N=5 M=1 K=5 L=3	Unit
			Min	Max				
1	Internal bus period (= T) Note)	t_{CYC}	10	800	10.0	10.4	20.8	ns
2	A0-A23 valid to D0-D15 input	t_{AD}	(N)T – 15.0		85.0	89.2	89.2	
3	SMCOEn fall to D0-D15 input	t_{OED}	(N-M)T – 10.0		60.0	62.8	73.3	
4	SMCOEn low level pulse width	t_{OEW}	(N-M)T – 8.0		62.0	64.9	75.3	
5	A0-A23 valid to SMCOEn fall	t_{AOE}	MT – 5.0		25	26.3	15.8	
6	SMCOEn rise to D0-D15 hold	t_{HR}	0		0	0	0	
7	SMCOEn high level pulse width	t_{OEHW}	MT – 8.0		22	23.3	12.8	

Write cycle (asynchronous mode)

8	D0-D15 valid to SMCWEn rise	t_{DW}	(L+1)T – 10.0		60.0	62.9	73.3	ns
9	D0-D15 valid to SMCWEn rise (bls=1)	t_{SDS}	(L+1)T – 10.0		60.0	62.9	73.3	
10	SMCWEn low level width	t_{WW}	LT – 8.0		52.0	54.5	54	
11	A0-A23 valid to SMCWEn fall	t_{AW}	T – 5.0		5.0	5.4	15.8	
12	SMCWEn rise to A0-A23 hold	t_{WA}	(K-L-1)T – 5.0		25.0	26.3	15.8	
13	SMCWEn rise to D0-D15 hold	t_{WD}	(K-L-1)T – 5.0		25.0	26.3	15.8	
14	SMCOEn rise to D0-D15 output	t_{OEO}	2		2.0	2.0	2.0	
15	Data byte control to write complete time	t_{SBW}	LT – 8.0		52.0	54.5	54.5	

- The variables used in the equations in the table are defined as follows:

  N = Number of t_{RC} cycles M = Number of t_{CEOE} cycles

  K = Number of t_{WC} cycles L = Number of t_{WP} cycles

Measuring Condition

Connection

1. DVCC3IO × 0.7 ≤ SELDVCCM ≤ DVCC3IO

Software configuration

1. PMCDRV<DRV_MEM1:0> = 0y11 (Full Drive at 1.8±0.1V)
2. PMCDRV<DRV_MEM1:0> = 0y01 (Half Drive at 3.3±0.3V)

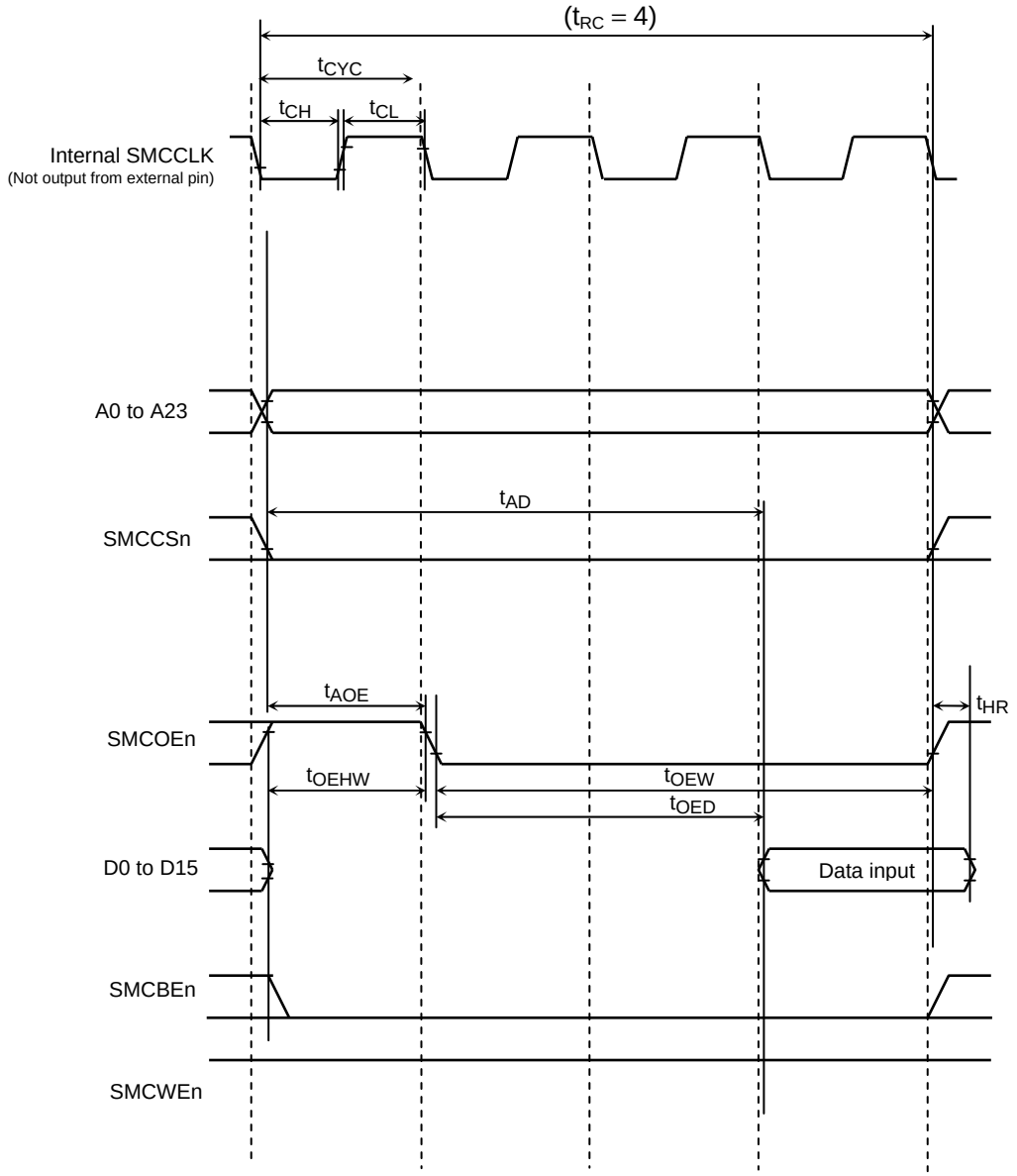
Loaded capacitance

CL (25 pF

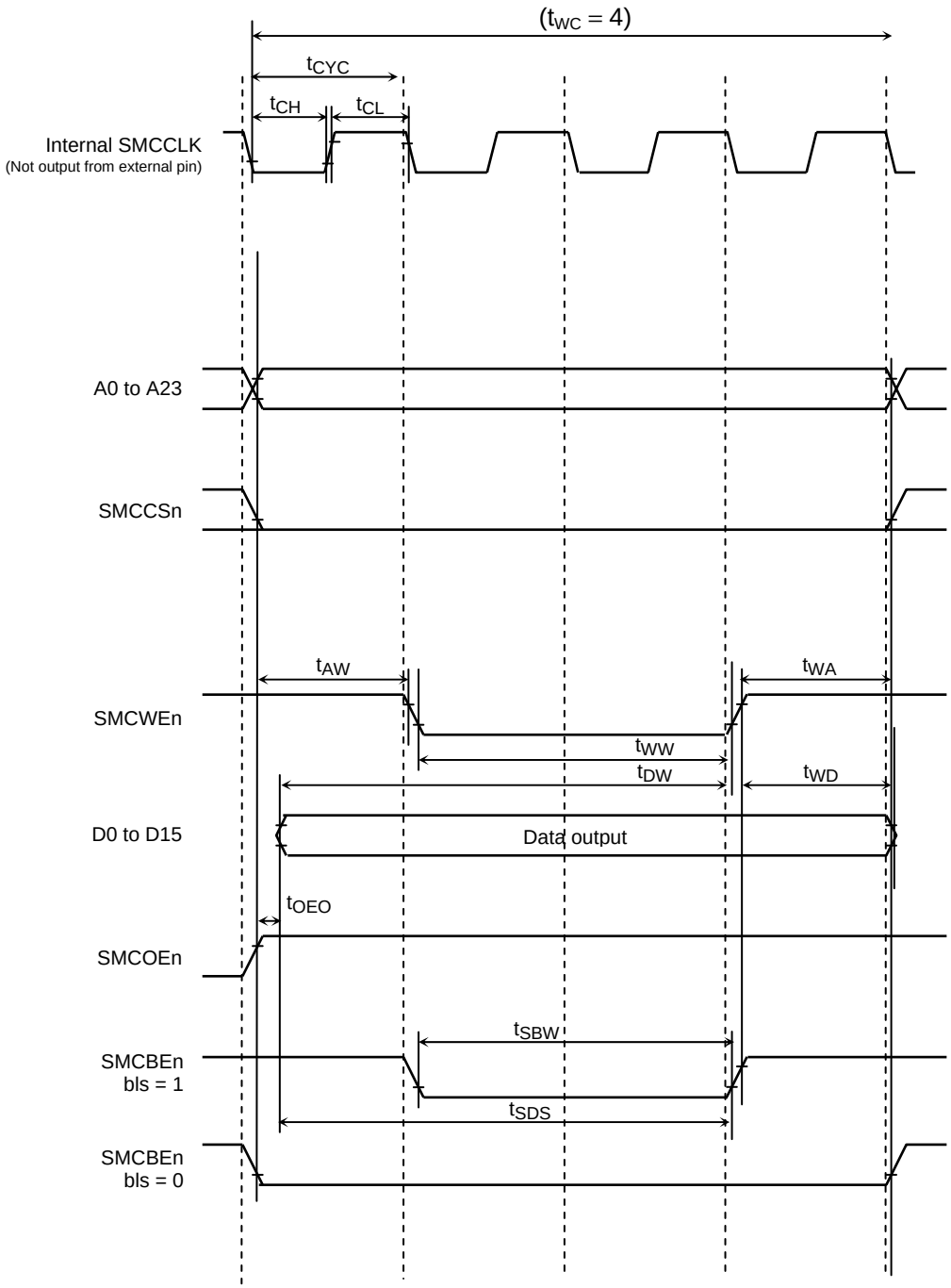
Note: The internal bus cycle is T=10ns minimum value when the guaranteed temperature is 0 to 70 degree.

The internal bus cycle is T=13.3ns minimum value when the guaranteed temperature is -20 to 85 degree.

(1) Asynchronous memory read cycle (tRC = 4, tCEOE = 1)



(2) Asynchronous memory write cycle (t_{WC} = 4, t_{WP} = 2)



4.3.2 DDR SDRAM Controller AC Electrical Characteristics

AC measurement conditions

- The letter "T" used in the equations in the table represents the period of internal bus frequency (f_{HCLK}), which is one-half of the CPU clock (f_{FCLK}).
- Output level: High = 0.7 × DVCCM, Low = 0.3 × DVCCM
- Input level: High = 0.9 × DVCCM, Low = 0.1 × DVCCM
- Clock output Differential level (VOD): VOD = 0.6 × DVCCM
- Clock output Differential Crosspoint level (VOX): High = 0.6 × DVCCM, Low = 0.4 × DVCCM

Note 1: Only DDR SDRAM devices of LVCMOS type are supported. DDR SDRAM devices of SSTIL (2.5 V) type are not supported.

Note 2: The "Equation" column in the table shows the specifications under the conditions DVCCM = 1.7 to 1.9 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

Symbol	Parameter	Min	Typ.	Max	Unit	Condition
VIX	AC Differential Cross point Voltage	0.4 × DVCCM		0.6 × DVCCM		1.7 ≤ DVCCM ≤ 1.9V

No.	Parameter	Symbol	Equation		100 MHz	96 MHz	Unit
			Min	Max			
1	DMCDCLKP/ DMCDCLKN cycle time Note)	t_{CK}	T		10	10.4	ns
2	DMCDCLKP&DMCDCLKN Clock Skew time		-0.35	0.35			
3	CLK Differential Crosspoint cycle	t_{CH}	0.5T - 0.5		4.5	4.7	
4	CLK Differential Crosspoint cycle	t_{CL}	0.5T - 0.5		4.5	4.7	
5	DMCDDQSx Access time from CLK(CL*=3)	t_{AC1}		2T-13.5	6.5	7.3	
6	Data Access time from CLK(CL*=3)	t_{AC2}		2T-13.5	6.5	7.3	
7	DQS to Data Skew time	t_{DQSQ}	0	0.7	0.7	0.7	
8	Address set-up time	t_{AS}	0.5T - 3.0		2.0	2.2	
9	Address hold time	t_{AH}	0.5T - 3.0		2.0	2.2	
10	CKE set-up time	t_{CKS}	0.5T - 3.0		2.0	2.2	
11	Command set-up time	t_{CMS}	0.5T - 3.0		2.0	2.2	
12	Command hold time	t_{CMH}	0.5T - 3.0		2.0	2.2	
13	Data Setup Time	t_{DS}	0.25T - 1.5		1.0	1.1	
14	Data Hold Time	t_{DH}	0.25T - 1.5		1.0	1.1	
15	DMCDDM Setup Time	t_{MS}	0.25T - 1.5		1.0	1.1	
16	DMCDDM Hold Time	t_{MH}	0.25T - 1.5		1.0	1.1	
17	Write command to 1'st DQS Latching Transition	t_{DQSS}	0.75T	1.25T	7.50 to 12.5	7.80 to 13.0	

*CL = CAS latency

In case of DDR_SDRAM, CL number counting method is defferent with SDR_SDRAM.

Memory controller CL number = (DDR_SDRAM's CL Number) – 1

Note: The internal bus cycle is T=10ns minimum value when the guaranteed temperature is 0 to 70 degree.

The internal bus cycle is T=13.3ns minimum value when the guaranteed temperature is -20 to 85 degree.

Measuring Condition

Connection

1. DVCC3IO × 0.7 ≤ SELDVCCM ≤ DVCC3IO
2. DVCC3IO × 0.7 ≤ SELMEMC ≤ DVCC3IO
3. DMCCLKIN pin connect to DMCDCLKP

Software configuration

1. PMCDRV<DRV_MEM1:0> = 0y11 (Full Drive)
2. dmc_user_config_5 = 0x0000_0058

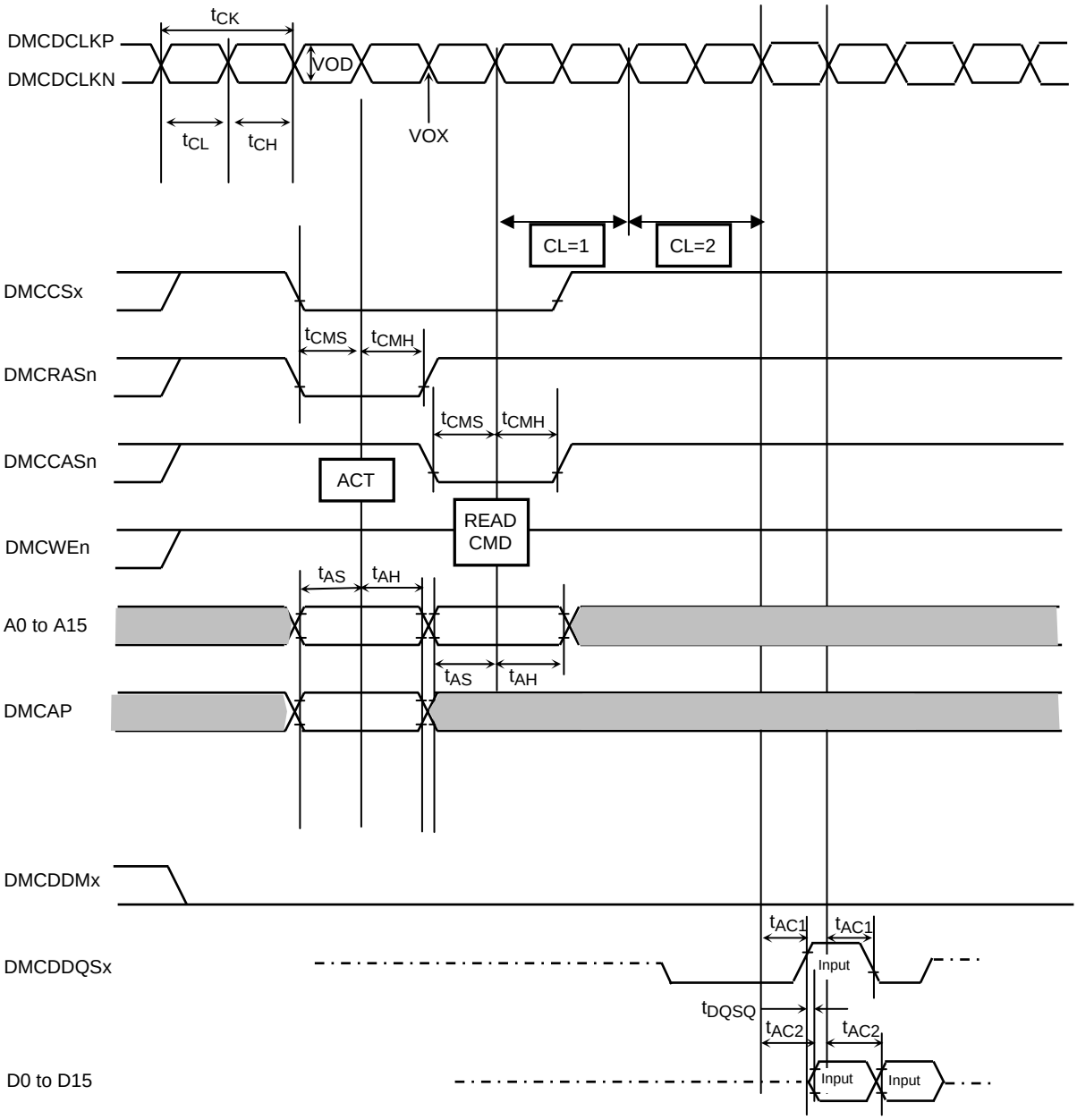
Loaded capacitance

DMCDCLKP pin and DMCDCLKN pin: CL = 15pF
Others: CL = 25 pF

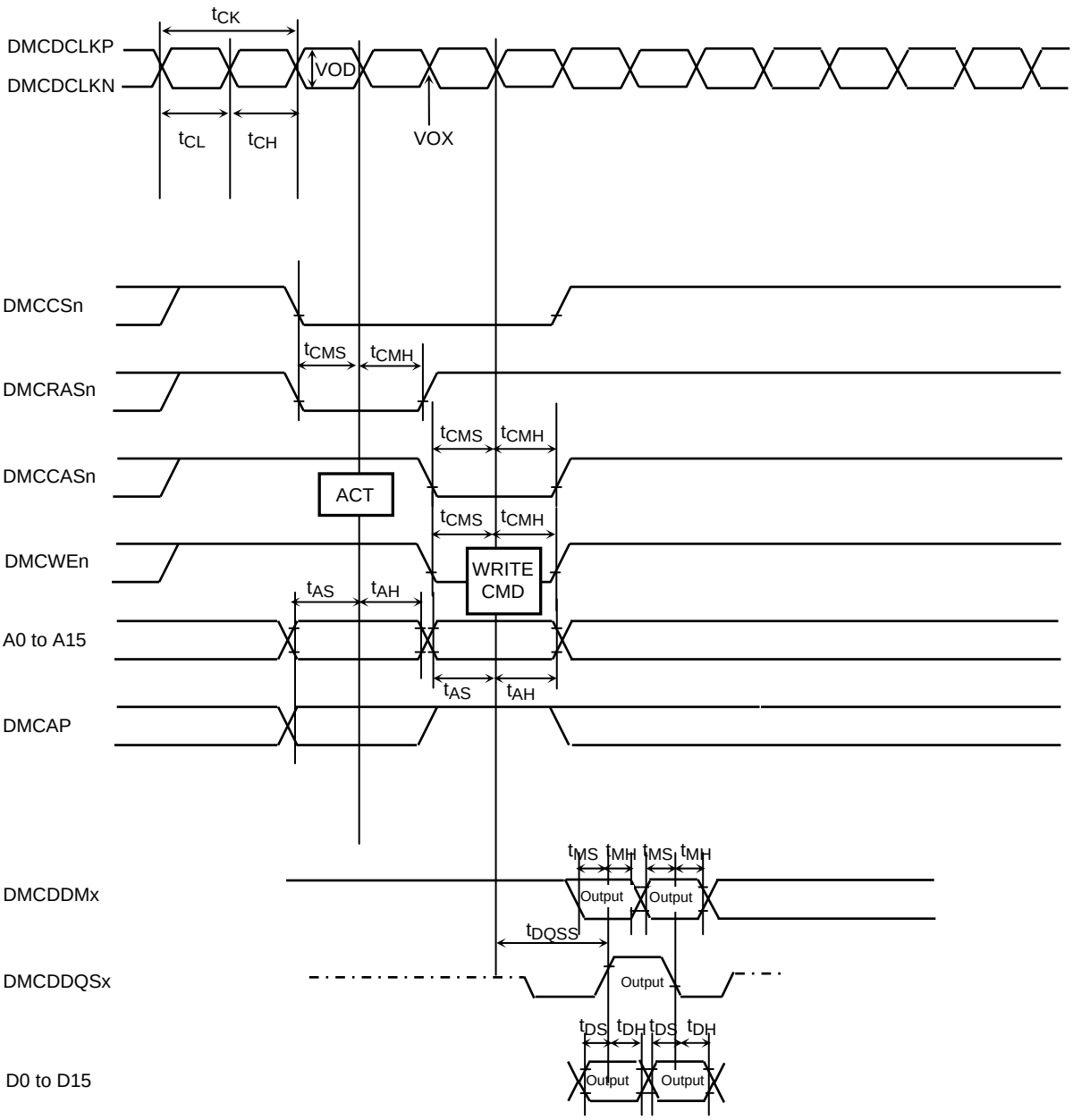
(1) DDR SDRAM read timing

(2-word read mode, Memory's CAS latency = 3

Memory controller's CAS latency = 2)



(2) DDR SDRAM write timing (2-word write mode)



4.3.3 Mobile SDR SDRAM Controller AC Electrical Characteristics

AC measurement conditions

- The letter “T” used in the equations in the table represents the period of internal bus frequency (f_{HCLK}), which is one-half of the CPU clock (f_{FCLK}).
- Output level: High = 0.7 × DVCCM, Low = 0.3 × DVCCM
- Input level: High = 0.9 × DVCCM, Low = 0.1 × DVCCM

Note: The “Equation” column in the table shows the specifications under the conditions DVCCM = 1.7 to 1.9 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

No.	Parameter	Symbol	Equation		100 MHz	96MHz	Unit
			Min	Max			
1	CLK cycle time	t_{CK}	T		10	10.4	ns
2	DMCSCLK high level width	t_{CH}	0.5T - 1.5		3.5	3.7	
3	DMCSCLK low level width	t_{CL}	0.5T - 1.5		3.5	3.7	
4	Access time from CLK(CL* = 2)	t_{AC}		T – 4.0	6.0	6.4	
5	Data hold time from internal read	t_{HR}	2.0		2.0	2.0	
6	Data setup time	t_{DS}	0.5T – 3.0		2.0	2.2	
7	Data hold time	t_{DH}	0.5T – 4.0		–	–	
8	Address setup time	t_{AS}	0.5T – 3.0		1.0	1.2	
9	Address hold time	t_{AH}	0.5T – 4.0		–	–	
10	CKE setup time	t_{CKS}	0.5T – 3.0		2.0	2.2	
11	Command setup time	t_{CMS}	0.5T – 3.0		1.0	1.2	
12	Command hold time	t_{CMH}	0.5T – 4.0		–	–	

*CL = CAS latency

Measuring Condition

Connection

1. DVSSCOMx ≤ SELDVCCM ≤ DVCC3IO × 0.3
2. DVSSCOMx ≤ SELMEMC ≤ DVCC3IO × 0.3
3. DMCCLKIN pin connect to DVSSCOMx

Software configuration

1. PMCDRV<DRV_MEM1:0> = 0y11 (Full Drive)
2. dmc_user_config_3 = 0x0000_0000 (16bit bus width memory)

Loaded capacitance

DMCSCLK pin: CL = 15pF
Others: CL = 25 pF

Note: The internal bus cycle is T=10ns minimum value when the guaranteed temperature is 0 to 70 degree.
The internal bus cycle is T=13.3ns minimum value when the guaranteed temperature is -20 to 85 degree.

4.3.4 SDR SDRAM Controller Electrical Characteristics

AC measurement conditions

- The letter “T” used in the equations in the table represents the period of internal bus frequency (f_{HCLK}), which is one-half of the CPU clock (f_{FCLK}).
- Output level: High = 0.7 × DVCCM, Low = 0.3 × DVCCM
- Input level: High = 0.9 × DVCCM, Low = 0.1 × DVCCM

Note: The “Equation” column in the table shows the specifications under the conditions DVCCM = 3.0 to 3.6 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

No.	Parameter	Symbol	Equation		100 MHz	96 MHz		Unit
			Min	Max				
1	CLK cycle time	t_{CK}	T		10	10.4		ns
2	DMCSCLK high level width	t_{CH}	0.5T – 1.5		3.5	3.7		
3	DMCSCLK low level width	t_{CL}	0.5T – 1.5		3.5	3.7		
4	Access time from CLK(CL* =2)	t_{AC}		T – 4.0	6.0	6.4		
5	Data hold time from internal read	t_{HR}	2.0		2.0	2.0		
6	Data set-up time	t_{DS}	0.5T – 3.0		2.0	2.2		
7	Data hold time	t_{DH}	0.5T – 4.0		1.0	1.2		
8	Address set-up time	t_{AS}	0.5T – 3.0		2.0	2.2		
9	Address hold time	t_{AH}	0.5T – 4.0		1.0	1.2		
10	CKE set-up time	t_{CKS}	0.5T – 3.0		2.0	2.2		
11	Command set-up time	t_{CMS}	0.5T – 3.0		2.0	2.2		
12	Command hold time	t_{CMH}	0.5T – 4.0		1.0	1.2		

*CL = CAS latency

Measuring condition

Connection

1. DVCC3IO × 0.7 ≤ SELDVCCM ≤ DVCC3IO
2. DVSSCOMx ≤ SELMEMC ≤ DVCC3IO × 0.3
3. DMCCLKIN pin connect to DVSSCOMx

Register

1. PMCDRV<DRV_MEM1:0> = 0y11 (Full Drive)
2. dmc_user_config3 = 0x0000_0000 (16bit bus width memory)

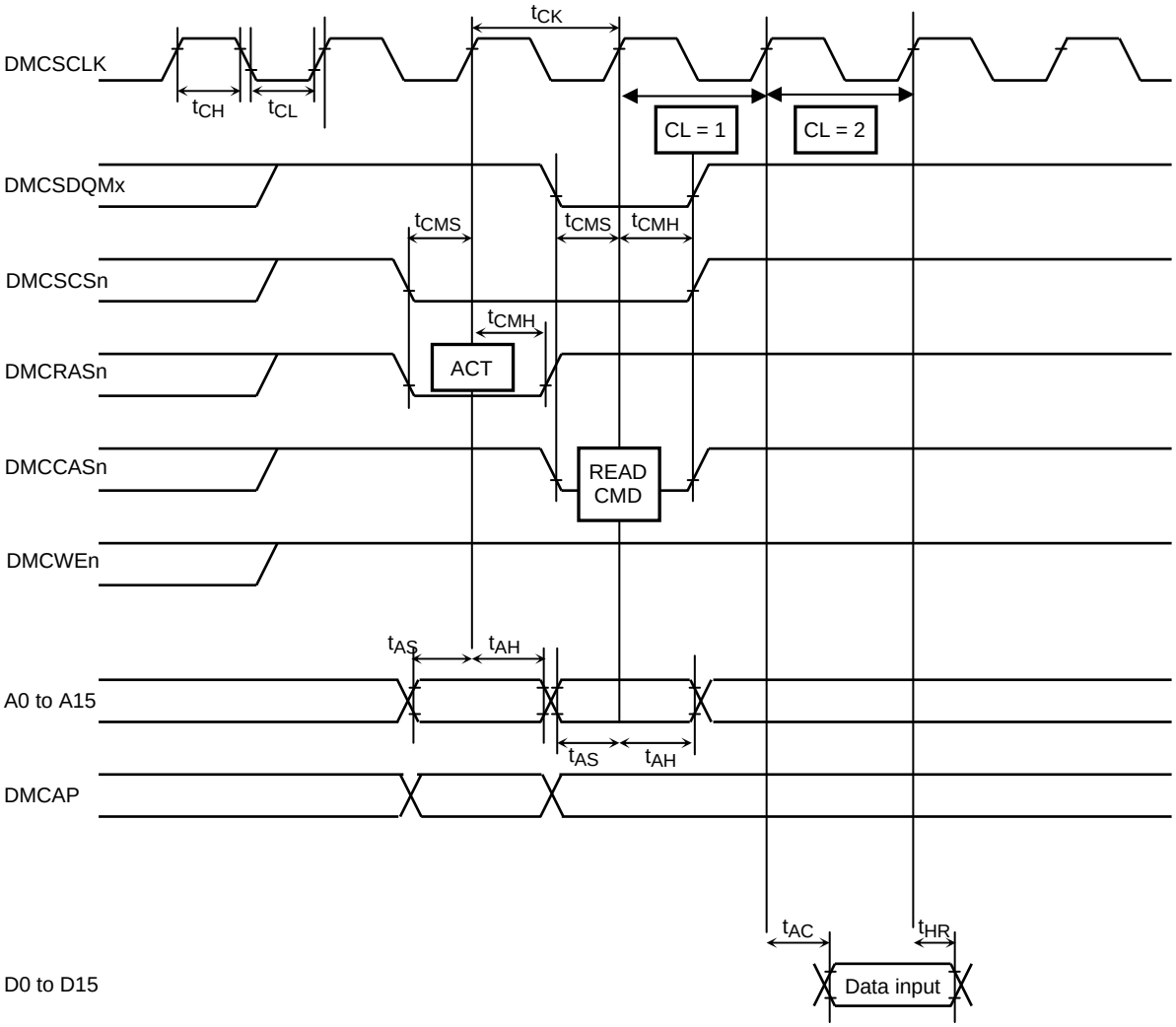
Loaded capacitance

DMCSCLK pin: CL = 15pF

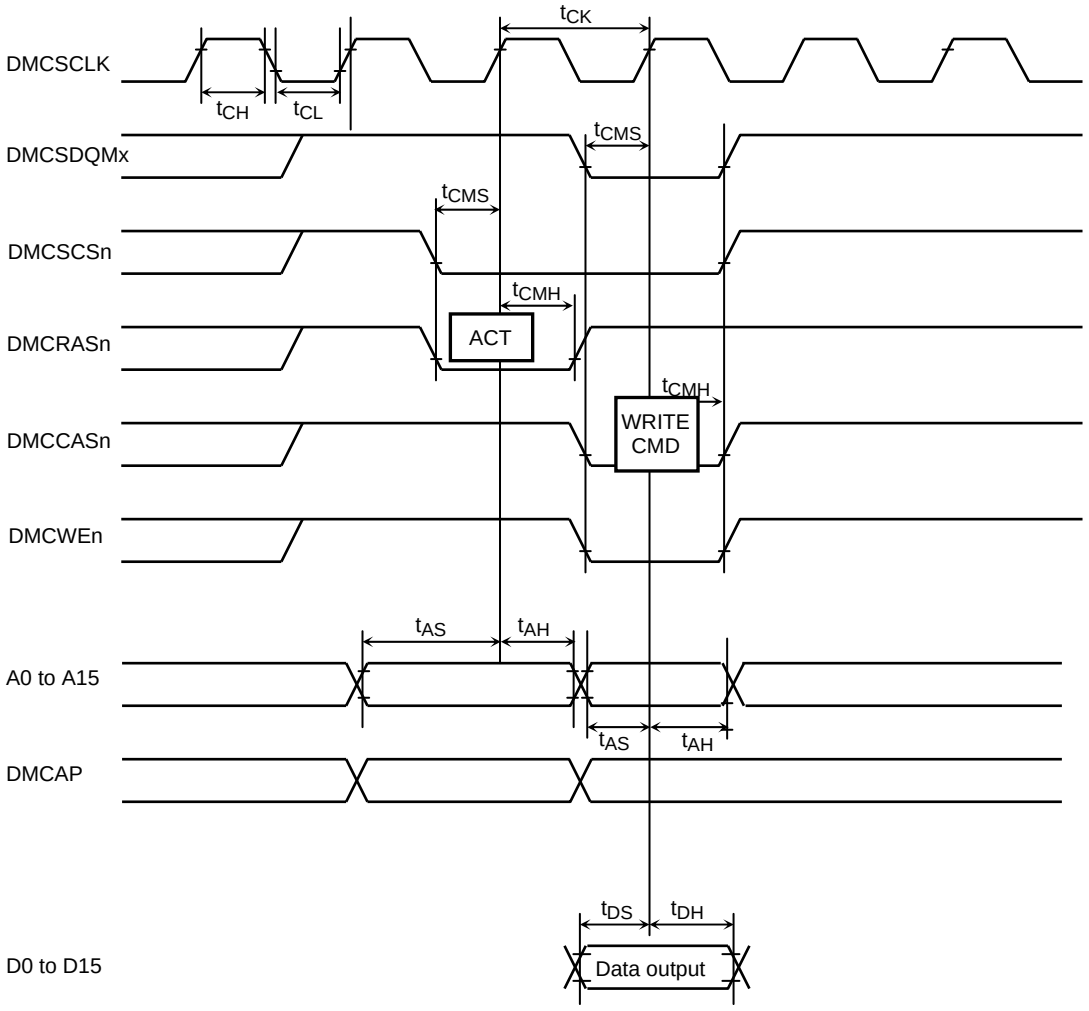
Others: CL = 25 pF

Note: The internal bus cycle is T=10ns minimum value when the guaranteed temperature is 0 to 70 degree.
The internal bus cycle is T=13.3ns minimum value when the guaranteed temperature is -20 to 85 degree.

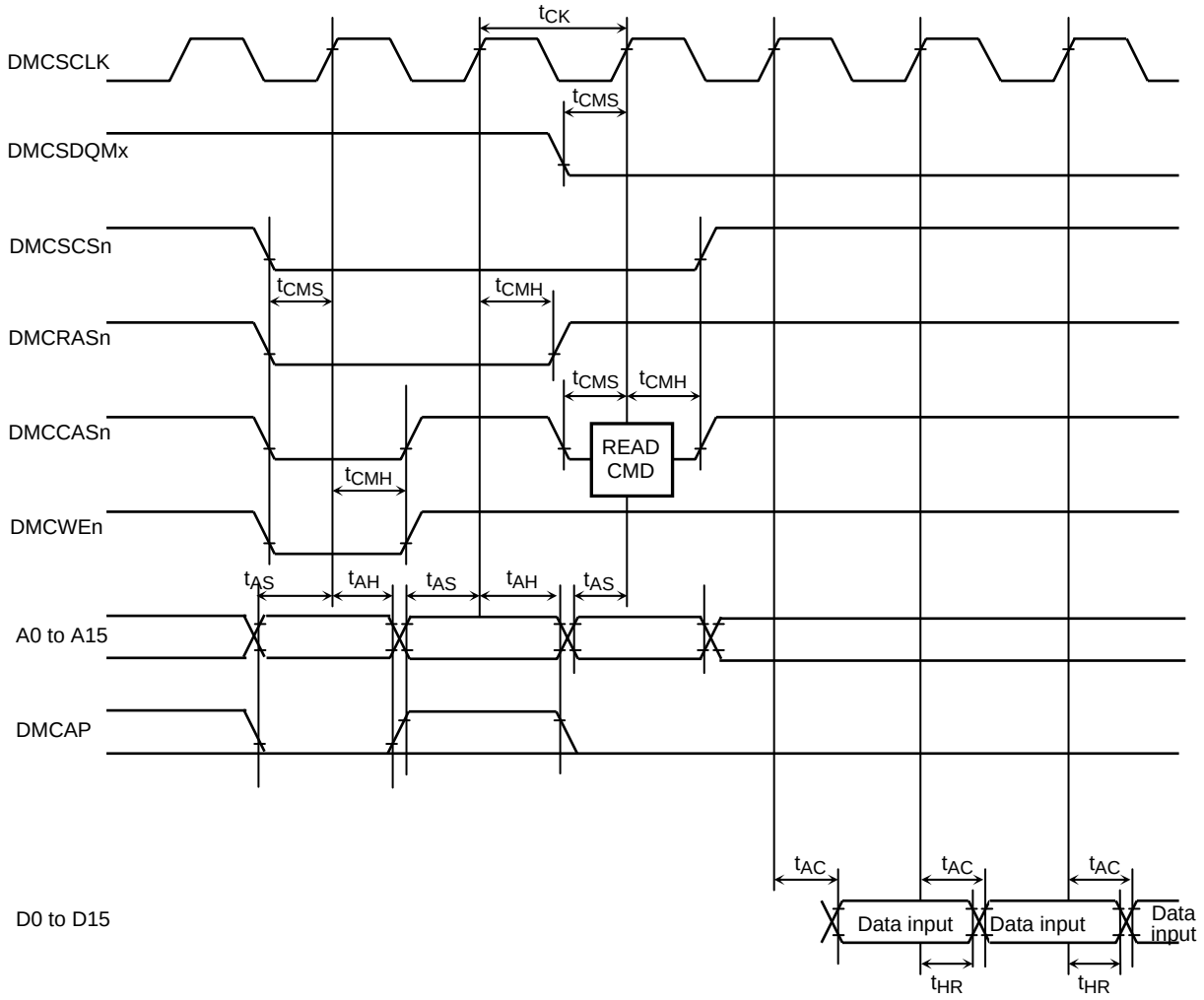
(1) SDRAM read timing (CAS latency = 2)



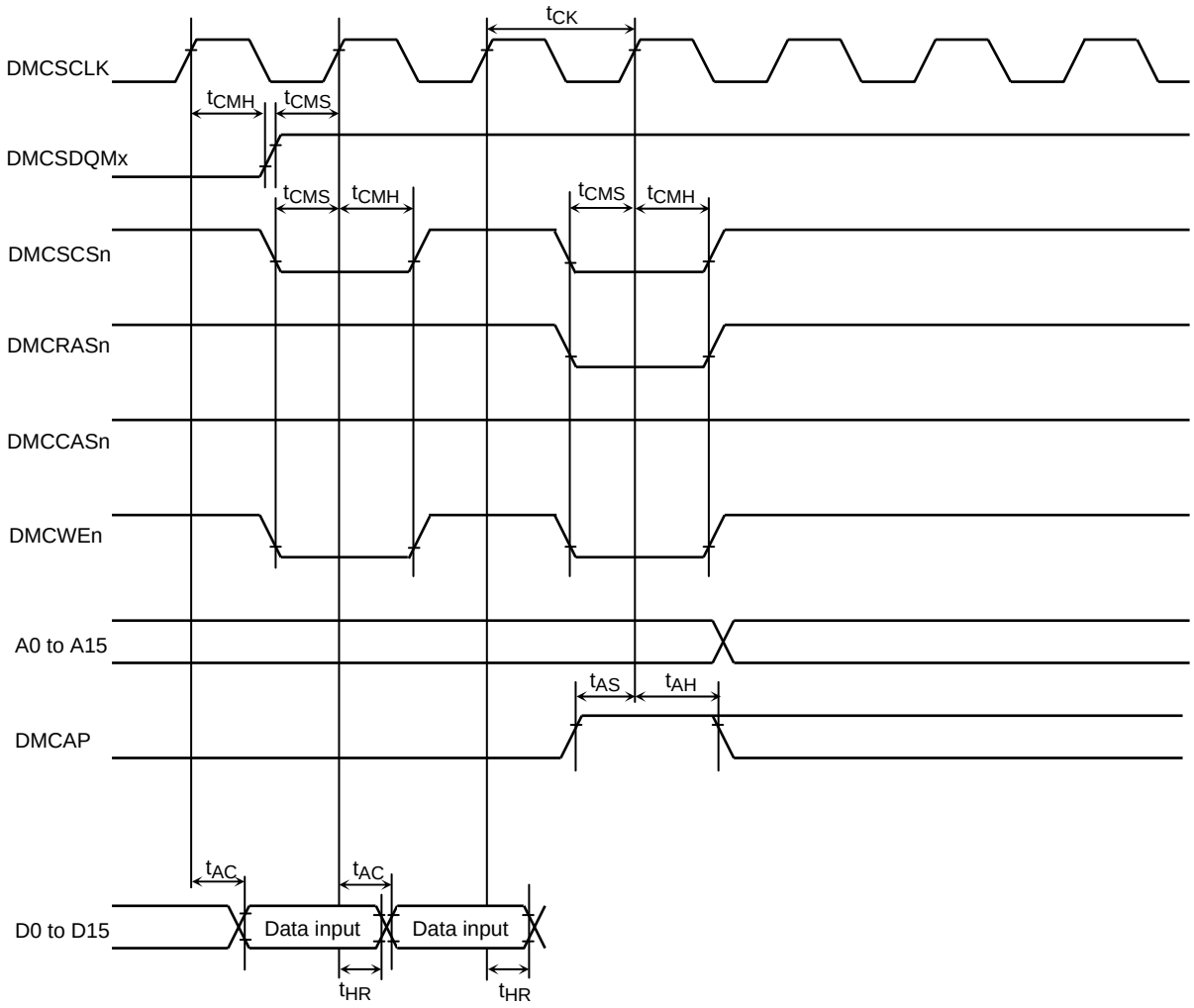
(2) SDRAM write timing



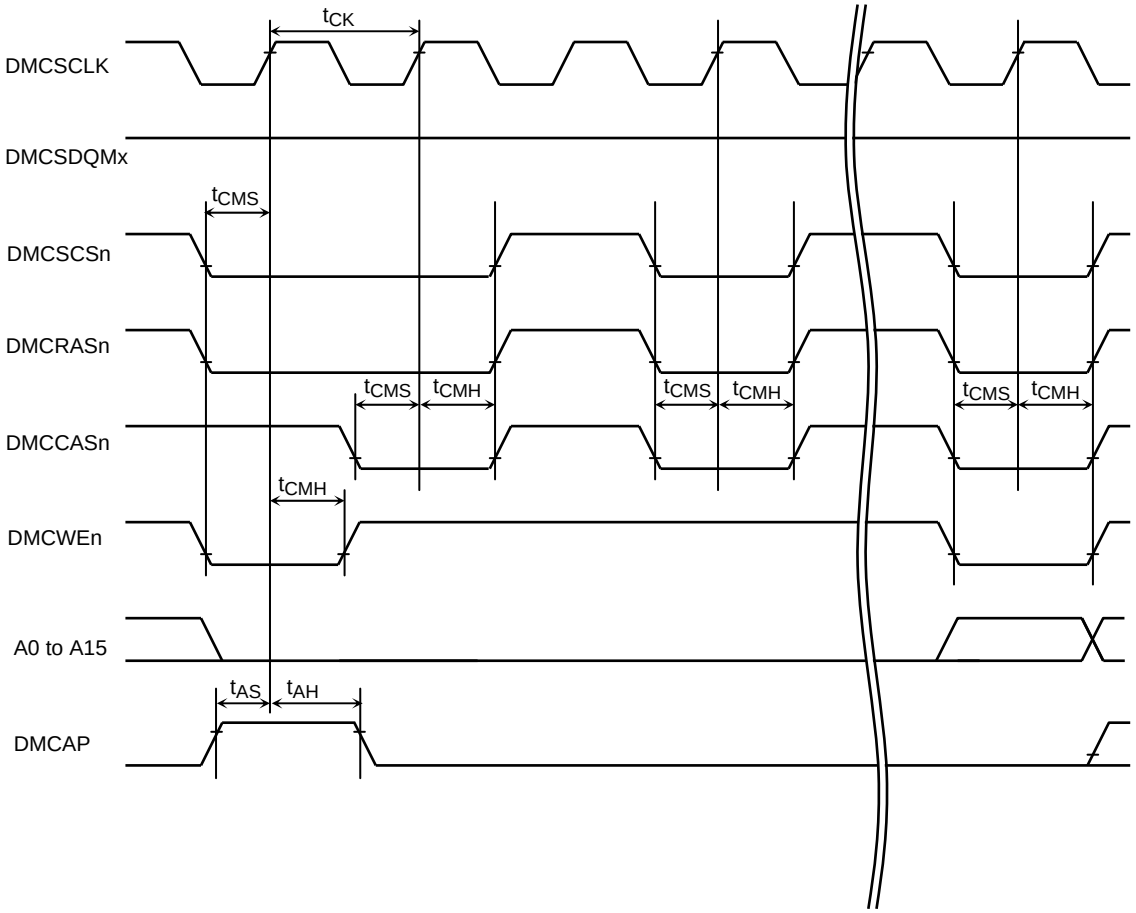
(3) SDRAM burst read timing (burst cycle start, CAS latency = 2)



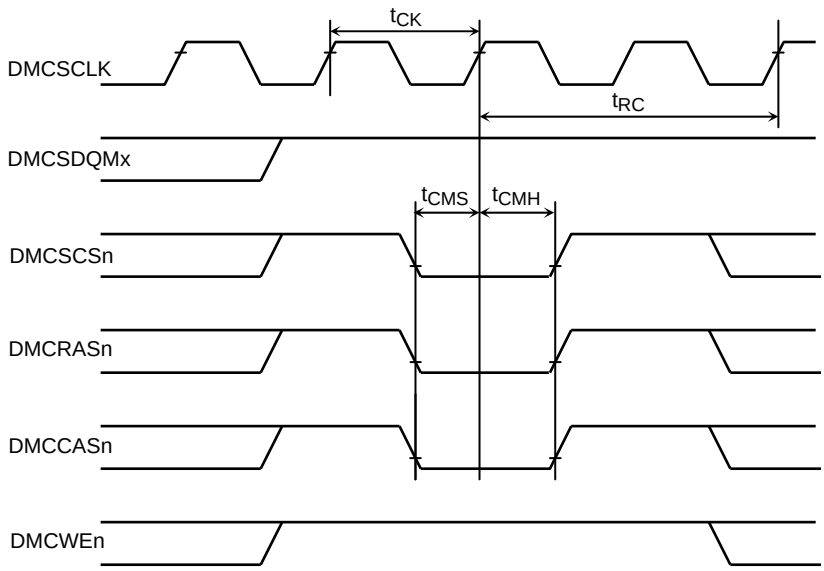
(4) SDRAM burst read timing (burst timing end)



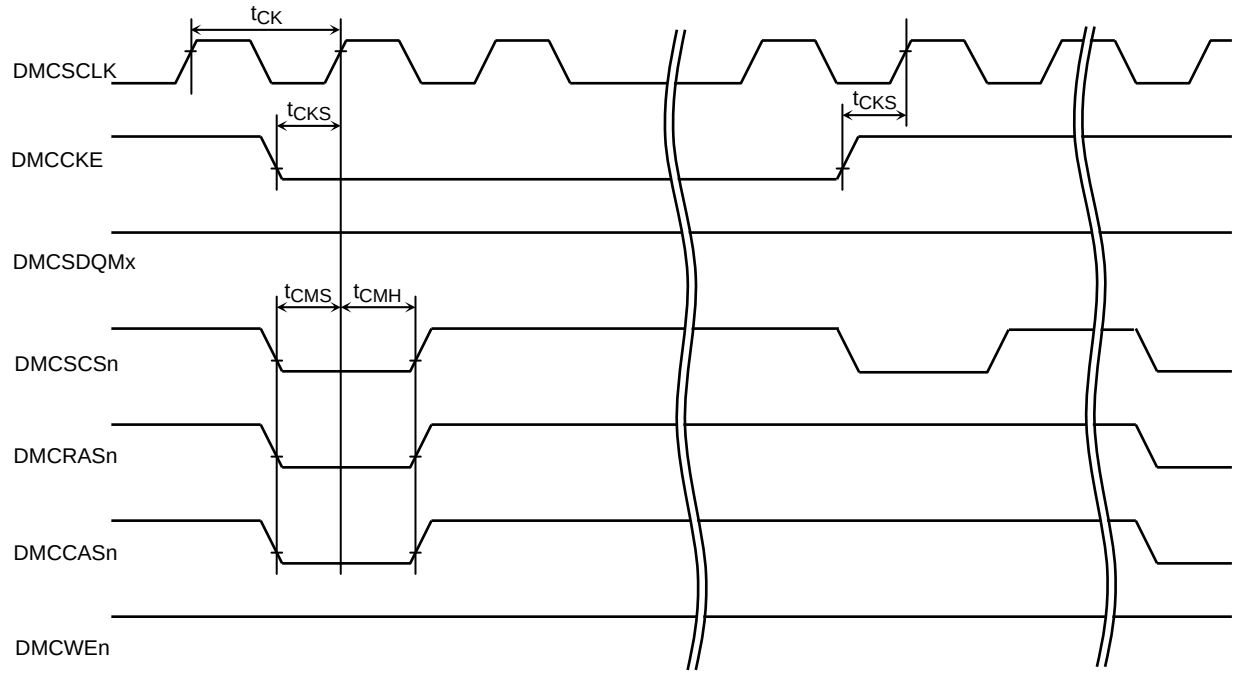
(5) SDRAM initialization timing



(6) SDRAM refresh timing



(7) SDRAM self-refresh timing



4.3.5 NAND Flash Controller AC Electrical Characteristics

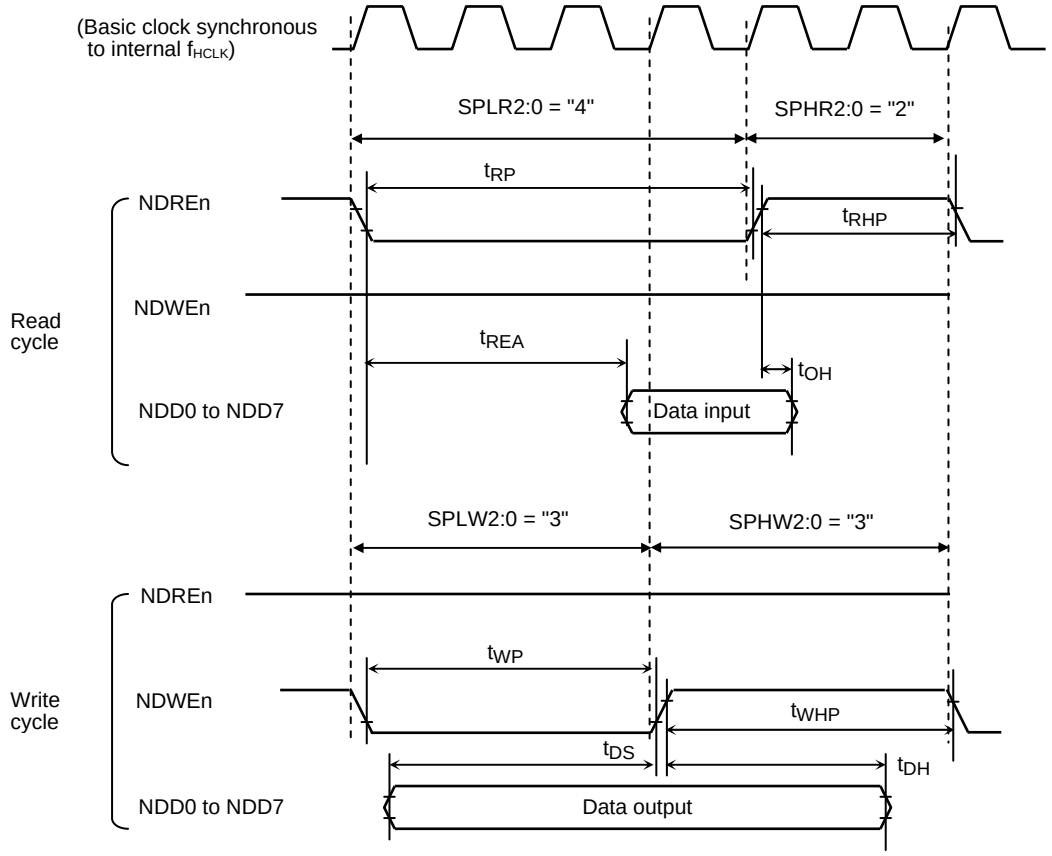
AC measurement conditions

- The letter “T” used in the equations in the table represents the period of internal bus frequency (f_{HCLK}), which is one-half of the CPU clock (f_{FCLK}).
- Output level: High = 0.7 × DVCC3IO, Low = 0.3 × DVCC3IO
- Input level: High = 0.9 × DVCC3IO, Low = 0.1 × DVCC3IO

Note 1: The “Equation” column in the table shows the specifications under the conditions DVCC3IO = 3.0 to 3.6 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

Note 2: The letter “n” in the equations represents the value set in NDFMCR2<SPLR[2:0]>, the letter “m” the value set in NDFMCR2<SPHR[2:0]>, the letter “k” the value in NDFMCR2<SPLW[2:0]>, and the letter “l” the value in NDFMCR2<SPHW[2:0]>. Care should be taken not to use values that produce negative results.

No.	Symbol	Parameter	Equation		100 MHz (n=3) (m=3) (k=3) (l=3)	96 MHz (n=3) (m=3) (k=3) (l=3)	Unit
			Min	Max			
1-1	t_{RC}	Read access cycle	(n + m) T		60.0	62.5	ns
1-2	t_{WC}	Write access cycle	(k + l) T		60.0	62.5	
2	t_{RP}	NDREn low level pulse width	(n) T -10.0		20.0	21.2	
3	t_{RHP}	NDREn high level pulse width	(m) T -10.0		20.0	21.2	
4	t_{REA}	NDREn data access time		(n) T – 14	16.0	17.2	
5	t_{OH}	Read data hold time	0		0	0	
6	t_{WP}	NDWEn low level pulse width	(k) T -10.0		20.0	21.2	
7	t_{WHP}	NDWEn high level pulse width	(l) T -10.0		20.0	21.2	
8	t_{DS}	Write data setup time	(k) T -10.0		20.0	21.2	
9	t_{DH}	Write data hold time	(l) T -10.0		20.0	21.2	



AC measurement conditions CL = 40 pF

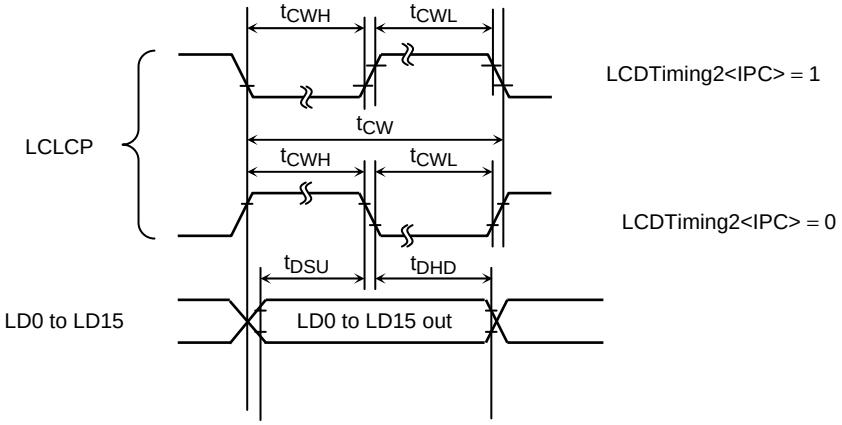
4.3.6 LCD Controller

AC measurement conditions

- The letter “T” used in the equations in the table represents the period of internal bus frequency (f_{HCLK}), which is one-half of the CPU clock (f_{FCLK}).
- Output level: High = 0.7 × DVCC3LCD, Low = 0.3 × DVCC3LCD

Note: The “Equation” column in the table shows the specifications under the conditions DVCC3LCD = 1.8 to 3.6 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

Parameter	Symbol	Equation		100 MHz (n=3)	96 MHz (n=3)	48MHz (n=2)	Unit
		Min	Max				
LCLCP clock period (nT)	t_{CW}	30		30.0	31.25	41.6	ns
LCLCP high level pulse width (including phase reversal)	t_{CWH}	When n = even (nT / 2) – 5.0 When n = odd ((n-1)T / 2) – 5.0		5.0	5.4	15.8	
LCLCP low level pulse width (including phase reversal)	t_{CWL}	When n = even (nT / 2) – 5 When n = odd ((n+1)T / 2) – 5		15.0	15.8	15.8	
Data valid to LCLCP fall (including phase reversal)	t_{DSU}	When n = even (nT / 2) – 6.0 When n = odd ((n-1)T / 2) – 6.0		4.0	4.4	14.8	
LCLCP fall to data hold (including phase reversal)	t_{DHD}	When n = even (nT / 2) – 6.0 When n = odd ((n+1)T / 2) – 6.0		14.0	14.8	14.8	



AC measurement conditions

- CL = 20 pF

Note: The letter “n” in the equations represents the value set in LCDTiming2<PCD_HI><PCD_LO> plus two. The following limitations apply to the “n” value depending on operating frequency.

Example 1: When f_{HCLK} = 100 MHz, LCDTiming2<PCD_HI>=0y00000 and <PCD_LO>=0y00001 (n= 3).

Example 2: When f_{HCLK} = 48 MHz, LCDTiming2<PCD_HI>=0y00000 and <PCD_LO>=0y00000 (n= 2).

4.3.7 SSP Controller

AC measurement conditions

- The letter “T” used in the equations in the table represents the period of internal bus frequency (f_{PCLK}), which is one-half of the CPU clock (f_{FCLK}).
  The internal bus cycle is T=10ns minimum value when the guaranteed temperature is 0 to 70 degree.
  The internal bus cycle is T=13.3ns minimum value when the guaranteed temperature is -20 to 85 degree.
- Output level: High = 0.7 × DVC3IOM, Low = 0.3 × DVCC3IO
- Input level: High = 0.9 × DVCC3IO, Low = 0.1× DVCC3IO

Note: The “Equation” column in the table shows the specifications under the conditions DVCC3IO = 3.0 to 3.6 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V.

Parameter	Symbol	Equation		PCLK 100MHz (m=6 n=12)	PCLK 96MHz (m=6 n=12)	Unit
		Min	Max			
SPxCLK Period (Master)	T_m	(m)T However more than 50ns		60.0	62.5	ns
SPxCLK Period (Slave)	T_s	(n)T		120.0	125.0	
SPxCLK rise up time	t_r		10.0	10.0	10.0	
SPxCLK fall down time	t_f		10.0	10.0	10.0	
Master mode: SPxCLK low level pulse width	t_{WLM}	(m)T / 2 - 7.0		23.0	24.3	
Master mode: SPxCLK high level pulse width	t_{WHM}	(m)T / 2 –7.0		23.0	24.3	
Slave mode: SPxCLK low level pulse width	t_{WLS}	(n)T / 2 - 7.0		53.0	55.5	
Slave mode: SPxCLK high level pulse width	t_{WHS}	(n)T / 2 - 7.0		53.0	55.5	
Master Mode: SPxCLK rise/fall to output data valid	t_{ODSM}		15.0	15.0	15.0	
Master Mode: SPxCLK rise/fall to output data hold	t_{ODHM}	(m)T/2 -10		20.0	21.3	
Master Mode: SPxCLK rise/fall to input data valid delay time	t_{IDSM}		(m)T /2 - 20	10.0	11.2	
Master Mode: SPxCLK rise/fall to input data hold	t_{IDHM}	5.0		5.0	5.0	
Master Mode: SPxFSS valid to SPxCLK rise/fall	t_{OFSM}	(m)T -10	(m)T + 10	50~70	52.5~72.5	
Slave mode: SPxCLK rise/fall to output data valid delay time	t_{ODSS}		(3T) + 25	55.0	56.3	
Slave mode: SPxCLK rise/fall to output data hold	t_{ODHS} Note1)	(n)T /2 + (2T)		80.0	83.3	
Slave mode: SPxCLK rise/fall to input data valid delay time	t_{IDSS}		(n)T /2 + (3T) - 10.0	80.0	83.8	
Slave mode: SPxCLK rise/fall to input data hold	t_{IDHS}	(3T) + 10		40.0	41.3	
Slave mode: SPxFSS valid to SPxCLK rise/fall	t_{OFSS}	(n)T		120	125	

Note1: Baud rate Clock is set under below condition

Master mode

$m = (<CPSDVSR> \times (1 \pm <SCR>)) = f_{PCLK}/SPxCLK$

<CPSDVR> is set only even number and “m” must set during $65204 \geq m \geq 2$

Slave Mode

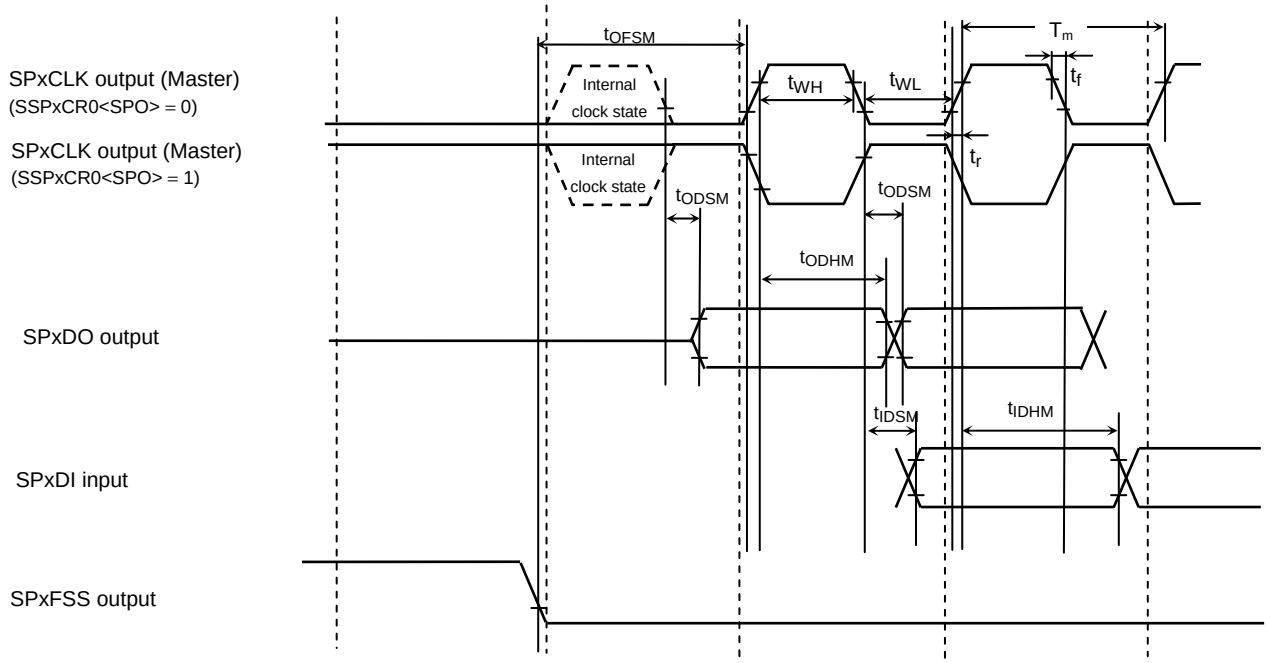
$n = f_{PCLK}/SPxCLK \quad (65204 \geq n \geq 12)$

AC measurement conditions:

Load capacitance CL = 25 pF

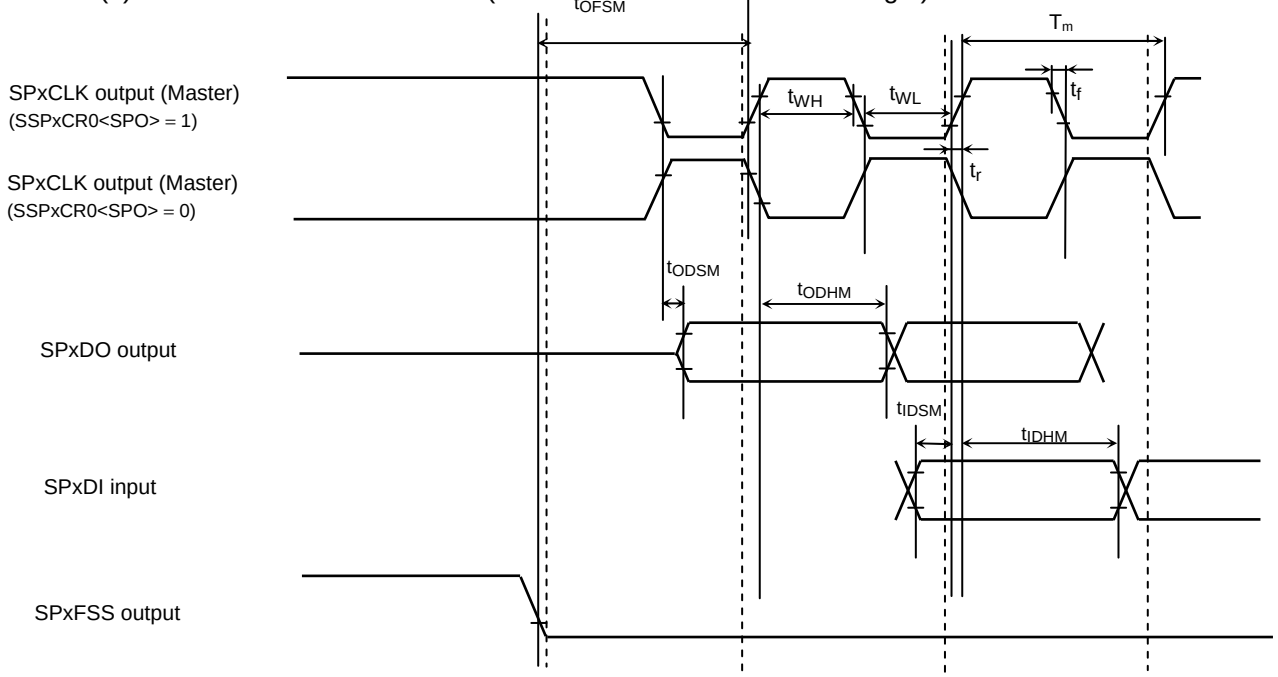
- SSP SPI mode (Master)
  * $f_{PCLK} \geq 2 \times$ SPxCLK (max)
  * $f_{PCLK} \geq 65204 \times$ SPxCLK (min)

(1) Master SSPxCR0<SPH> = 0 (Data is latched on the first edge.)



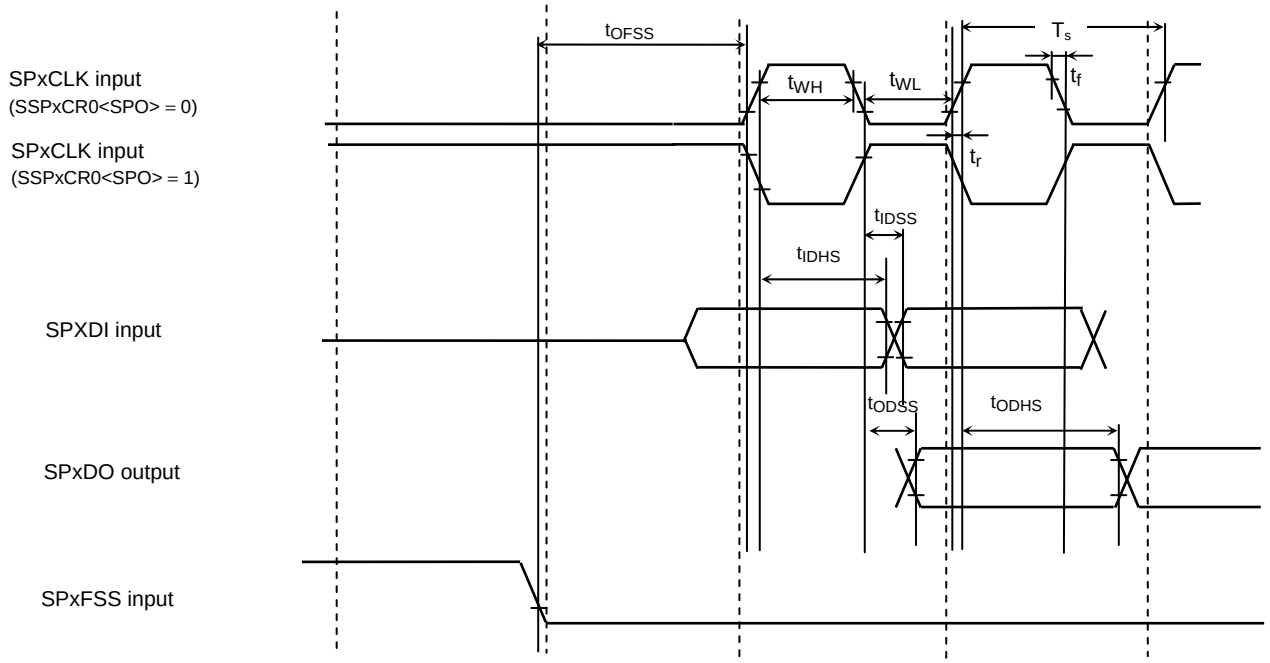
- SSP SPI mode (Master)

(2) Master SSPxCR0<SPH> = 1 (Data is latched on the second edge.)



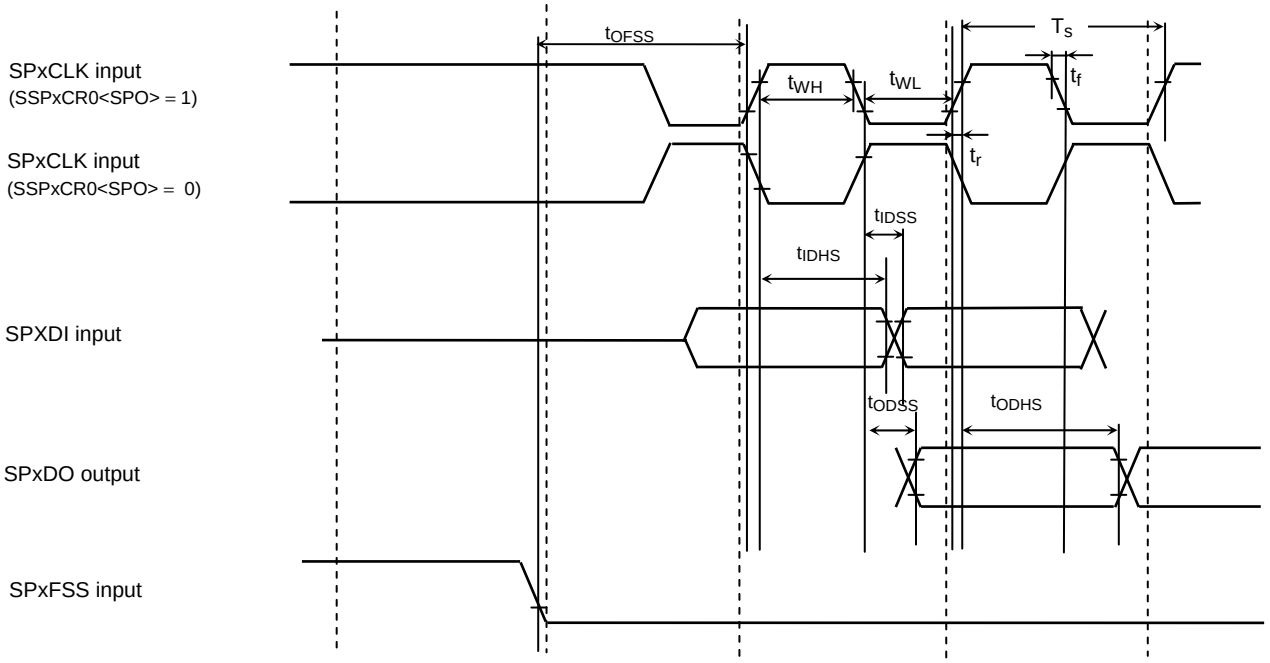
- SSP SPI mode (Slave)
  - $f_{PCLK} \geq 12 \times$ SPxCLK (max)
  - * $f_{PCLK} \geq 65204 \times$ SPxCLK (min)

(3) Slave SSPxCR0<SPH> = 0 (Data is latched on the first edge.)



- SSP SPI mode (Slave)

(4) Slave SSPxCR0<SPH> = 1 (Data is latched on the second edge.)



4.3.8 I^2S

AC measurement conditions

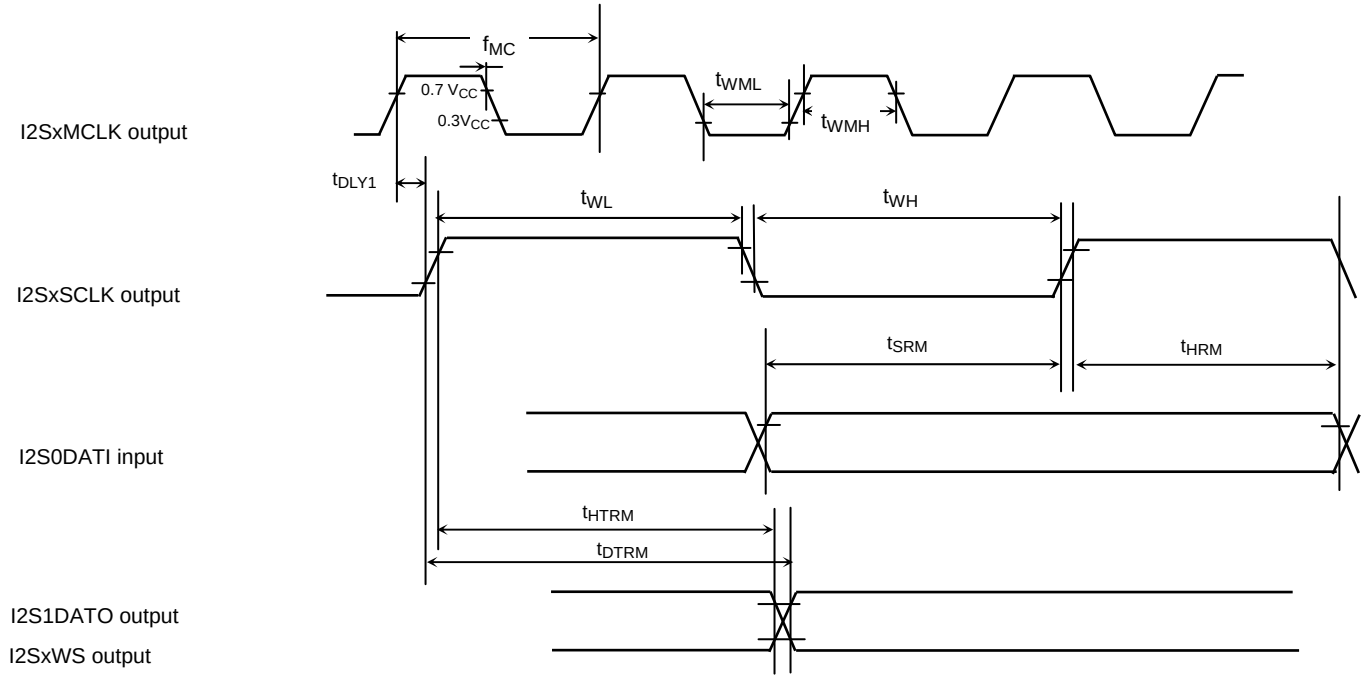
- The letter “T” used in the equations in the table represents the period of internal bus frequency (f_{PCLK}), which is one-half of the CPU clock (f_{FCLK}). $f_{PCLK} \geq 16 \times$ I2SxSCLK.
- The letter “t” used in the equations in the table represents the period of frequency (f_{OSCH}).
- Output level: High = 0.7 × DVCC3I2S, Low = 0.3 × DVCC3I2S
- Input level: High = 0.9 × DVCC3I2S, Low = 0.1 × DVCC3I2S

Note: The “Equation” column in the table shows the specifications under the conditions DVCC3I2S = 1.8 to 3.6 V and DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6 V. In slave mode, the stabilization time is required after I2SxCLK input.

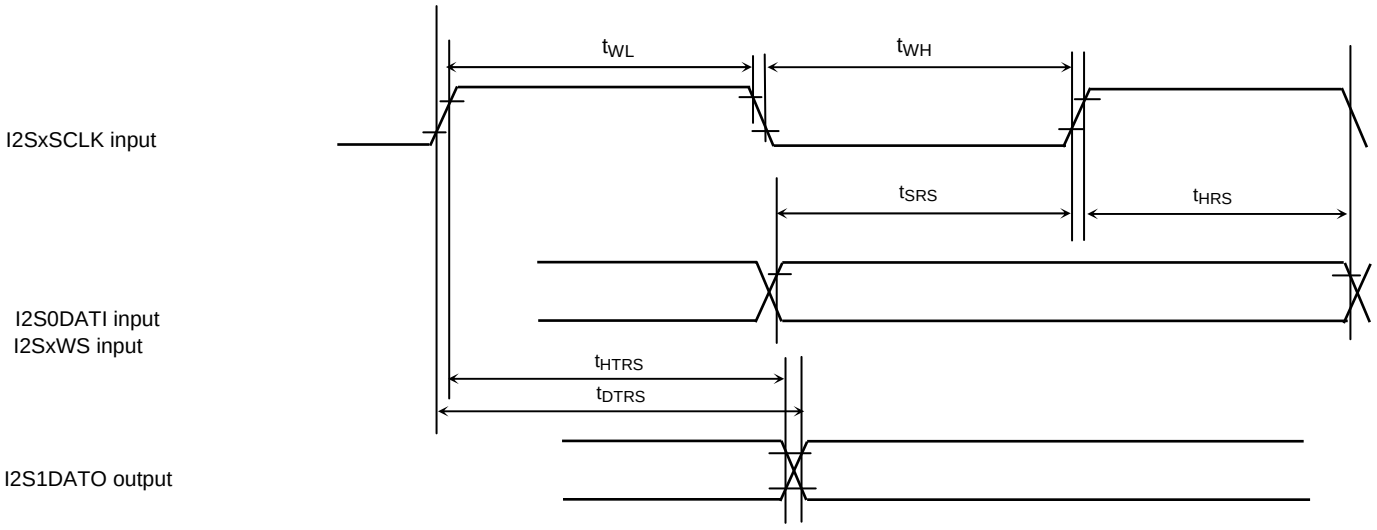
Parameter	Symbol	Equation		100MHz	96 MHz	Unit
		Min	Max			
I2SMCLK Out f_{OSCH}	f_{MC}	t /4	25	24	24	MHz
I2SxMCLK high level pulse width	t_{WMH}	10	–	10	10	ns
I2SxMCLK low level pulse width	t_{WML}	10	–	10	10	
I2SxCLK clock output period	f_{SCKM}	333	–	333	333	
I2SxCLK clock input period	f_{SCKS}	160	–	160	160	
I2SxCLK high level pulse width	t_{WH}	0.45 f_{SCKM}	–	149	149	
I2SxCLK low level pulse width	t_{WL}	0.45 f_{SCKM}	–	149	149	
Master mode: I2S1DATO, I2SxWS hold time	t_{HTRM}	0.5 f_{SCKM} + 2T	–	186	187	
Master mode: I2S1DATO, I2SxWS delay time	t_{DTRM}	–	0.5 f_{SCKM} + 3T + 10	206	207	
Master mode: I2S0DATI setup time	t_{SRM}	10.0	–	10	10	
Master mode: I2S0DATI hold time	t_{HRM}	0.2 f_{SCKM}	–	66	66	
Master mode: I2SxMCLK, I2SxSCLK delay time	t_{DLY1}	–	10.0	10	10	
Slave mode: I2S1DATO, I2SxWS hold time	t_{HTRS}	0.5 f_{SCKS} + 2T	–	100	100	
Slave mode: I2S1DATO, I2SxWS delay time	t_{DTRS}	–	0.8 f_{SCKS} + 3T + 20	178	179	
Slave mode: I2S0DATI setup time	t_{SRS}	10.0	–	10	10	
Slave mode I2S0DATI hold time	t_{HRS}	0.2 f_{SCKS}	–	32	32	

AC measurement conditions: Load capacitance CL = 25 pF

(1) I²S master mode



(2) I²S slave mode



4.4 AD Conversion Characteristics

Note: “Caltulation” of following table is effective in a range of AVCC3AD = 3.0V to 3.6V, DVCC1A = DVCC1B = DVCC1C = 1.4 to 1.6V.

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Analog reference voltage (+)	VREFH		AVCC3AD	AVCC3AD	AVCC3AD	V
Analog reference voltage (–)	VREFL		DVSSCOMn	DVSSCOMn	DVSSCOMn	
AD converter power supply voltage	AVCC3AD		3.0	3.3	3.6	
AD converter GND	AVSS		DVSSCOMn	DVSSCOMn	DVSSCOMn	
Analog input voltage	AVIN		VREFL		VREFH	
Analog reference voltage	IREFON	<VREFON> = 1		2.1	3.5	mA
Power supply current	IREFOFF	<VREFON> = 0		0.1	10	μA
Full Scale Error	EFULL			+1	-1 to +4	LSB
Offset Error	EOFF			-3	-4 to +1	LSB
Differential Error	EDNL			-1 to +2	±2	LSB
Integral Error	EINL			-2 to +3	±3	LSB

Note 1: Error = (“conversion result” – “theoretical value”)

1 LSB = (VREFH – VREFL)/1024[V]

Note 2: The quantization error does not include.

Note 3: Minimum operating frequency

The minimum operating clock and maximum operating clock (ADCLK) of the AD converter is 3 MHz and 33 MHz, respectively. (3MHz ≤ ADCLK ≤ 33MHz)

Minimum conversion time is 1.39μs at 33MHz, and maximum conversion time is 15.3μs at 3MHz.

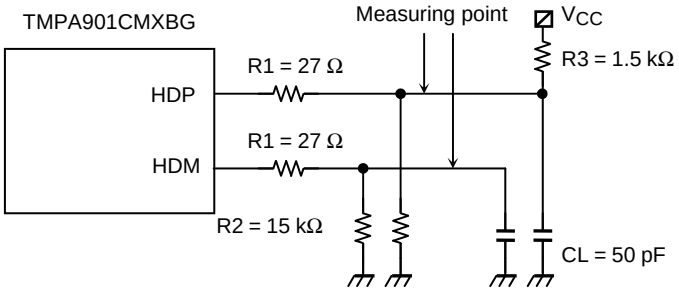
4.5 USB Host Controller

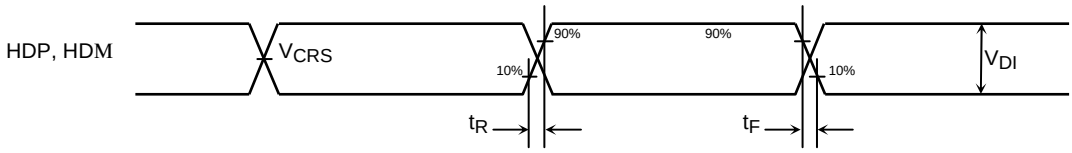
AVCC3H= 3.3 ± 0.3 V / f_{USB} = 48 MHz

項目	記号	Min	Max	単位	備考
HDP, HDM rising time	t_R	4	20	nS	Full Speed
HDP, HDM falling time	t_F	4	20		Full Speed
Differential Common mode range	VDI	0.2		V	|(HDP) –(HDM)|
Output signal crossover voltage	V_{CRS}	1.3	2.0		V

AC measurement conditions

＜Full Speed＞



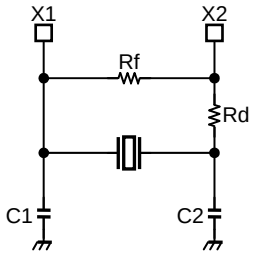


4.6 Recommended Oscillation Circuit

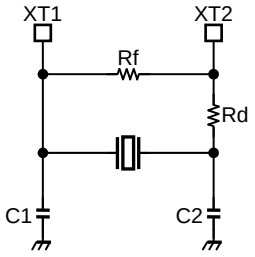
The TMPA901CM is evaluated by using the resonators shown below. Please use this information when selecting the resonator to be used.

Note: The total load capacitance of the oscillation pins is the sum of the external (or internal) load capacitances C1and C2 and the stray capacitance on the circuit board. Even if the recommended C1 and C2 constants are used, the total load capacitance may vary with each board. In board design, the patterns around the oscillation circuit must be as short as possible. We also recommend that oscillation evaluations be performed on the actual board to be used.

(1) Connection example



Connection diagram of high-frequency oscillator



Connection diagram of low-frequency oscillator

(2) Recommended ceramic resonator: KYOCERA KINSEKI Corporation

The table below shows the recommended circuit constants for the high-frequency oscillation circuit.

MCU	Oscillation frequency [MHZ]	Type	Recommended resonator	Recommended constants				Recommended operating conditions	
				C1 [pF]	C2 [pF]	Rd [Ω]	Rf [Ω]	Power supply voltage range [V]	Temperature range [°C]
TMPA900CMXBG	27.000	SMD	CX2520SB27000B0HLQZ1	6	6	0	Open	1.4 to 1.6	-20 to 85
	24.000	SMD	CX2520SB24000C0HLQZ1	7	7				

Note 1: Constants C1 and C2 indicates values for the built-in load capacitance type.

Note 2: The part numbers and specifications of Kyocera's products are updated as occasion requires. For details, please check the website of Murata.
http://global.kyocera.com/

Note 3: When the ceramic resonator is used for high-frequency oscillator circuit, USB device and host controller requires using the clock in high precision. In this case, the clock of X1USB pin should be used the clock precision is 24MHz ±100ppm.

(3) Recommended crystal low-frequency resonator: KYOCERA KINSEKI Corporation

The table below shows the recommended circuit constants of the low-frequency oscillation circuit.

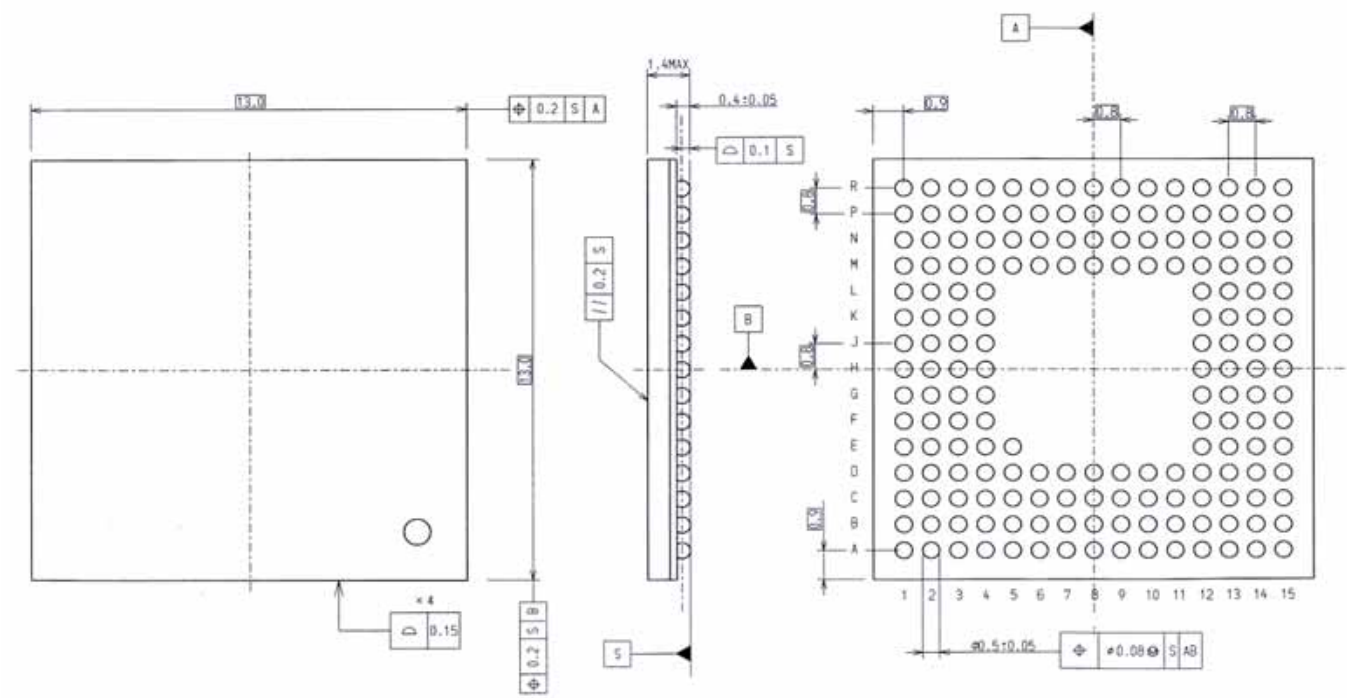
MCU	Oscillation frequency [kHz]	Recommended resonator	Recommended constants				Recommended operating conditions	
			C1 [pF]	C2 [pF]	Rd [Ω]	Rf [Ω]	Power supply voltage range [V]	Temperature range [°C]
TMPA901CMXBG	32.768	ST3215SB	7	7	820	Built-In	3.0 to 3.6	-20 to 85

Note: The part numbers and specifications of KYOCERA KINSEKI's products are updated as occasion requires. For details, please check the website of KYOCERA KINSEKI.
http://global.kyocera.com/

5. Package Dimensions

Package name: P-FBGA177-1313-0.8C4

Unit: mm



RESTRICTIONS ON PRODUCT USE

- Toshiba Corporation, and its subsidiaries and affiliates (collectively “TOSHIBA”), reserve the right to make changes to the information in this document, and related hardware, software and systems (collectively “Product”) without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA’s written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**
- Product is intended for use in general electronics applications (e.g., computers, personal equipment, office equipment, measuring equipment, industrial robots and home electronics appliances) or for specific applications as expressly stated in this document. Product is neither intended nor warranted for use in equipment or systems that require extraordinarily high levels of quality and/or reliability and/or a malfunction or failure of which may cause loss of human life, bodily injury, serious property damage or serious public impact (“Unintended Use”). Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, devices related to electric power, and equipment used in finance-related fields. Do not use Product for Unintended Use unless specifically permitted in this document.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. TOSHIBA assumes no liability for damages or losses occurring as a result of noncompliance with applicable laws and regulations.